Statistical Static Timing Analysis of Nonzero Clock Skew Circuits

A Thesis

Submitted to the Faculty

of

Drexel University

by

Shannon Michael Kurtas

in partial fulfillment of the

requirements for the degree

of

Master of Science in Computer Engineering

June 2007

Dedications

To my family for their unconditional love and support.

Acknowledgments

I would first like to thank my advisor, Dr. Baris Taskin, for his guidance and supervision of this work, as well as the members of my thesis committee, Drs. Tim Kurzweg and Suryadevara Basavaiah. It has been a pleasure working with them and studying under their tutelage. I'd also like to acknowledge other faculty for their useful discussions regarding this project, including Prof. Kurt Schmidt and Dr. Pat Henry.

I'm indebted to my colleagues and supervisors at Intel, particularly Greg Vaccaro, Sal Bhimji, and Syed Rahman, for introducing me to the semiconductor industry and providing priceless experience that has accelerated my education more than any course could have. Similarly, I thank my coworkers and managers from earlier internships at Metrologic Instruements – BJ Zhu, Barry Schwartz, and Jacky Liu – for giving me interesting and challenging work so early on.

I've been fortunate to make wonderful friends over the past five years who have provided great memories and who have been there through the thick and thin. Thank you for everything. I'd also like to thank a number of other great people who I've had the opportunity to collaborate with in various capacities, namely Daniela Ascarelli, Dr. Spanier, and all the folks from Drexel Cycling.

Finally, my most sincere and profound gratitude goes to my Aunt Janet and Uncle Harry for the encouragement and direction that they have provided from day one. Your boundless support has been crucial in more ways than can be accurately conveyed here.

Table of Contents

List of Tables

List of Figures

Abstract

Statistical Static Timing Analysis of Nonzero Clock Skew Circuits
Shannon Michael Kurtas
Baris Taskin, Ph.D.

As microprocessor and ASIC manufacturers continue to push the limits of transistor sizing into the sub-100nm regime, variations in the manufacturing process lead to increased uncertainty about the exact geometry and performance of the resulting devices. Traditional corner-based Static Timing Analysis (STA) assumes worst-case values for process parameters such as transistor channel length and threshold voltage when verifying integrated circuit timing performance. This has become unreasonably pessimistic and causes over-design that degrades full-chip performance, wastes engineering effort, and erodes profits while providing negligible yield improvement. Recently, Statistical Static Timing Analysis (SSTA) methods, which model process variations statistically as probability distribution functions (PDFs) rather than deterministically, have emerged to more accurately portray integrated circuit performance. This analysis has been thoroughly performed on traditional zero clock skew circuits where the synchronizing clock signal is assumed to arrive in phase with respect to each register. However, designers will often schedule the clock skew to different registers in order to decrease the minimum clock period of the entire circuit. Clock skew scheduling (CSS) imparts very different timing constraints that are based, in part, on the topology of the circuit. In this thesis, SSTA is applied to nonzero clock skew circuits in order to determine the accuracy improvement relative to their zero skew counterparts, and also to assess how the results of skew scheduling might be impacted with more accurate statistical modeling. For 99.7% timing yield (3σ variation), SSTA is observed to improve the accuracy, and therefore increase the timing margin, of nonzero clock skew circuits by up to 2.5x, and on average by 1.3x, the amount seen by zero skew circuits.

Chapter 1. Introduction

1.1 Background

Microprocessor and ASIC designers constantly weigh the tradeoffs of area, delay, and power in state of the art IC design. Although all of these criteria are highly critical, the performance of IC chips is traditionally correlated to the maximum frequency at which they can be operated. The physical design steps of IC development, whether completed by automated tools or through designer interaction, are formulated to meet a desired operating frequency. Despite this vast amount of emphasis and planning, the final product can still be far from the targeted timing budget due to simulatenously increasing product requirements and manufacturing variability. Variations in the geometry and electronic properties of the transistors within the chip inevitably occur during fabrication and significantly impact their timing. In order to compensate for these variations in process parameters, designs go through static timing analysis (STA) as part of a post-processing performance verification CAD flow. This timing analysis establishes a safety factor such that, even with this unavoidable process variation, the chips manufactured will function as desired. Because of worsening variation in deep sub-micron (DSM) design, the safety factors introduced by STA have become unreasonably pessimistic and, as a result, statistical techniques such as statistical static timing analysis (SSTA) are emerging in order to more accurately portray circuit performance.

1.1.1 Sources of Variation

A typical Si-based semiconductor process begins with Silicon ingots being sliced into a “lot” of thin “wafers,” which are then processed into tiny “die” with generally identical functionality. Process parameters vary on each die and wafer due to imprecisions in the manufacturing process. Process variations are typically characterizated as either being inter-die or intra-die. Inter-die variations occur in devices between multiple die on a wafer. Intra-die variations, conversely, occur between multiple devices on the same die, as shown in Figure 1.1.

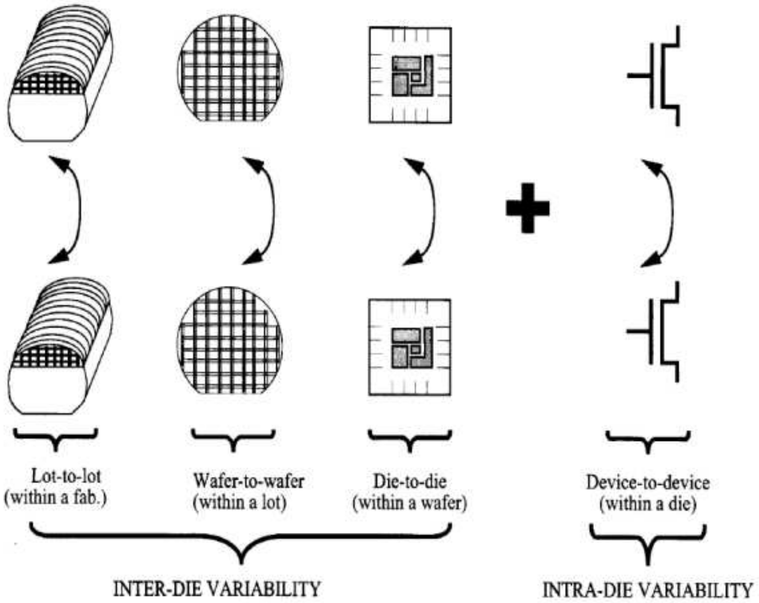

Figure 1.1: Classification of Process Variation

Intra-die variation is further broken down in to systematic and random variation. Systematic variation implies spatial correlation between devices, whereas random variation is independent for each device regardless of location [1]. Lithographic optics, for instance, are known to produce systematic variation in MOSFET (Figure 1.2) channel length (L_e) across a die. Random dopant fluctuations are responsible for varying transistor threshold gate voltage (V_T). Although this is not an exhaustive list of the parameters which vary, channel length and threshold voltage have the most significant impact on transistor performance and, as is common, are the two sources of variation that will be considered in this study. As will be described later in Chapter 3, the framework used is easily extensible for any number of systematic and random parameters.

1.1.2 Future Implications of Variation for Microprocessors & ASICs

As the minimum feature size of VLSI circuits continues to shrink, process variations become significantly worse. The amount of variation in channel length and threshold voltage can be represented as a growing percentage of their nominal values. For example, a ±.1V variation in V_T is

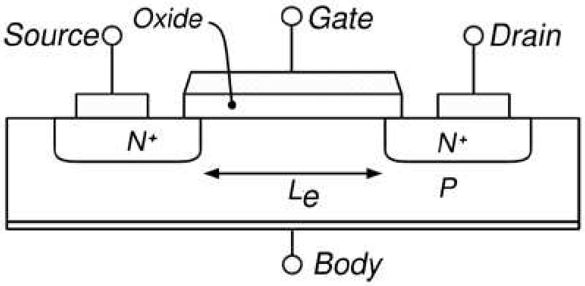

Figure 1.2: MOSFET Cross Section

less problematic in a process where V_T is nominally 1V(10%) as opposed to one where V_T is nominally .7V(14.3%). Although worst case variation has previously corresponded to less than a 10% deviation from the nominal value of most process parameters, it is known to be worse than 15% for most sources of variation in deep sub-micron circuits [2]. As it has been succinctly put, "critical dimensions are scaling faster than our control of them, and the variability of these dimensions is proportionately increasing [3]." These variations greatly complicate the design and verfication of IC designs in DSM technologies. The accelerated time-to-market demands in both the microprocessor and ASIC markets have exacerbated the need for efficient and reliable timing analysis that can accurately deal with process variability. For approximately 20 years static timing analysis (STA) has been able to meet that need. However, as a result of worsening variation, the deterministic guardbanding of STA has resulted in undesireable degrees of pessimism. STA does not take into account the known *probabilities* of different types of variation, and therefore does not give designers a reliable picture of what percentage of the manufactured chips will operate at which clock frequencies. STA is also considered "risky" because, although the worst case figures in deterministic guardbanding are meant to ensure guaranteed operation at the target frequency, it is not practical to conduct this deterministic bounding for all variation corners. In order to address these pitfalls, statistical static timing analysis (SSTA) [4, 5] has been proposed. In SSTA, variations are represented by random variables rather than deterministic best and worst case values. Statistical modeling and computation therefore provide a more accurate assessment of circuit performance which includes the probability of the circuit performing at any given frequency. The widespread use of statistical design and analysis will be essential within the next decade in addressing Design for Manufacturability (DFM) concerns of deep sub-micron technologies.

The information provided by SSTA is of importance to both microprocessor and ASIC designers, despite their different profit functions as depicted in Figure 1.3. Across a wafer of microprocessors, some of the manufactured chips are able to operate at "b1" while others may have to operate at "a1," due to process variations. The chips at a1, although slower, can be still be sold at a lower profit; that is, they can be *speed binned* at a lower frequency. Statistical *yield* information provides a picture of how a chip would be speed binned prior to manufacturing, and thereby enables informed decision making regarding changes in design or chip specification [6]. Similarly, ASIC manufacturers need to know the probability of a design having a clock period at $> b_2$, as such a chip could not be sold and would simply be discarded. SSTA therefore enables high-performance targeting simultaneously with precise risk management [3, 7].

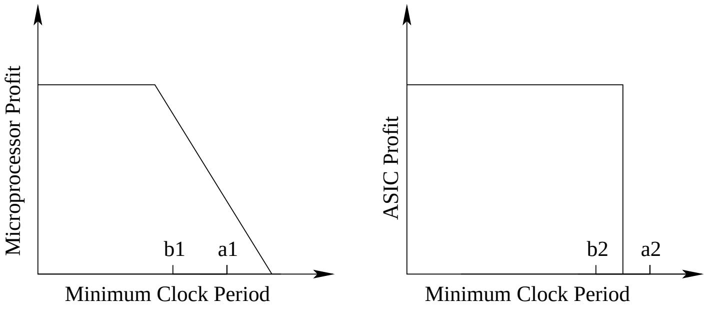

Figure 1.3: Microprocessor vs. ASIC Profit

1.2 Problem Statement & Thesis Contributions

Within the last decade, SSTA methods have been discovered and have reached a certain level of maturity. Researchers are focusing on utilizing these methods in performing circuit optimization and simulatenously analyzing timing, power, and area constraints [8, 9, 10]. SSTA has been thoroughly applied to the timing analysis and characterization of zero clock skew (ZCS) circuits. ZCS circuits work on the assumption that the synchronizing clock signal arriving at all of the registers throughout the circuit is in phase at each of these points. In other words, they have "zero skew" where skew is defined as the relative difference in clock arrival time between registers. Uncertainty in this assumption is traditionally handled by further deterministic guardbanding. However, a significant post-processing step for both microprocessors and ASICs is nonzero clock skew scheduling (CSS). In nonzero clock skew (NZCS) circuits, clock signal delays are intentionally manipulated in order to further improve the circuit's maximum operating frequency [11, 12]. The factors which limit the maximum operating frequency f_{Max} (or minimum clock period T_{min}) are quite different between ZCS and NZCS circuits.

In this thesis, SSTA is applied to nonzero clock skew circuits in order to determine their relative improvement in T_{min} that can be uncovered with statistical analysis. The results will demonstrate whether the benefits of skew scheduling are enhanced or lessened by more accurate modeling, and if the NZCS clock period is limited by different gates or paths in the statistical vs. deteministic domain.

1.3 Organization of Thesis

The remainder of this thesis is organized as follows. In Chapter 2, traditional static timing analysis (STA) for zero clock skew circuits is reviewed. In order to help the SSTA discussion, are the differences between path-based and block-based analysis are presented and the timing limits for the minimum clock period are defined. In Chapter 3, statistical static timing analysis (SSTA) and the relevant underlying mathematics are introduced. Also, the accuracy improvement of statistical modeling is highlighted by examples. In Chapter 4, nonzero clock skew circuits are discussed and the timing limits that they introduce are examined. Chapter 5 summarizes the experimental setup for this work including the 90nm cell library generation and characterization. In Chapter 6, statistical timing results are presented for both zero and nonzero clock skew circuits. Finally in Chapter 7, these results are discussed thoroughly, conclusions are drawn, and future improvements and extensions to this work are proposed.

Chapter 2. Static Timing Analysis

Determining the clock frequency at which a circuit can operate requires being able to measure propagation delays at different points within that circuit. Although transient analysis simulations with Spice can provide extremely accurate measurements by taking in to account all of the physical intracacies of the underlying transistors, the amount of computation time needed to perform these simulations on an entire circuit quickly becomes impractical for larger circuits. By using simplified delay models for logic gates and graph representation, static timing analysis aims to efficiently compute the slowest, frequency limiting path throughout a circuit (the *critical path*) [13].

2.1 Corner Based Cell Delay Models

Static timing analysis (STA) guardbands against process variations by assuming best and worst case values for the process parameters in question. This is shown graphically in Figure 2.1 for two process parameters X_1 and X_2 [14] (any number of random variables can be included in the same manner). The "nominal" corner occurs when both parameters assume their mean values. The worst case "slow" or "ss" and best case "fast" or "ff" corners, conversely, occur when both parameters assume their worst and best case values, respectively [15].

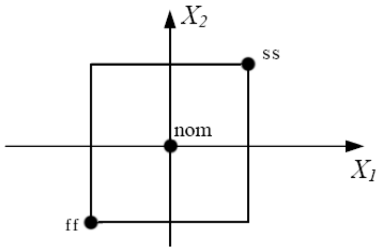

Figure 2.1: Worst Case Corner Analysis

Table 2.1: Example 90nm Process Corner Values

	ff	nom	ss
L_e	81nm	90nm	99nm
V_T	270mV	300mV	330mV

For example, assuming 10% threshold voltage V_T and channel length L_e variation in a 90nm process, these corners would correspond to the values listed in Table 2.1.

Each combinational logic gate within a cell library is characterized using this corner analysis method. Simulations are performed for each gate at a range of output load capacitances (C_L) and input transition slopes (tt_{in}) in order to find the minimum and maximum propagation delay $[\tau_{Pmin}, \tau_{Pmax}]$ of the gate at this particular (C_L, tt_{in}) combination at the appropriate corner. These values typically go in to a table or database such that once a circuit configuration is known, the exact loading and slope can be used to interpolate the exact τ_{Pmin} and τ_{Pmax} for each instantiation of a cell as in [16].

2.2 Circuit Representation

Synchronous circuits consist of combinational logic gates and sequential gates (simply called registers). For simplicity, all registers in this study are assumed to be edge-triggered flip flops, which are the most common type of registers. A local data path is formed between any two sequentially adjacent registers R_i and R_f connected by some collection of combinational logic gates. This is shown in Figure 2.2, where the combinational gates are lumped into a single combinational block.

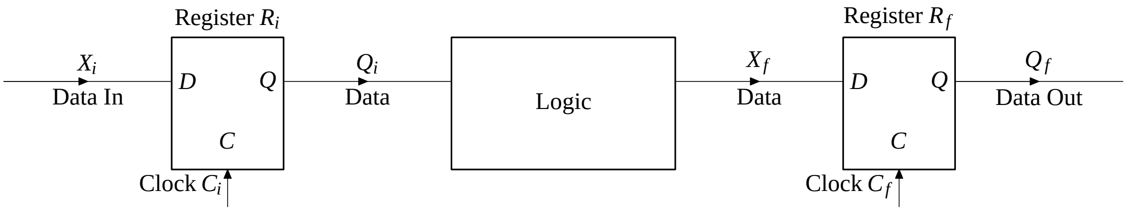

Figure 2.2: Local Data Path

A circuit is typically represented in directed graph form where each gate or register is represented by a vertex and each wire by an edge. In a typical circuit, several local data paths between a register pair R_i and R_f will exist, each with their own minimum and maximum total propagation delays, i.e. $\left[d_{Pm}^{if_x}, d_{PM}^{if_x}\right]$ for a local data path p^{if_x}, where x is used to differentiate between these multiple local data paths. This is depicted in Figure 2.3.

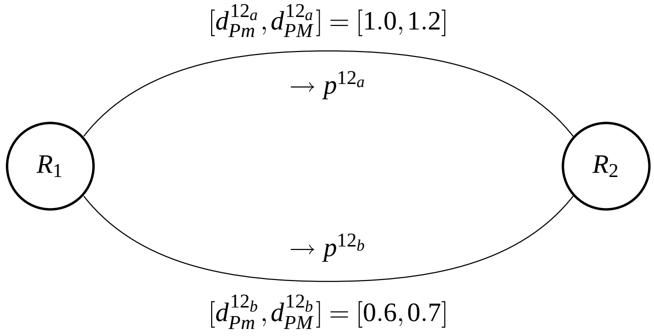
$[d_{Pm}^{12_a}, d_{PM}^{12_a}] = [1.0, 1.2]$
$\rightarrow p^{12_a}$
R_1
R_2
$\rightarrow p^{12_b}$
$[d_{Pm}^{12_b}, d_{PM}^{12_b}] = [0.6, 0.7]$

Figure 2.3: Multiple Local Data Paths

A *reduced graph* is defined as a directed graph where each register is represented by a vertex, and local data paths as a whole are represented by an edge, as shown in Figure 2.4. In this manner, an edge labeled p^{if} corresonds to a particular local data path from register R_i to register R_f [17]. When several local data paths between a register pair exist, the minimum and maximum data propagation time $\left[d_{Pm}^{if}, d_{PM}^{if}\right]$ between registers R_i and R_f is defined as the minimum and maximum data propagation times of all such paths as seen in Equations 2.1 and 2.2. For example, $\left[d_{Pm}^{if}, d_{PM}^{if}\right]$ for all of the local data paths between registers R_i and R_f from Figure 2.3 would evaluate to be $[0.6, 1.2]$ as see in Figure 2.3.

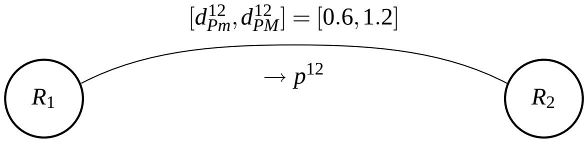

Figure 2.4: Reduced Graph of Local Data Path

$$d^{if}_{Pm} = \underset{\forall p^{if}}{min} \left[d^{if_x}_{Pm} \right] \tag{2.1}$$

$$d^{if}_{PM} = \underset{\forall p^{if}}{max} \left[d^{if_x}_{PM} \right] \tag{2.2}$$

2.3 Calculating Local Path Delays: MAX, MIN, & ADD Operations

For deterministic static timing analysis, calculating local path delays is a straightforward process. Minimum and maximum total data propagation delays $\left[d^{if}_{Pm}, d^{if}_{PM} \right]$ for a specific path p^{if_x} between registers R_i and R_f are the sum of the "ss" and "ff" corner logic gate propagation delays (τ_{Pmin}, τ_{PMax}) along that path, respectively, as seen in Equations 2.3 and 2.4.

$$d^{if_x}_{Pm} = \sum_{\forall gates \subset p^{if_x}} \tau_{Pmin} \tag{2.3}$$

$$d^{if_x}_{PM} = \sum_{\forall gates \subset p^{if_x}} \tau_{PMax} \tag{2.4}$$

Rather than enumerating all of the possible data paths and adding cell propagation delays repetitively for gates that exist in multiple paths, a *topological sort* of the intermediate logic gates between registers can be found for acyclic circuits. This topological sort is an ordering of gates to visit in the circuit graph that can be used to calculate the minimum and maximum *arrival time* (AT_{min}, AT_{Max}) at each gate in the circuit, such that no gate is visited before all of its predecessors have been visited. The procedure for finding a topological sort is displayed in Figure 2.5.

This topological sort is traversed in order to find the arrival times $[AT^g_{min}, AT^g_{Max}]$ for each gate in the circuit network. These calculations require the use of three essential static timing analysis functions, $ADD()$, $MIN()$, and $MAX()$. For a two input logic gate $g = f(a, b)$ with inputs a and b, the minimum and maximum arrival times $[AT^g_{min}, AT^g_{Max}]$ are calculated as in Equations 2.5 and 2.6. Any number of inputs can be handled by using nested calls to the $MIN()$ and $MAX()$ functions.

```
findTopsort(gate g) {
  for each gate o in g.outputs {
    if o has not been visited {
      findTopsort(o);
      push g on to topsortStack;
      mark g as visited;
    }
  }
}
```

Figure 2.5: Finding a Topological Sort

$$AT^{g}_{min} = ADD\left(\tau^{g}_{Pmin}, MIN\left(AT^{a}_{min}, AT^{b}_{min}\right)\right) \tag{2.5}$$

$$AT^{g}_{Max} = ADD\left(\tau^{g}_{PMax}, MAX\left(AT^{a}_{Max}, AT^{b}_{Max}\right)\right) \tag{2.6}$$

2.4 Zero Clock Skew Timing Limitations

Since zero clock skew (ZCS) circuits presume that the synchronizing clock signal arrives in phase at all of the registers, the minimum period T^{zcs}_{min} at which the clock must be operated depends on the slowest local data path in the circuit. The internal delays of the registers, the setup time δ_S of R_f and the clock-to-output time $d^{R_i}_{CQM}$ of R_f, are also considered. This is defined mathematically in Equation 2.7, both in terms of the worst case “ss” corner local data path delays among all register pairs as well as simply the maximum arrival time among all registers.

$$T^{zcs}_{min} = \underset{\forall(R_i,R_f)}{max}\left[d^{R_i}_{CQM} + d^{if}_{PM} + \delta^{R_f}_{S}\right] = \underset{\forall R_f}{max}\left[AT^{Rf}_{Max} + \delta^{R_f}_{S}\right] \tag{2.7}$$

Once this limitation is calculated, certain paths in the circuit may need modification in order to reduce the worst case clock period to meet the desired clock frequency of the product. If such modifications are not possible due to other constraints, the specifications of the chip will need to be relaxed and profits will be lost.

Chapter 3. Statistical Static Timing Analysis

Statistical static timing analysis has recently emerged in order to mitigate the pessimism of traditional deterministic static timing analysis that has become severely problematic in deep submicron technologies. Statistical models are used to represent process variations and delays in order to improve the accuracy of timing analysis while still maintaining its efficiency and speed relative to ciruit simulation.

3.1 Statistical Delay Models & Sensitivities

Instead of the deterministic worst case corner models used by STA described in Chapter 2, statistical static timing analysis (SSTA) methods model delays and arrival times as random variables with mean μ and standard deviation σ. Statistical process information from fabrication facilities includes information on the distribution of parameters such as L_e and V_T that can now be represented as random variables for the purposes of timing analysis. Revisiting Figure 3.1 from Chapter 2, it is seen that the worst case corner values for process parameters actually corresponded to points along these statistical distributions.

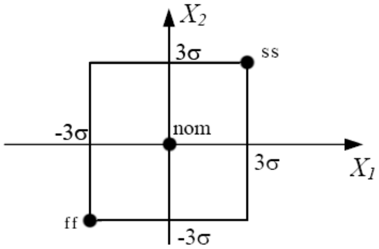

Figure 3.1: Statistical Analysis

Instead of evaluating these distributions at a given point (typically $\mu \pm 3\sigma$) and using those deterministic values throughout the static timing analysis flow, statistical static timing analysis propagates the statistical distributions and performs mathematical operations on the distributions themselves in order to improve accuracy.

3.1.1 Gaussian Random Variables

For this study in particular, variations, delays and the results of all statistical operations are modeled as Gaussian random variables. Although methods to efficiently handle non-Gaussian distributions are currently being investigated [18, 19], Gaussian modeling has been the most popular in statistical static timing analysis research. A sample definition of a Gaussian random variable is shown in Equation 3.1. The probability density function (PDF) of such a Gaussian random variable X is given by Equation 3.2.

$$X \sim N(\mu, \sigma^2) \tag{3.1}$$

$$f(x; \sigma, \mu) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right] \tag{3.2}$$

In order capture variation sensitivity, these random variables can be put into a *canonical form*, as in Equation 3.3, that captures their nominal value as well as the effects of different types of process variation [5].

$$A = a_o + \sum_{i=1}^{n} a_i \Delta X_i + a_{n+1}\Delta R_a \equiv N\left(a_o, \sum_{i=1}^{n+1} a_i^2\right) \tag{3.3}$$

Here, a_o is the mean value of a random variable A. ΔX_{i-n} and ΔR_a correspond to systematic global and random local sources of variation, respectively, and a_{i-n} and a_{n+1} are the *sensitivities* of random variable A to these sources of variation. The sensitivity s_A^X of a random variable A to variation in a random variable X is defined as in Equation 3.4 [20].

$$s_A^X = \frac{\partial A}{\partial X} \tag{3.4}$$

3.2 Statistical MIN, MAX, & ADD Operations

SSTA methods use canonical forms of logic gate propagation delays as well as arrival times in order to perform the $MAX()$, $MIN()$, and $ADD()$ functions. Unlike with deterministic modeling, these operations are no longer straightforward. Rather than simply comparing real numbers for a $MAX()$ operation, random variables must be compared. When examining two random variables A and B, the *tightness probability* of A, T_A, is the probability that the random variable A is greater than, or dominates, B as defined in Equation 3.5 [21, 5]. The probability that B dominates A, then, is simply $(1 - T_A)$.

$$T_A = \Phi\left(\frac{a_o - b_o}{\theta}\right), T_B = (1 - T_A) \tag{3.5}$$

This tightness probability is defined in terms of the cumulative distribution function (CDF) $\Phi(y)$ (Equation 3.7), using the mean values of A and B (a_o and b_o), the standard normal PDF $\phi(x)$ (Equation 3.6), the correlation coefficient ρ (Equation 3.9), and the expression for θ as defined by [22] (Equation 3.8).

$$\phi(x) \equiv \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \tag{3.6}$$

$$\Phi(y) \equiv \int_{-\infty}^{y} \phi(x)dx \tag{3.7}$$

$$\theta \equiv (\sigma_A^2 + \sigma_B^2 - 2\rho\sigma_A\sigma_B)^{1/2} \tag{3.8}$$

$$\rho = \frac{\sum_{i=1}^{n} a_i b_i}{\sigma_A \sigma_B} \tag{3.9}$$

The addition of two random variables in canonical form results in a new random variable as defined in Equation 3.10.

$$\begin{aligned} add(A,B) &= (a_o+b_o)+\textstyle\sum_{i=1}^{n}(a_i+b_i)\Delta X_i+\left(\sqrt{a_{n+1}^2+b_{n+1}^2}\right)\Delta R_a \\ &= c_o+\textstyle\sum_{i=1}^{n}c_i\Delta X_i+c_{n+1}\Delta R_a \end{aligned} \tag{3.10}$$

The subtraction operator is defined similarly by 3.11.

$$\begin{aligned} sub(A,B) &= (a_o-b_o)+\textstyle\sum_{i=1}^{n}(a_i+b_i)\Delta X_i+\left(\sqrt{a_{n+1}^2+b_{n+1}^2}\right)\Delta R_a \\ &= d_o+\textstyle\sum_{i=1}^{n}d_i\Delta X_i+d_{n+1}\Delta R_a \end{aligned} \tag{3.11}$$

For the $MAX()$ operation, the resulting random variable has a mean and variance as shown in Equations 3.12 and 3.13, respectively.

$$\begin{aligned} \mu_{max(a,b)} &= E[max(A,B)] \\ &= a_oT_A+b_oT_B+\theta\phi\left[\tfrac{a_o-b_o}{\theta}\right] \end{aligned} \tag{3.12}$$

$$\begin{aligned} \sigma_{max(a,b)}^2 &= var[max(A,B)] \\ &= (\sigma_A^2+a_o^2)T_A+(\sigma_B^2+b_o^2)T_B+(a_o+b_o)\theta\phi\left(\tfrac{a_o-b_o}{\theta}\right)-\mu_{max(a,b)}^2 \end{aligned} \tag{3.13}$$

Where $G = MAX(A,B)$, G can be put back in to canonical form as shown in Equations 3.14, 3.15, and 3.16.

$$g_o = \mu_{max(a,b)} = E[max(A,B)] \tag{3.14}$$

$$g_i = T_Aa_i + T_Bb_i \tag{3.15}$$

$$g_{n+1} = \sqrt{\sigma_{max(a,b)}^2 - \sum_{i=1}^{n} g_i^2} \tag{3.16}$$

One very important difference between the statistical version of the $MAX()$ function and the deterministic version is that with SSTA, the result of the $MAX()$ operation is a *new* random variable with its own mean and variance, rather than being identical to one of the operands. Furthermore, when taking the $MAX()$ of more than two random variables by using nested $MAX()$ calls, it has been shown that these calls need to be in order of the operands with increasing mean values [21].

The statistical $MIN()$ operation is very similar to the $MAX()$ operation. The mean and variance of the resulting random variable is defined as in Equations 3.17 and 3.18, respectively. Nested calls to the $MIN()$ operation must also be made in order of the operands with increasing mean values.

$$\begin{aligned} \mu_{min(a,b)} &= E[min(A,B)] \\ &= a_o T_B + b_o T_A - \theta\phi\left[\tfrac{a_o - b_o}{\theta}\right] \end{aligned} \tag{3.17}$$

$$\begin{aligned} \sigma^2_{min(a,b)} &= var[min(A,B)] \\ &= (\sigma_A^2 + a_o^2)T_B + (\sigma_B^2 + b_o^2)T_A - (a_o + b_o)\theta\phi\left(\tfrac{a_o - b_o}{\theta}\right) - \mu^2_{min(a,b)} \end{aligned} \tag{3.18}$$

Where $H = MIN(A,B)$, H can be put back in to canonical form as shown in Equations 3.19, 3.20, and 3.21.

$$h_o = \mu_{min(a,b)} = E[min(A,B)] \tag{3.19}$$

$$h_i = T_B a_i + T_A b_i \tag{3.20}$$

$$h_{n+1} = \sqrt{\sigma^2_{min(a,b)} - \sum_{i=1}^{n} h_i^2} \tag{3.21}$$

3.3 Statistical Zero Clock Skew Timing Limitations

Remember from Section 2.4 that the clock frequency of zero clock skew circuits depends on the maximum *arrival time* to all registers within a circuit. In statistical analysis, this limitation still holds true; however, that maximum is now represented as the probability density function of a random variable rather than a deterministic number.

$$T_{min}^{zcs} = \underset{\forall R_f}{MAX}\left(AT_{Max}^{R_f}\right) \sim X(\mu_{AT}, \sigma_{AT}^2) \tag{3.22}$$

As Equation 3.22 implies, designers can see the probability of a circuit being able to function at a specific clock frequency, i.e. the *timing yield* at each frequency. For microprocessor manufacturers this means a much more accurate method of realizing how a design will be *speed binned* prior to manufacturing [6]. For ASIC manufacturers, it means a clear picture of the yield vs. design effort tradeoff and quicker timing sign-off.

3.3.1 Pessimism of STA

The main reason why deterministic static timing analysis is so pessimistic is that it presumes worst case (μ+3σ) values for all variation sources and uses those values at each gate to determine the overall performance impact on the circuit. Although it is known that the probability of a particular random variable falling within a (μ+/-3σ) window is 99.73% as in equation 3.23, the probability of the sum (i.e. after performing the $ADD()$ operation for an entire circuit) of several random variables falling within this window grows as in Equation 3.24 with n random variables [23].

$$\Phi(3) - \Phi(-3) = 99.73\% \tag{3.23}$$

$$\Phi(3\sqrt{n}) - \Phi(-3\sqrt{n}) \tag{3.24}$$

The result of the pessimism in deterministic STA, and the improvement in T_{min} seen by SSTA is portrayed graphically in Figure 3.2 for an example circuit s13207 from the ISCAS'89 suite of benchmark circuits.

In Figure 3.2, the nominal and worst cast corner minimum clock period T_{min} are computed with STA, while the 99.73% yield clock period is computed with SSTA. Deterministic STA analysis would suggest a performance limit of ~1740ps that could be met by 100% of manufactured chips. SSTA analysis, however, reveals the exact distribution of performance and indicates that 99.73% of chips manufactured would be able to operate with a clock period of ~1570ps.

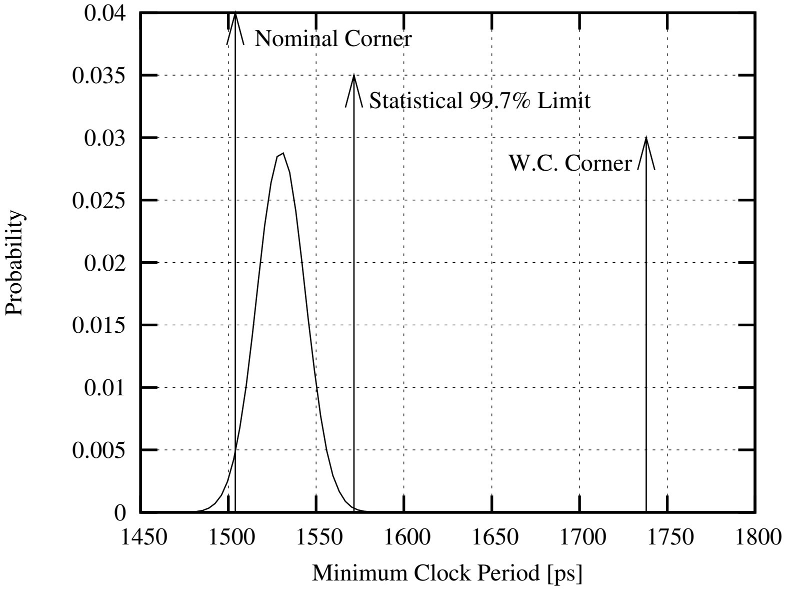

Figure 3.2: Probability Density Function of T_{min} for s13207 Circuit Compared to Corner Analysis

Unlike with the deterministic minimum clock period calculated with STA, the minimum clock period probability distribution provided by SSTA allows designers to assess the profitability and functionality of a chip prior to manufacturing, to accurately determine which paths in a chip might need modification, as well as exactly what impact any such modifications will have on the timing yield of a design.

Chapter 4. Nonzero Clock Skew Circuits

In previous chapters, circuits were assumed to have zero clock skew between all registers in a circuit; that is, the the synchronizing clock signal arrives in phase with respect to all register pairs as seen in Figure 4.1. This assumption implies that each local data path in a circuit has an equal amount of time in which to propagate its signal between registers, regardless of whether or not each path *needs* that much time. As was defined in Chapter 2, the minimum clock period T_{min} of a circuit is set by the worst (slowest) local data path. The remainder of the local data paths may actually require less time, and therefore have some *slack* as defined by Equation 4.1 for a local data path between registers R_i and R_f.

$$slack_{p^{if}} = T_{min} - \left[d_{CQM}^{R_i} + d_{PM}^{if} + \delta_S^{R_f} \right] \tag{4.1}$$

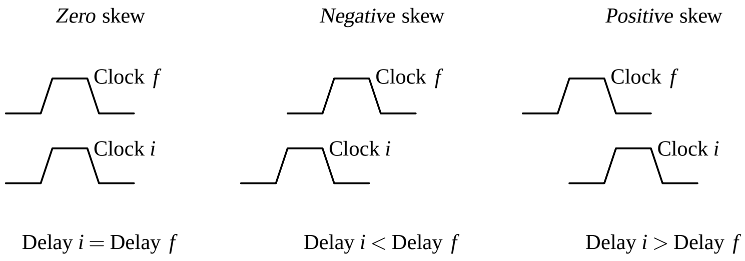

Figure 4.1: Different Clock Skews

In *nonzero clock skew* systems, the clock skews between register pairs are manipulated in order to make use of the slack on faster paths and to thereby provide additional time to the slower paths as seen in Figure 4.1. Such systematic assignment of positive or negative skew to local data paths effectively *decreases* the T_{min} of the overall circuit [17]. The registers and local data paths of a circuit form a large, interconnected graph. Any positive or negative skew assignment, therefore, affects the timing constraints of other registers and paths. The set of clock delays to each register is called the *clock skew schedule*. The process of finding this clock skew schedule in order to minimize T_{min} of the circuit is called *clock skew scheduling* [11].

4.1 Clock Frequency Limitations

An important distinction must be drawn between T_{min}^{zcs}, the minimum clock period for zero clock skew circuits as defined in Equation 2.7, and T_{min}^{NZCS}, the minimum clock period for nonzero clock skew circuits. Whereas T_{min}^{zcs} depends solely on the slowest local data path of a circuit, the maximum frequency gain achievable with clock skew schedule (T_{min}^{NZCS}) depends on three new and very unique limitations based in part on the topology of registers within the circuit as presented in Reference [17]. These three limitations, listed below, are discussed in detail in Sections 4.1.1, 4.1.2, and 4.5.

I. $T_{min}^{NZCS,I}$: Uncertainty of local data path propagation delays

II. $T_{min}^{NZCS,II}$: Data path cycle propagation delays

III. $T_{min}^{NZCS,III}$: Difference in propagation delays among reconvergent paths

The resulting T_{min}^{NZCS} is set by the worst of these three limits, as in Equation 4.2.

$$T_{min}^{NZCS} = max\left[T_{min}^{NZCS,I}, T_{min}^{NZCS,II}, T_{min}^{NZCS,III}\right] \tag{4.2}$$

The first of these limits occurs on every single local data path, while the second and third limits only occur for circuits where the circuit topology includes cycles and reconvergent paths, respectively [17].

4.1.1 Uncertainty of local data path propagation delays

Nonzero clock skew circuits depend not only on the slowest local path delay, but also on the difference between the maximum and minimum delays on a local data path between any register pair R_i and R_f, as seen in Figure 4.2. As defined by [17], the clock period cannot be minimized by clock skew scheduling any further than seen in Equation 4.3.

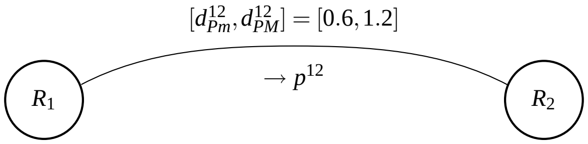

Figure 4.2: Local Data Path

$$T_{min}^{NZCS,I} = \max_{\forall R_i \rightsquigarrow R_f} \left[d_{PM}^{if} + \delta_S - (d_{Pm}^{if} + \delta_H) \right] \tag{4.3}$$

For instance in Figure 4.2, assuming negligible register delays ($\delta_S = \delta_H = 0$), the minimum clock period limit imposed by this local data path would be $T_{min}^{NZCS,I} = (1.2 - 0.6) = .6$.

4.1.2 Data path cycle propagation delays

In nonzero clock skew circuits, however, the timing relationships between registers in a data path cycle, as seen in Figure 4.3, limit the clock period achievable by clock skew scheduling.

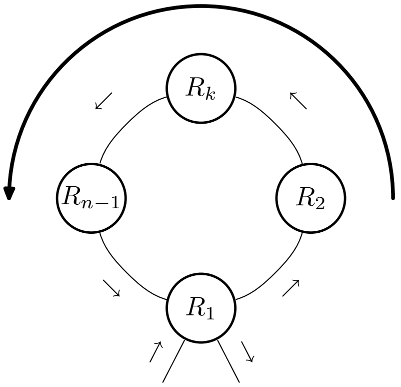

Figure 4.3: Data Path Cycle with n Registers

The limitation imposed on the clock period by data path cycles is dependent upon the maximum local data path delays between registers on the cycle as well as the number of registers on the cycle, as defined in Equation 4.4.

$$T_{min}^{NZCS,II} = \max_{\forall cycles} \left[\frac{\sum_{\forall R_i \rightsquigarrow R_f on cycle} (d_{CQM}^{R_i} + d_{PM}^{if} + \delta_S)}{n} \right] \tag{4.4}$$

For instance in Figure 4.3, assume negligible register delays ($\delta_S = d_{CQM} = 0$), and $\left(d_{PM}^{12} = 3, d_{PM}^{23} = 5, d_{PM}^{34} = 2, d_{PM}^{41} = 10\right)$. The limit that this cycle imposes on the minimum clock period would be $T_{min}^{NZCS,II} = \left(\dfrac{3+5+2+10}{4}\right) = 5$.

In order to account for this limitation, the reduced circuit graph must be searched for all existing cycles. This search can be performed using a depth first search (DFS) including a numbering scheme for labeling nodes (registers) as they are first visited and completed, as in the algorithm defined by Figure 4.4. Any "backedge" seen during this search implies that a cycle exists. Upon detection, a vector listing the registers on the cycle is pushed on to a list for future analysis.

```
for each register r in circuit {
  findCycles(r,0);
}

findCycles(register r, int val) {
  if r has not been visited {
    push r on to pathStack;
    r.visitValue1 = val;

    for each register o in r.connectedRegisters {
      if(o.visitValue1 is NULL) {
        if(r.visitValue2 is NULL) {
          r.visitValue2 = findCycles(o, val1+1); } else {
          r.visitValue2 = findCycles(o, g.visitValue2+1); }
      }

      if(o.visitValue1 is not NULL AND o.visitValue2 is NULL) {
        # detected cycle
        push o on to cycleStack;
        do ( push ( pop pathStack ) on to cycleStack )
          while top of cycleStack is not o;
      }
    }
    return ( r.visitValue2 + 1 )
  }
}
```

Figure 4.4: Detecting Cycles in a Directed Graph

4.1.3 Difference in propagation delays among reconvergent paths

As discovered fairly recently [12], the topology of reconvergent data paths, as depicted in Figure 4.5, imposes a limitation similar to that of data path cycles. The minimum clock period possible with clock skew scheduling depends on the differences in data propagation times between parallel reconvergent paths as well their relative lengths as defined by Equation 4.5.

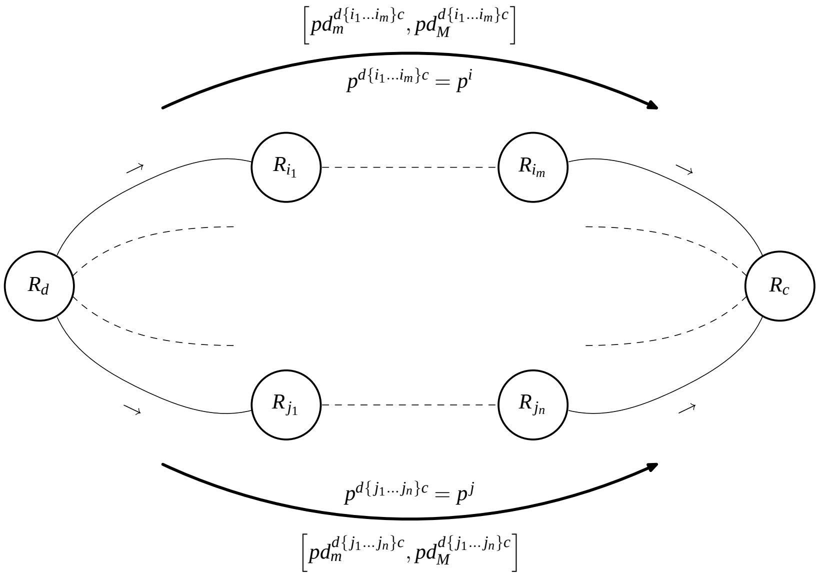
$\left[pd_m^{d\{i_1 \dots i_m\}c}, pd_M^{d\{i_1 \dots i_m\}c}\right]$
$p^{d\{i_1 \dots i_m\}c} = p^i$
R_{i_1}
R_{i_m}
R_d
R_c
R_{j_1}
R_{j_n}
$p^{d\{j_1 \dots j_n\}c} = p^j$
$\left[pd_m^{d\{j_1 \dots j_n\}c}, pd_M^{d\{j_1 \dots j_n\}c}\right]$

Figure 4.5: Reconvergent Register-to-Register Paths

$$T_{min}^{NZCS,III} = \underset{\forall(R_d,R_c)}{max} \left[\underset{\forall(p^i,p^j)}{max} \left(\frac{pd_M^{p^i} - pd_m^{p^j} + \delta_S + \delta_H}{|m-n+1|} \right) \right] \tag{4.5}$$

This limit depends on minimum and maximum reconvergent path propagation delays, $[pd_m, pd_M]$ as defined by Equations 4.6 and 4.7, as well as the number of registers along each path. Furthermore, as Equation 4.5 indicates, all possible pairs of paths between a divergent and reconvergent register must be examined.

$$pd_m^{p^i} = \sum_{i=1}^{n-2} d_{PM}^{i,i+1} + d_{Pm}^{n-1,n} \tag{4.6}$$

$$pd_{M}^{p^i} = \sum_{i=1}^{n-1} d_{PM}^{i,i+1} \tag{4.7}$$

The thorough detection of reconvergent data paths is somewhat challenging in cyclic sequential circuits. Algorithms for detection have been proposed [24], though they are unable to detect all possible reconvergent paths in the presence of data path cycles. This work defines an algorithm for this detection by Figure 4.6 that is relatively inefficient since its time complexity grows with n registers as $O(n^2)$. This is suitable for research performed in this thesis on relatively small academic benchmark circuits; however, a more robust algorithm should be investigated in order to apply these concepts to larger industrial designs.

```
for each register r in circuit {
  findReconvergence(r,0);
}

findReconvergence(register r, int val) {
  push r on to pathStack;
  r.visitValue1 = val;

  for each register o in r.connectedRegisters {
    if(o.visitValue1 is NULL) {
      if(r.visitValue2 is NULL) {
        r.visitValue2 = findReconvergence(o, val1+1); } else {
        r.visitValue2 = findReconvergence(o, g.visitValue2+1); }
    }

    if(o.visitValue1 is not NULL AND o.visitValue2 is not NULL) {
      # detected reconvergent register
      reconvRegister = o;
      do ( push ( pop pathStack ) on to reconvPathStack )
        while top of reconvPathStack is not o;
      diverRegister = reconvPathStack.top;

      DFS to find all paths from diverRegister to reconvRegister;
    }
  }
  return ( r.visitValue2 + 1 )
}
```

Figure 4.6: Detecting Reconvergent Paths

4.2 Statistical Representation & Analysis of NZCS Limits

In order to perform statistical timing analysis of nonzero clock skew circuits, the three new timing limits must be represented in random variable form. For each of $T_{min}^{NZCS,I}$, $T_{min}^{NZCS,II}$, and $T_{min}^{NZCS,III}$, an additional subscript of "$ssta$" denotes that the limit is a Gaussian random variable rather than a deterministic number. Similarly, capital letters used for local path delays and reconvergent branch delays also indicate random variables (e.g. D_{PM} instead of d_{pm}).

The statistical limit imposed by uncertainty in all local data paths can be calculated relatively easily using the operations that were defined in Chapter 3, as shown in Equation 4.8.

$$T_{min,ssta}^{NZCS,I} = \underset{\forall R_i \rightsquigarrow R_f}{MAX} \left[SUB \left(D_{PM}^{if}, D_{Pm}^{if} \right) \right] \tag{4.8}$$

For simplicity, negligible internal register delays (δ_S=δ_H=d_{CQ}=0) are assumed in this study, although these delays could incorporated in either a deterministic or statistical manner. It is important to note that the Gaussian result $SUB\left(D_{PM}^{if}, D_{Pm}^{if}\right)$ is first calculated for each local data path in the system, and then these results are compared by nested calls to the $MAX()$ function in order of increasing mean value. The minimum and maximum local path delays $[D_{Pm}, D_{PM}]$ are calculated in the same manner as performed in Chapter 3.

The statistical limit imposed by local data path cycles can be calculated as shown by Equation 4.9.

$$T_{min,ssta}^{NZCS,II} = \underset{\forall cycles}{MAX} \left[\frac{\underset{\forall R_i \rightsquigarrow R_f oncycle}{ADD} \left(D_{PM}^{if}, D_{PM}^{cycle} \right)}{n} \right] \tag{4.9}$$

The summation of all maximum local data path delays must first be calculated for each cycle and scaled by n. The results for each cycle are then compared using nested calls to the $MAX()$ function in the proper order. The scaling of a Gaussian random variable by an integer is performed as shown in Equation 4.10 and in Equation 4.11 for a random variable in canonical form [25].

$$\frac{N\left(\mu, \sigma^2\right)}{c} = N\left(\frac{\mu}{c}, \left(\frac{\sigma}{c}\right)^2\right) \tag{4.10}$$

$$\frac{A}{c} = \frac{a_o}{c} + \sum_{i=1}^{n} \frac{a_i}{c} \Delta X_i + \frac{a_{n+1}}{c} \Delta R_a \tag{4.11}$$

The statistical limit imposed by reconvergent local data paths can be calculated as shown by Equation 4.12.

$$T_{min,ssta}^{NZCS,III} = \underset{\forall (R_d, R_c)}{MAX} \left[\underset{\forall (p^i, p^j)}{MAX} \left(\frac{SUB\left(PD_M^{p^i}, PD_m^{p^j}\right)}{|m - n + 1|} \right) \right] \tag{4.12}$$

All possible pairs of reconvergent branches $\left[p^i, p^j\right]$ between a divergent and reconvergent register pair $[R_d, R_c]$ must first be compared using the $MAX()$ function in the proper order. Finally, the $MAX()$ of this result for each divergent and reconvergent register pair $[R_d, R_c]$ in the circuit is calculated.

By applying SSTA to the different timing limitations of nonzero clock skew circuits, an accurate picture is developed as to how aggressively these circuits can be clocked without noticeably affecting the timing yield of a design. This more accurate modeling, along with clock skew scheduling, allows the highest level of performance to be achieved.

Chapter 5. Experimental Setup

In order to assess the efficacy of SSTA on NZCS circuits, experiments are performed on academically available benchmark circuits ISCAS'85 (combinational) and ISCAS'89 (sequential). Physical implementations of these circuits are generated by technology mapping to a 90nm cell library. Spice modeling and simulation are performed to characterize delays and process sensitivities.

5.1 Predictive Technology Models

Predictive Technology Models (PTM) [26, 27] are used in order to model MOSFETs in future nanoscale CMOS technologies, where fabrication data is not yet available. Arizona State University researchers have created and maintained a new generation of these models, originally from UC Berkeley, which have proven to be reasonably accurate and thus have become popular in circuit design and automation research. The PTM team has also created a tool that generates corner-case models for channel length (L_e) and threshold voltage variation (V_T). In this work, 90nm models are used where 3σ variation corresponds to 10% deviation from nominal values for both L_e and V_T, in accordance with the International Technology Roadmap for Semiconductors (ITRS). Below 90nm, this variation worsens and 3σ variation is over 15% [2].

5.2 Cell Library & Technology Mapping

MVSIS, another UC Berkeley software package modeled after the original SIS, is comprised of a number of different circuit analysis tools for circuit synthesis, combinational optimization, verfication, and technology mapping. Included in the package are a number of cell libraries in *.genlib* format, the most popular of which are the MCNC and Lib2 libraries. The *.genlib* format defines a number of logic gates by their boolean function and by technology independent delay and area numbers. This ensures that when MVSIS technology maps a particular circuit on to a library, the

Table 5.1: Lib2 Cell Library Gates

inv	nor2	aoi33	oai22
xor	nor3	aoi211	oai32
xnor	aoi21	aoi221	oai33
nand2	aoi31	aoi222	oai211
nand3	aoi22	oai21	oai221
nand4	aoi32	oai31	oai222

result is a netlist with realistic gate choice and gate output loading. The Lib2 library is chosen for this work because of its rich collection of gates, as seen in Table 5.1.

The ISCAS benchmark circuit netlists used in this study are provided in the *.BENCH* format that defines the primary inputs, primary outputs, intermediate nodes and the logic functions of a circuit. These logic functions are only in terms of AND, OR, NOT, NAND, and NOR, and provide no information as to the circuit implementation, nor do they limit the number of inputs or output loading to a physically feasible configuration. As in [28], MVSIS is used to map these boolean functions to the Lib2 library and provides a new netlist in the Berkeley Logic Interchange Format (*.BLIF*). The final *.BLIF* format provides a final physical netlist of the ISCAS circuit using the gates in the selected library. A typical run of this procedure is shown in Figure 5.1.

```
Mvsis> Read_library Lib2.genlib
Mvsis> Read_bench c432.bench
Mvsis> Map -s
Mvsis> Write_gate .n c432.blif
```

Figure 5.1: MVSIS Technology Mapping

5.3 Physical Cell Definition, Delay & Sensitivity Characterization

In order to characterize the actual nominal delays, worst case delays, and variation sensitivities of the logic cells in the Lib2 library for 90nm technology, Spice simulations is performed. For these simulations, physical Spice *.subckt* definitions are created for minimum size devices with a beta ratio designed for near equal rise and fall times. Using the nominal and corner-case predictive models, four versions of each cell are produced to correspond to each corner of V_T and L_e variation:

- nominal V_T, nominal L_e
- slow V_T, nominal L_e
- nominal V_T, slow L_e
- slow V_T, slow L_e

For brevity, these corners are abbreviated as “NN”, “SN”, “NS”, and “SS”, respectively. The different corners are used to calculate each cell’s sensitivity to both types of variation, as well their nominal and worst case propagation delays. Cell sensitivites are calculated as in Equations 5.1 and 5.2. Since this is block based timing analysis, particular slews are ignored and the worst value for sensitivity is used.

$$s_{\tau_P}^{V_T} = \frac{\partial \tau_P}{\partial V_T} = \underset{\forall slews}{max} \left[\frac{\tau_{P,SN} - \tau_{P,NN}}{\Delta V_T} \right] \tag{5.1}$$

$$s_{\tau_P}^{L_e} = \frac{\partial \tau_P}{\partial L_e} = \underset{\forall slews}{max} \left[\frac{\tau_{P,NS} - \tau_{P,NN}}{\Delta L_e} \right] \tag{5.2}$$

As in [16], 2-D tables are constructed to record propagation delays and variation sensitivities for each cell at different output capacitances (C_L) and input slopes (tt_{in}). Once the circuit is parsed and the loading capacitances are known, bilinear interpolation is used to calculate precise values for nominal and worst case propagation delays as well as delay sensitivities. These delays and sensitivities are the values used in the canonical delay form dicussed in Chapter 3 for all timing analysis calculations.

For simplicity of library creation and characterization, internal register delays (δ_H, δ_S, and d_{CQ}) are assumed to be negligible in this study, although they could easily be incorporated in to future work. Futhermore, the variations in the clock distribution networks themselves are also ignored, although future accuracy improvements should include these variations as well. This is justified since there is much greater control over the delay to different points in the clock distribution network, both during design and after fabrication, than there is in local data paths. A number of techniques to exercise such control include regional active deskew (RAD) feedback, second level clock buffers (SLCBs), and post-Silicon tuning (PST) as discussed in [29].

5.4 System Configuration

All programming and analysis is conducted on a 3GHz Intel Pentium 4 machine with 1GB of RAM running Fedora Core 6, linux 2.6.18, gcc 4.1.1 and Perl 5.8.8. Simulations are performed with Ngspice [30], a derivative of Berkeley Spice3f5 [31], compiled with BSIM4.6.0 Spice model support [32]. As mentioned in Section 5.1, the 90nm Predictive Technology Model V1.0 model cards are used for all simulations [26].

Chapter 6. Experimental Results & Discussion

A number of experiments are first carried out to examine the 90nm cell library created for this work, and to verify the implementation of the mathematics relevant to statistical static timing analysis and nonzero clock skew circuits. In Section 6.1, the sensitivities of the 90nm logic gates to process variations are calculated to confirm that they follow the linear assumption made in Chapter 3. The nonzero clock skew circuit timing limits are applied in Section 6.2 in order to compare the minimum clock period calculations in this work with other published results. Similarly, statistical static timing analysis of the benchmark circuits assuming zero clock skew is conducted in Section 6.3 to verify the proper handling of random variables in finding the statistical (μ+3σ) minimum clock period T_{min}. In Section 6.4, the improvements in the minimum clock period seen by zero clock skew circuits are compared with those seen by nonzero clock skew circuits. Clock period improvements with both clock skew scheduling and statistical static timing analysis are shown in Section 6.5. A discussion comparing nonzero clock skew timing limits in the determinisitc and statistical domains is presented in Section 6.6 and, lastly, CPU run times are compared for zero clock skew and nonzero clock skew statistical static timing analysis in Section 6.7.

6.1 90nm Cell Sensitivity Analysis

As discussed in Chapter 5, the sensitivities of logic gate propagation delay to process variations are linear with respect to the nominal delay of the gate, i.e. $s_d \sim cf(C_L, tt_{in})$. Such linearity enables the interpolation of sensitivities for fast analysis. This work characterizes the propagation delays and process variation sensitivities for the 90nm cell library used with Spice simulation, and confirms this linear relationship experimentally as shown Figures 6.1 and 6.2.

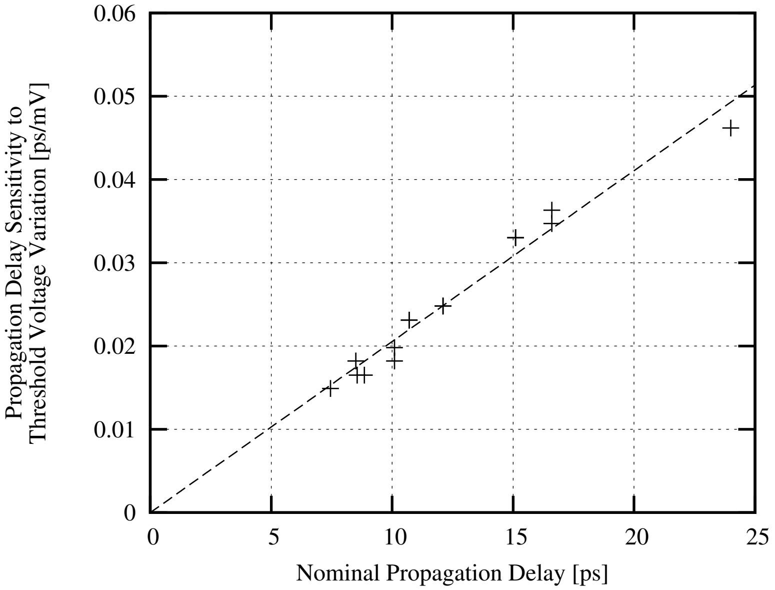

Figure 6.1: 90nm Inverter Propagation Delay V_T Sensitivity vs. Nominal Delay

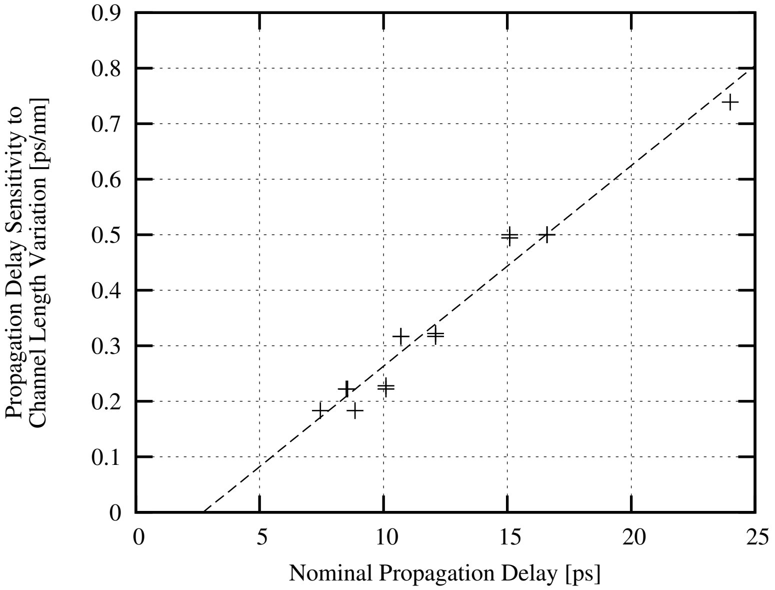

Figure 6.2: 90nm Inverter Propagation Delay L_e Sensitivity vs. Nominal Delay

In Figures 6.1 and 6.2, the data points correspond to an inverter characterized at different sizes, output loads (C_L), and input slopes (tt_{in}).

6.2 CSS Improvement with Deterministic Models

In order to confirm the experimental setup, deterministic corner-based timing analysis is first carried out at both the nominal corner and worst case "ss" corner, assuming zero clock skew. The deterministic "ss" corner results for the minimum clock period T_{min} are the pessimistic baseline results that the techniques outlined in this thesis aim to improve.

The limitations of clock skew scheduling are assessed in order to determine CSS improvements for this implementation, and to validate that the calculation of these limits was carried out properly. As shown in Table 6.1, an average improvement of 29.67% is achieved, which agrees with [11]. The number of gates, number of registers, minimum deterministic zero skew clock period $T_{min,ss}^{zcs}$, minimum deterministic nonzero skew clock period $T_{min,ss}^{NZCS}$, the percentage improvement in T_{min} with clock skew scheduling as defined by Equation 6.1, and the limiting factor are shown in Table 6.1.

$$\% \text{ Improvement}_{\text{CSS}} = \frac{T_{min,ss}^{zcs} - T_{min,ss}^{NZCS}}{T_{min,ss}^{zcs}} \times 100\% \tag{6.1}$$

6.3 ZCS Circuit SSTA Improvement

Statistical static timing analysis is performed on each benchmark circuit assuming zero clock skew operation in order to see a baseline performance increase possible with SSTA alone, and to validate the implementation of the underlying SSTA computations. Results and improvements can be seen in Table 6.2 which agree with those reported in [4]. In particular, the number of inputs, number of gates, nominal corner minimum clock period $T_{min,nom}^{zcs}$, worst case corner minimum clock period $T_{min,ss}^{zcs}$, statistical (μ+3σ) minimum clock period $T_{min,ssta}^{zcs}$, and the improvement between $T_{min,ss}^{zcs}$ and $T_{min,ssta}^{zcs}$ as defined by Equation 6.2 are shown in Table 6.2.

$$\% \text{ Improvement}_{\text{SSTA}}^{\text{zcs}} = \frac{T_{min,ss}^{zcs} - T_{min,ssta}^{zcs}}{T_{min,ss}^{zcs}} \times 100\% \tag{6.2}$$

6.4 SSTA Improvement for ZCS Limit vs. NCZS Limit

The objective of this thesis is to determine the relative importance of statistical static timing analysis (SSTA) for nonzero clock skew circuits. Since clock skew scheduling introduces new, unique,

Table 6.1: Summary of Corner-Based Clock Skew Scheduling Improvement

Circuit	# Gates	# Registers	$T^{zcs}_{min,ss}$[ps]	$T^{NZCS}_{min,ss}$[ps]	NZCS Limit	% Improvement
c17	8	0	71.033	36.70	1	48.33%
c432	152	0	940.45	478.23	1	49.15%
c499	405	0	868.33	535.87	1	38.29%
c880	241	0	675.30	606.73	1	10.15%
c1355	412	0	830.98	546.67	1	34.21%
c1908	430	0	1050.20	739.00	1	29.63%
c2670	701	0	772.68	286.32	1	62.95%
c3540	844	0	1401.30	742.83	1	46.99%
c5315	1204	0	1073.90	280.47	1	73.89%
c6288	3090	0	3241.60	2151.60	1	33.62%
c7552	1624	0	2394.40	2168.50	1	9.43%
s27	12	3	103.31	88.618	2	14.22%
s208	71	8	283.43	137.26	3	51.57%
s298	85	14	442.61	389.86	1	11.92%
s349	109	15	548.09	273.77	1	50.05%
s382	113	21	397.93	261.93	2	34.18%
s386	106	6	374.66	309.10	1	17.50%
s400	118	21	363.82	236.60	2	34.97%
s420	143	16	503.97	141.50	1	71.92%
s444	142	21	471.30	292.69	1	37.90%
s510	163	6	443.56	350.58	1	20.96%
s526	156	21	436.62	383.88	1	12.08%
s641	140	19	788.12	635.62	1	19.35%
s713	146	19	818.67	666.17	1	18.63%
s820	214	5	670.07	593.32	1	11.45%
s832	219	5	656.60	579.85	1	11.69%
s838	287	32	934.57	198.37	1	78.77%
s953	316	29	453.71	389.91	2	14.06%
s1196	358	18	694.15	574.09	1	17.30%
s1238	387	18	788.32	405.84	1	48.52%
s1423	432	74	2609.30	2233.10	1	14.42%
s1488	389	6	1053.00	979.82	1	6.95%
s1494	395	6	1095.10	1021.90	1	6.69%
s13207	2900	511	1738.00	1508.80	3	13.19%
s35932	13278	1728	17958.00	17921.00	1	0.20%
s38417	8917	1535	6207.30	5405.10	1	12.92%
					Average	29.67%

and topologically dependent timing limitations, the impact of statistical modeling and analysis is projected to be significant. Futhermore, it is desirable to determine which circuit topologies benefit the most from the SSTA, if any.

The results for relative improvement (Equation 6.4) in the minimum clock period for zero clock skew and nonzero clock skew (Equation 6.3) circuits are shown in Table 6.3. On average, this

Table 6.2: Summary of Zero Clock Skew Circuit SSTA Improvement

Circuit	# In	# Out	# Gates	$T^{zcs}_{min,nom}$[ps]	$T^{zcs}_{min,ss}$[ps]	$T^{zcs}_{min,ssta}$[ps]	% Impr.
c17	5	2	8	61.43	71.03	62.32	12.26%
c432	36	7	152	811.55	940.45	835.40	11.17%
c499	41	32	405	748.57	868.33	775.70	10.67%
c880	60	26	241	581.38	675.30	595.23	11.86%
c1355	41	32	412	717.40	830.98	743.00	10.59%
c1908	33	25	430	907.23	1050.20	928.58	11.58%
c2670	233	130	701	666.85	772.68	687.87	10.98%
c3540	50	22	844	1214.60	1401.30	1251.00	10.73%
c5315	178	109	1204	925.32	1073.90	948.92	11.64%
c6288	32	32	3090	2799.90	3241.60	2889.10	10.87%
c7552	207	95	1624	2078.00	2394.40	2190.40	8.52%
s27	4	1	12	90.31	103.31	93.92	9.09%
s208	11	2	71	244.17	283.43	250.49	11.62%
s298	3	6	85	382.15	442.61	392.90	11.23%
s349	9	11	109	475.84	548.09	489.19	10.75%
s382	3	6	113	343.69	397.93	352.36	11.45%
s386	7	7	106	324.33	374.66	334.00	10.85%
s400	3	6	118	315.13	363.82	323.39	11.11%
s420	19	2	143	435.85	503.97	448.16	11.07%
s444	3	6	142	408.19	471.30	420.74	10.73%
s510	19	7	163	383.36	443.56	394.95	10.96%
s526	3	6	156	376.91	436.62	387.38	11.28%
s641	35	22	140	678.78	788.12	697.35	11.52%
s713	35	21	146	706.48	818.67	728.35	11.03%
s820	18	19	214	581.96	670.07	606.92	9.42%
s832	18	19	219	570.23	656.60	594.58	9.45%
s838	35	2	287	809.73	934.57	833.45	10.82%
s953	16	23	316	393.73	453.71	404.57	10.83%
s1196	14	14	358	600.55	694.15	614.65	11.45%
s1238	14	14	387	685.86	788.32	709.00	10.06%
s1423	17	5	432	2268.20	2609.30	2340.40	10.31%
s1488	8	19	389	914.21	1053.00	945.43	10.22%
s1494	8	19	395	950.88	1095.10	983.81	10.16%
s9234	36	19	1759	1431.30	1644.40	1513.00	7.99%
s13207	62	246	2900	1504.30	1738.00	1571.60	9.57%
s35932	35	320	13278	15647.00	17958.00	17302.00	3.65%
s38417	28	19	8917	5402.30	6207.30	5892.20	5.08%
						Average	10.34%

relative improvement of the minimum clock period T_{min} is 1.3x fold for nonzero clock skew circuits. Up to a 2.5x fold is seen in some circuits, although no particular topology tends to have more of an improvement than another.

$$\%\ \text{Improvement}^{\text{NZCS}}_{\text{SSTA}} = \frac{T^{NZCS}_{min,ss} - T^{NZCS}_{min,ssta}}{T^{NZCS}_{min,ss}} \times 100\% \qquad (6.3)$$

Table 6.3: Summary of Relative T_{min} Improvement from SSTA for ZS vs. NZS Circuits

Circuit	# Gates	# Registers	% Improvement$_{ssta}^{zcs}$	% Improvement$_{ssta}^{NZCS}$	Rel. Improv.
c17	8	0	12.26%	27.34%	2.23x
c432	152	0	11.17%	14.39%	1.29x
c499	405	0	10.67%	13.27%	1.24x
c880	241	0	11.86%	12.44%	1.05x
c1355	412	0	10.59%	13.23%	1.25x
c1908	430	0	11.58%	11.20%	0.97x
c2670	701	0	10.98%	10.37%	0.94x
c3540	844	0	10.73%	11.66%	1.09x
c5315	1204	0	11.64%	9.11%	0.78x
c6288	3090	0	10.87%	10.92%	1.00x
c7552	1624	0	8.52%	9.19%	1.08x
s27	12	3	9.09%	9.63%	1.06x
s208	71	8	11.62%	24.25%	2.09x
s298	85	14	11.23%	12.33%	1.10x
s349	109	15	10.75%	13.36%	1.24x
s382	113	21	11.45%	10.38%	0.91x
s386	106	6	10.85%	10.85%	1.00x
s400	118	21	11.11%	10.71%	0.96x
s420	143	16	11.07%	24.50%	2.21x
s444	142	21	10.73%	21.53%	2.01x
s510	163	6	10.96%	15.43%	1.41x
s526	156	21	11.28%	12.39%	1.10x
s641	140	19	11.52%	13.06%	1.13x
s713	146	19	11.03%	12.77%	1.16x
s820	214	5	9.42%	10.03%	1.06x
s832	219	5	9.45%	5.91%	0.63x
s838	287	32	10.82%	21.13%	1.95x
s953	316	29	10.83%	10.76%	0.99x
s1196	358	18	11.45%	11.86%	1.04x
s1238	387	18	10.06%	13.02%	1.29x
s1423	432	74	10.31%	10.50%	1.02x
s1488	389	6	10.22%	10.89%	1.07x
s1494	395	6	10.16%	10.81%	1.06x
s9234	1759	192	7.99%	10.84%	1.36x
s13207	2900	511	9.57%	8.46%	0.88x
s35932	13278	1728	3.65%	9.00%	2.46x
s38417	8917	1535	5.08%	9.67%	1.90x
				Average	1.27x

$$\text{Relative Improvement} = \frac{\%\ \text{Improvement}_{\text{SSTA}}^{\text{NZCS}}}{\%\ \text{Improvement}_{\text{SSTA}}^{\text{zcs}}} \tag{6.4}$$

6.5 CSS & SSTA Combined Improvement from Deterministic Models

For a comprehensive analysis, the potential combined improvement using clock skew scheduling and statistical timing analysis over the baseline worst case corner analysis is assessed. Results are shown in Table 6.4. The performance improvement, as defined by Equation 6.5, and flexibility possible with applying both of these strategies is rather compelling with an average of a ∼38% improvement in the minimum clock period T_{min}.

$$\% \text{ Improvement}_{\text{CSS\&SSTA}} = \frac{T^{zcs}_{min,ss} - T^{NZCS}_{min,ssta}}{T^{zcs}_{min,ss}} \times 100\% \tag{6.5}$$

6.6 CSS Improvement with Deterministic Models vs. Statistical Models

Although the improvement in the minimum clock period T_{min} observed with clock skew scheduling is approximately the same for circuits using deterministic and statistical modeling, as expected, the limiting topology (i.e. local data path, cycle, or reconvergent paths) is found to be different in ∼11% of the circuits. However, there is no distinguishable pattern as to which types of circuits would see such a change. It can only be concluded, then, that this may be the case for when multiple paths, cycles, or reconvergent branches are *near critical*. This reinforces the fact that statistical modeling is essential in accurately determining which portions of the circuit are limiting T_{min}.

6.7 SSTA Run Time for ZCS vs. NZCS Circuits

The CPU time required to perform statistical static timing analysis on nonzero clock skew circuits is compared with that required for zero clock skew circuits in Table 6.5. The slowdown in NZCS analysis is measured as shown in Equation 6.6. The added performance uncovered by applying SSTA to nonzero clock skew circuits comes at a modest 2.16x increase in analysis time on average, although this slowdown is up to ∼5.5x for some larger circuits with highly connected graphs. This can be attributed to the numerous computations needed for nonzero clock skew timing limitations, as well as the topological complexity of sequential circuits that impacts the time required for graph traversal & analysis algorithms. In particular, algorithmic improvements for the detection of reconvergent paths should be investigated.

Table 6.4: Summary of Overall T_{min} Improvement Using SSTA & Skew Scheduling

Circuit	# Gates	# Registers	% Impr.$^{zcs}_{ssta}$	% Impr.$^{det}_{CSS}$	% Impr.$_{CSS\&SSTA}$
c17	8	0	12.26%	48.33%	62.46%
c432	152	0	11.17%	49.15%	56.46%
c499	405	0	10.67%	38.29%	46.48%
c880	241	0	11.86%	10.15%	21.33%
c1355	412	0	10.59%	34.21%	42.92%
c1908	430	0	11.58%	29.63%	37.52%
c2670	701	0	10.98%	62.95%	66.79%
c3540	844	0	10.73%	46.99%	53.17%
c5315	1204	0	11.64%	73.89%	76.26%
c6288	3090	0	10.87%	33.62%	40.88%
c7552	1624	0	8.52%	9.43%	17.76%
s27	12	3	9.09%	14.22%	22.48%
s208	71	8	11.62%	51.57%	63.32%
s298	85	14	11.23%	11.92%	22.78%
s349	109	15	10.75%	50.05%	56.73%
s382	113	21	11.45%	34.18%	41.01%
s386	106	6	10.85%	17.50%	26.45%
s400	118	21	11.11%	34.97%	41.93%
s420	143	16	11.07%	71.92%	78.80%
s444	142	21	10.73%	37.90%	51.27%
s510	163	6	10.96%	20.96%	33.16%
s526	156	21	11.28%	12.08%	22.98%
s641	140	19	11.52%	19.35%	29.88%
s713	146	19	11.03%	18.63%	29.02%
s820	214	5	9.42%	11.45%	20.33%
s832	219	5	9.45%	11.69%	16.91%
s838	287	32	10.82%	78.77%	83.26%
s953	316	29	10.83%	14.06%	23.31%
s1196	358	18	11.45%	17.30%	27.11%
s1238	387	18	10.06%	48.52%	55.22%
s1423	432	74	10.31%	14.42%	23.41%
s1488	389	6	10.22%	6.95%	17.09%
s1494	395	6	10.16%	6.69%	16.77%
s13207	2900	511	9.57%	13.19%	20.53%
s35932	13278	1728	3.65%	0.20%	9.18%
s38417	8917	1535	5.08%	12.92%	21.34%
				Average	38.23%

$$\text{Slowdown} = \frac{\text{Run Time}^{NZCS}_{ssta}}{\text{Run Time}^{zcs}_{ssta}} \tag{6.6}$$

Table 6.5: SSTA CPU Run Time & Nonzero Clock Skew Circuit Slowdown

Circuit	# Gates	# Registers	Run Time$^{zcs}_{ssta}$[s]	Run Time$^{NZCS}_{ssta}$[s]	Slowdown
c17	8	0	0.012	0.016	1.33x
c432	152	0	0.188	0.196	1.04x
c499	405	0	0.516	0.576	1.12x
c880	241	0	0.296	0.332	1.12x
c1355	412	0	0.520	0.588	1.13x
c1908	430	0	0.556	0.580	1.04x
c2670	701	0	0.932	1.092	1.17x
c3540	844	0	1.104	1.152	1.04x
c5315	1204	0	1.692	1.800	1.06x
c6288	3090	0	5.128	5.224	1.02x
c7552	1624	0	2.396	2.476	1.03x
s27	12	3	0.020	0.028	1.40x
s208	71	8	0.088	0.168	1.91x
s298	85	14	0.104	0.252	2.42x
s349	109	15	0.136	0.312	2.29x
s382	113	21	0.144	0.484	3.36x
s386	106	6	0.132	0.292	2.21x
s400	118	21	0.152	0.516	3.39x
s420	143	16	0.184	0.368	2.00x
s444	142	21	0.180	0.708	3.93x
s510	163	6	0.200	0.508	2.54x
s526	156	21	0.200	0.544	2.72x
s641	140	19	0.180	0.608	3.38x
s713	146	19	0.188	0.620	3.30x
s820	214	5	0.268	0.464	1.73x
s832	219	5	0.272	0.460	1.69x
s838	287	32	0.384	0.804	2.09x
s953	316	29	0.408	1.260	3.09x
s1196	358	18	0.448	0.676	1.51x
s1238	387	18	0.488	0.708	1.45x
s1423	432	74	0.632	3.501	5.54x
s1488	389	6	0.488	0.944	1.93x
s1494	395	6	0.496	1.000	2.02x
s9234	1759	192	2.948	14.969	5.08x
s13207	2900	511	6.532	12.905	1.98x
s35932	13278	1728	66.824	133.664	2.00x
s38417	8917	1535	40.138	110.463	2.75x
				Average	2.16x

Chapter 7. Conclusions & Future Work

This thesis has shown statistical static timing analysis (SSTA) to be of particular importance in discovering the maximum performance gain possible with clock skew scheduling. Nonzero clock skew circuits suffer from the pessimism of traditional deterministic corner based static timing analysis in three separate timing limitations. This pessimism is compounded because the frequency limits of skew scheduled circuits depend not only on the slowest paths in the circuit, but also on the quickest paths and the relative speeds between paths. Nonzero clock skew circuits are seen to benefit from SSTA by up to 2.5x (1.3x on average) the amount seen by their zero skew counterparts, and by mitigating pessimism with SSTA, the minimum clock period T_{min} is seen to improve, on average, an additional 8.5% above what clock skew scheduling alone can achieve. An average clock period improvement of 38.25% is seen by applying both strategies, assuming a target yield of 99.73%. Futhermore, it has been found that the frequency limiting local data path, cycle, or reconvergent register pair in such circuits may change with more accurate statistical modeling, which would impact optimization applications as well as the results of skew scheduling itself.

The additional circuit performance uncovered by applying SSTA to nonzero clock skew circuits required twice the computation time, on average. This slowdown could likely be lessened by more efficient graph algorithms for locating and keeping track of graph cycles and reconvergent paths. Although a linear increase in run time would be acceptable given the performance benefit of applying SSTA to nonzero clock skew circuits, this slowdown is expected to be exacerbated in larger industrial circuits. Furthermore, criticality heuristics could be used in order to identify a subset of the entire circuit that requires the most accurate (i.e. time consuming) analysis. Such methods are presently being investigated in order to prune circuit graphs for in-depth path-based SSTA and higher order accuracy SSTA operations [33].

Future directions for this work can broadly be classified as those that improve accuracy (e.g. wire analysis, correlation analysis, non-gaussian methods) and those that involve the applications of statistical analysis (e.g. yield optimization, variation aware skew scheduling & delay insertion).

7.1 Wire & Clock Network Delay

Interconnect wire parasitics within integrated circuits are becoming responsible for a larger portion of the overall propagation delay of a signal, and the delays along longer wires are worsened by crosstalk with neighboring wire tracks [20]. Variations in wire dimensions can easily be handled using the same canonical form discussed in this thesis. With commercial layout synthesis, placement and routing tools, wire parasitics and more precise output loading capacitances can be extracted. These data could be incorporated in to this study along with variation information on the wires constituting the clock distribution network in order to achieve a higher order accuracy. Furthermore, recent studies which attempt to account for crosstalk in a statistical manner should also be investigated for analysis of both local data paths as well as clock networks [34, 35].

7.2 Correlation Analysis, Optimization, Variation Aware Scheduling & Delay Insertion

The concept of criticality is rather simple in static timing analysis since for each $MAX()$ operation performed, there is one operand that dominates 100% (is *critical*). This notion changes with SSTA as operands to the $MAX()$ function assume a non-integer tightness probability representing their criticality. Keeping track of these tightness probabilities as well as spatial correlations between paths that share gates [14, 33] is very important for optimizing a circuit for timing, power, and/or area [8, 36, 37]. Recent research has investigated efficient ways of finding statistical criticalities without full path-based analysis [38] such that algorithms can quickly find which gates or paths to modify in order for the circuit to meet these different constraints. These concepts could be extended to this work in order to perform intelligent clock skew scheduling and delay insertion in nonzero clock skew circuits [12, 39].

7.3 Non-Gaussian Variation, Non-Linear Sensitivity

The statistical $MAX()$ function discussed in this thesis is slightly pessimistic itself and introduces minor inaccuracies in assuming the results of a statistical $MAX()$, $MIN()$, $ADD()$, or $SUB()$ to have a perfect Gaussian distribution. As both the complexity and dimensionality of performance-impacting process variations are growing, inclusion of more variations is bound to exacerbate these inaccuracies. Similarly, some of these variations impose non-linear performance changes unlike L_e and V_T as discussed in this work. This study should be later extended to utilize non-Gaussian mathematics and non-linear sensitivites that are currently under investigation [40, 18, 19].

Bibliography

[1] Kunhyuk Kang, B. C. Paul, and K. Roy. Statistical timing analysis using levelized covariance propagation. In *Design, Automation and Test in Europe, 2005. Proceedings*, pages 764–769, 2005.

[2] Andrew B. Kahng. A roadmap and vision for physical design. In *ISPD '02: Proceedings of the 2002 international symposium on Physical design*, pages 112–117, New York, NY, USA, 2002. ACM Press.

[3] Chandu Visweswariah. Death, taxes and failing chips. In *DAC '03: Proceedings of the 40th conference on Design automation*, pages 343–347, New York, NY, USA, 2003. ACM Press.

[4] Anirudh Devgan and Chandramouli Kashyap. Block-based static timing analysis with uncertainty. In *ICCAD '03: Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design*, page 607, Washington, DC, USA, 2003. IEEE Computer Society.

[5] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan. First-order incremental block-based statistical timing analysis. In *DAC '04: Proceedings of the 41st annual conference on Design automation*, pages 331–336, New York, NY, USA, 2004. ACM Press.

[6] Animesh Datta, Swarup Bhunia, Jung Hwan Choi, Saibal Mukhopadhyay, and Kaushik Roy. Speed binning aware design methodology to improve profit under parameter variations. In *ASP-DAC '06: Proceedings of the 2006 conference on Asia South Pacific design automation*, pages 712–717, New York, NY, USA, 2006. ACM Press.

[7] S. R. Nassif, V. Pitchumani, N. Rodriguez, D. Sylvester, C. Bittlestone, and R. Radojcic. Variation-aware analysis: savior of the nanometer era? In *DAC '06: Proceedings of the 43rd annual conference on Design automation*, pages 411–412, New York, NY, USA, 2006. ACM Press.

[8] D. Sinha, N. V. Shenoy, and Hai Zhou. Statistical gate sizing for timing yield optimization. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 1037–1041, Washington, DC, USA, 2005. IEEE Computer Society.

[9] K. Chopra, S. Shah, A. Srivastava, D. Blaauw, and D. Sylvester. Parametric yield maximization using gate sizing based on efficient statistical power and delay gradient computation. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 1023–1028, Washington, DC, USA, 2005. IEEE Computer Society.

[10] Aseem Agarwal, Kaviraj Chopra, David Blaauw, and Vladimir Zolotov. Circuit optimization using statistical static timing analysis. In *DAC '05: Proceedings of the 42nd annual conference on Design automation*, pages 321–324, New York, NY, USA, 2005. ACM Press.

[11] Xun Liu, Marios C. Papaefthymiou, and Eby G. Friedman. Maximizing performance by retiming and clock skew scheduling. In *DAC '99: Proceedings of the 36th ACM/IEEE conference on Design automation*, pages 231–236, New York, NY, USA, 1999. ACM Press.

[12] Baris Taskin and Ivan S. Kourtev. Delay insertion method in clock skew scheduling. In *ISPD '05: Proceedings of the 2005 international symposium on Physical design*, pages 47–54, New York, NY, USA, 2005. ACM Press.

[13] Louis Scheffer, Luciano Lavagno, and Grant Martin. *EDA for IC System Design, Verification, and Testing (Electronic Design Automation for Integrated Circuits Handbook).* CRC Press, Inc., Boca Raton, FL, USA, 2006.

[14] Yaping Zhan, A. J. Strojwas, M. Sharma, and D. Newmark. Statistical critical path analysis considering correlations. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 699–704, Washington, DC, USA, 2005. IEEE Computer Society.

[15] S. R. Nassif. Modeling and forecasting of manufacturing variations. In *Design Automation Conference, 2001. Proceedings of the ASP-DAC 2001. Asia and South Pacific*, pages 145–149, January/February 2001.

[16] Hanif Fatemi, Shahin Nazarian, and Massoud Pedram. Statistical logic cell delay analysis using a current-based model. In *DAC '06: Proceedings of the 43rd annual conference on Design automation*, pages 253–256, New York, NY, USA, 2006. ACM Press.

[17] Wai-Kai Chen. *The VLSI Handbook, Second Edition (Electrical Engineering Handbook).* CRC Press, Inc., Boca Raton, FL, USA, 2006.

[18] Yaping Zhan, Andrzej J. Strojwas, Xin Li, Lawrence T. Pileggi, David Newmark, and Mahesh Sharma. Correlation-aware statistical timing analysis with non-gaussian delay distributions. In *DAC '05: Proceedings of the 42nd annual conference on Design automation*, pages 77–82, New York, NY, USA, 2005. ACM Press.

[19] Lizheng Zhang, Weijen Chen, Yuhen Hu, John A. Gubner, and Charlie Chung-Ping Chen. Correlation-preserved non-gaussian statistical timing analysis with quadratic timing model. In *DAC '05: Proceedings of the 42nd annual conference on Design automation*, pages 83–88, New York, NY, USA, 2005. ACM Press.

[20] S. R. Nassif. Modeling and analysis of manufacturing variations. In *Custom Integrated Circuits, 2001, IEEE Conference on.*, pages 223–228, San Diego, CA, May 2001.

[21] Charles E. Clark. The greatest of a finite set of random variables. *Operations Research*, 9(2):145–162, mar 1961.

[22] Michael Cain. The moment-generating function of the minimum of bivariate normal random variables. *The American Statistician*, 48(2):124–125, may 1994.

[23] Farid N. Najm. On the need for statistical timing analysis. In *DAC '05: Proceedings of the 42nd annual conference on Design automation*, pages 764–765, New York, NY, USA, 2005. ACM Press.

[24] Shiy Xu and E. Edirisuriya. A new way of detecting reconvergent fanout branch pairs in logic circuits. In *ATS '04: Proceedings of the 13th Asian Test Symposium (ATS'04)*, pages 354–357, Washington, DC, USA, 2004. IEEE Computer Society.

[25] Milton Abramowitz. *Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables,.* Dover Publications, Incorporated, 1974.

[26] Wei Zhao and Yu Cao. Predictive technology model for nano-cmos design exploration. *J. Emerg. Technol. Comput. Syst.*, 3(1):1, 2007.

[27] Wei Zhao and Yu Cao. New generation of predictive technology model for sub-45nm design exploration. In *ISQED '06: Proceedings of the 7th International Symposium on Quality Electronic Design*, pages 585–590, Washington, DC, USA, 2006. IEEE Computer Society.

[28] Rung-Bin Lin and Meng-Chiou Wu. A new statistical approach to timing analysis of vlsi circuits. In *VLSID '98: Proceedings of the Eleventh International Conference on VLSI Design: VLSI for Signal Processing*, page 507, Washington, DC, USA, 1998. IEEE Computer Society.

[29] Vaibhav Nawale and Thomas W. Chen. Optimal useful clock skew scheduling in the presence of variations using robust ilp formulations. In *ICCAD '06: Proceedings of the 2006 IEEE/ACM international conference on Computer-aided design*, pages 27–32, New York, NY, USA, 2006. ACM Press.

[30] *Ngspice Circuit Simulator.* http://ngspice.sourceforge.net.

[31] University of California at Berkeley, http://bwrc.eecs.berkeley.edu/Classes/IcBook/SPICE/. *Spicef5.*

[32] Chenming Hu. BSIM model for circuit design using advanced technologies. In *VLSI Circuits, 2001. Digest of Technical Papers. 2001 Symposium on*, pages 5–10, Kyoto, Japan, 2001.

[33] Jinjun Xiong, Vladimir Zolotov, Natesan Venkateswaran, and Chandu Visweswariah. Criticality computation in parameterized statistical timing. In *DAC '06: Proceedings of the 43rd annual conference on Design automation*, pages 63–68, New York, NY, USA, 2006. ACM Press.

[34] B. Choi and D. M. H. Walker. Timing analysis of combinational circuits including capacitive-coupling and statistical process variation. In *VLSI Test Symposium, 2000. Proceedings. 18th IEEE*, pages 49–54, Montreal, Que., Canada, 2000.

[35] D. Sinha and Hai Zhou. A unified framework for statistical timing analysis with coupling and multiple input switching. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 837–843, Washington, DC, USA, 2005. IEEE Computer Society.

[36] M. R. Guthaus, N. Venkateswarant, C. Visweswariaht, and V. Zolotov. Gate sizing using incremental parameterized statistical timing analysis. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 1029–1036, Washington, DC, USA, 2005. IEEE Computer Society.

[37] Aseem Agarwal, Kaviraj Chopra, and David Blaauw. Statistical timing based optimization using gate sizing. In *DATE '05: Proceedings of the conference on Design, Automation and Test in Europe*, pages 400–405, Washington, DC, USA, 2005. IEEE Computer Society.

[38] Xin Li, Jiayong Le, Mustafa Celik, and L. T. Pileggi. Defining statistical sensitivity for timing optimization of logic circuits with large-scale process and environmental variations. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 844–851, Washington, DC, USA, 2005. IEEE Computer Society.

[39] T. Yoda, A. Takahashi, and Y. Kajitani. Clock period minimization of semi-synchronous circuits bygate-level delay insertion. In *Design Automation Conference, 1999. Proceedings of the ASP-DAC '99. Asia and South Pacific*, pages 125–128, Wanchai, Hong Kong, January 1999.

[40] Lizheng Zhang, Jun Shao, and Charlie Chung-Ping Chen. Non-gaussian statistical parameter modeling for ssta with confidence interval analysis. In *ISPD '06: Proceedings of the 2006 international symposium on Physical design*, pages 33–38, New York, NY, USA, 2006. ACM Press.

[41] Chirayu S. Amin, Noel Menezes, Kip Killpack, Florentin Dartu, Umakanta Choudhury, Nagib Hakim, and Yehea I. Ismail. Statistical static timing analysis: how simple can we get? In *DAC '05: Proceedings of the 42nd annual conference on Design automation*, pages 652–657, New York, NY, USA, 2005. ACM Press.

[42] Osama Neiroukh and Xiaoyu Song. Improving the process-variation tolerance of digital circuits using gate sizing and statistical techniques. In *DATE '05: Proceedings of the conference on Design, Automation and Test in Europe*, pages 294–299, Washington, DC, USA, 2005. IEEE Computer Society.

[43] Masanori Hashimoto and Hidetoshi Onodera. A performance optimization method by gate sizing using statistical static timing analysis. In *ISPD '00: Proceedings of the 2000 international symposium on Physical design*, pages 111–116, New York, NY, USA, 2000. ACM Press.

[44] Massimo Conti, Paolo Crippa, Simone Orcioni, Marcello Pesare, Claudio Turchetti, Loris Vendrame, and Silvia Lucherini. An integrated cad methodology for yield enhancement of vlsi cmos circuits including statistical device variations. *Analog Integr. Circuits Signal Process.*, 37(2):85–102, 2003.

[45] Olivier Coudert, Ramsey Haddad, and Srilatha Manne. New algorithms for gate sizing: a comparative study. In *DAC '96: Proceedings of the 33rd annual conference on Design automation*, pages 734–739, New York, NY, USA, 1996. ACM Press.

[46] Hratch Mangassarian and Mohab Anis. On statistical timing analysis with inter- and intra-die variations. In *DATE '05: Proceedings of the conference on Design, Automation and Test in Europe*, pages 132–137, Washington, DC, USA, 2005. IEEE Computer Society.

[47] Joseph F. Ryan, Jiajing Wang, and Benton H. Calhoun. Analyzing and modeling process balance for sub-threshold circuit design. In *GLSVLSI '07: Proceedings of the 17th great lakes symposium on Great lakes symposium on VLSI*, pages 275–280, New York, NY, USA, 2007. ACM Press.

[48] Xiaoliang Bai, Chandu Visweswariah, and Philip N. Strenski. Uncertainty-aware circuit optimization. In *DAC '02: Proceedings of the 39th conference on Design automation*, pages 58–63, New York, NY, USA, 2002. ACM Press.

[49] E. T. A. F. Jacobs and M. R. C. M. Berkelaar. Gate sizing using a statistical delay model. In *DATE '00: Proceedings of the conference on Design, automation and test in Europe*, pages 283–291, New York, NY, USA, 2000. ACM Press.

[50] Sreeja Raj, Sarma B. K. Vrudhula, and Janet Wang. A methodology to improve timing yield in the presence of process variations. In *DAC '04: Proceedings of the 41st annual conference on Design automation*, pages 448–453, New York, NY, USA, 2004. ACM Press.

[51] J. A. G. Jess, K. Kalafala, S. R. Naidu, R. H. J. M. Otten, and C. Visweswariah. Statistical timing for parametric yield prediction of digital integrated circuits. In *DAC '03: Proceedings of the 40th conference on Design automation*, pages 932–937, New York, NY, USA, 2003. ACM Press.

[52] Lou Scheffer. Explicit computation of performance as a function of process variation. In *TAU '02: Proceedings of the 8th ACM/IEEE international workshop on Timing issues in the specification and synthesis of digital systems*, pages 1–8, New York, NY, USA, 2002. ACM Press.

[53] Kenta Yamada and Noriaki Oda. Statistical corner conditions of interconnect delay (corner lpe specifications). In *ASP-DAC '06: Proceedings of the 2006 conference on Asia South Pacific design automation*, pages 706–711, New York, NY, USA, 2006. ACM Press.

[54] J. L. Neves and E. G. Friedman. Topological design of clock distribution networks based on nonzeroclock skew specifications. In *Circuits and Systems, 1993., Proceedings of the 36th Midwest Symposium on*, pages 468–471, Detroit, MI, USA, August 1993.

[55] J. L. Neves and E. G. Friedman. Design methodology for synthesizing clock distribution networksexploiting nonzero localized clock skew. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 4(2):286–291, June 1996.

Appendix A. List of Symbols

L_e	MOSFET channel length
V_T	MOSFET threshold voltage
C_L	Output load capacitance
tt_{in}	Input transition slope
f_{Max}	Maximum operating frequency
T_{min}	Minimum clock period
τ_P	Logic gate propagation delay.
τ_{Pmin}	Minimum deterministic logic gate propagation delay
τ_{PMax}	Maximum deterministic logic gate propagation delay
$\delta_S^{R_i}$	Setup time of a register R_i
$\delta_H^{R_i}$	Hold time of a register R_i
$d_{CQm}^{R_i}$	Minimum deterministic clock-to-output time of a register R_i
$d_{CQM}^{R_i}$	Maximum deterministic clock-to-output time of a register R_i
p^{if_x}	A particular local data path “x” between registers R_i and R_f
$d_{Pm}^{if_x}$	Minimum total deterministic propagation delay between registers R_i and R_f on a local data path p^{if_x}
$d_{PM}^{if_x}$	Maximum total deterministic propagation delay between registers R_i and R_f on a local data path p^{if_x}
$D_{Pm}^{if_x}$	Random variable for minimum total propagation delay between registers R_i and R_f on a local data path p^{if_x}
$D_{PM}^{if_x}$	Random variable for maximum total propagation delay between registers R_i and R_f on a local data path p^{if_x}
AT_{min}^g	Minimum deterministic arrival time at a gate or register g
AT_{Max}^g	Maximum deterministic arrival time at a gate or register g
p^{if}	The local data paths between registers R_i and R_f
d_{Pm}^{if}	Minimum total deterministic propagation delay among all local data paths between registers R_i and R_f
d_{PM}^{if}	Maximum total deterministic propagation delay among all local data paths between registers R_i and R_f
D_{Pm}^{if}	Random variable for minimum total propagation delay among all local data paths between registers R_i and R_f
D_{PM}^{if}	Random variable for maximum total propagation delay among all local data paths between registers R_i and R_f

$pd_m^{p^i}$	Minimum total deterministic propagation delay between reconvergent registers R_d and R_c on reconvergent path p^i
$pd_M^{p^i}$	Maximum total deterministic propagation delay between reconvergent registers R_d and R_c on reconvergent path p^i
$PD_m^{p^i}$	Random variable for minimum total propagation delay between reconvergent registers R_d and R_c on reconvergent path p^i
$PD_M^{p^i}$	Random variable for maximum total propagation delay between reconvergent registers R_d and R_c on reconvergent path p^i
μ	Mean
σ	Standard deviation
σ^2	Variance
s_A^X	Sensitivity of a random variable A to variation in a random variable X
$s_{\tau_P}^{L_e}$	Logic gate propagation delay sensitivity to L_e variation
$s_{\tau_P}^{V_T}$	Logic gate propagation delay sensitivity to V_T variation
T_A	Tightness probability of A for use in the $MAX()$ or $MIN()$ function
T_{min}^{zcs}	Minimum deterministic clock period of a zero clock skew circuit
T_{min}^{NZCS}	Minimum deterministic clock period of a nonzero clock skew circuit
$T_{min}^{NZCS,I}$	Minimum deterministic clock period of a nonzero clock skew circuit (limit I)
$T_{min}^{NZCS,II}$	Minimum deterministic clock period of a nonzero clock skew circuit (limit II)
$T_{min}^{NZCS,III}$	Minimum deterministic clock period of a nonzero clock skew circuit (limit III)
$T_{min,ssta}^{zcs}$	Minimum statistical (μ+3σ) clock period of a zero clock skew circuit)
$T_{min,ssta}^{NZCS}$	Minimum statistical (μ+3σ) clock period of a nonzero clock skew circuit)
$T_{min,ssta}^{NZCS,I}$	Minimum statistical (μ+3σ) clock period of a nonzero clock skew circuit (limit I)
$T_{min,ssta}^{NZCS,II}$	Minimum statistical (μ+3σ) clock period of a nonzero clock skew circuit (limit II)
$T_{min,ssta}^{NZCS,III}$	Minimum statistical (μ+3σ) clock period of a nonzero clock skew circuit (limit III)

Appendix B. Example Cell Spice Definition

```
*********************************
* OAI21
*********************************

.subckt oai21_a a1 a2 b O VDD VSS
mp1 1 a1 vdd vdd pmos L=.090u W=.48u
mp2 O a2 1   vdd pmos L=.090u W=.48u
mp3 O b  vdd vdd pmos L=.090u W=.24u
mn1 2 a1 vss vss nmos L=.090u W=.24u
mn2 2 a2 vss vss nmos L=.090u W=.24u
mn3 O b  2   vss nmos L=.090u W=.24u
.ends

.subckt oai21_b a1 a2 b O VDD VSS
mp1 1 a1 vdd vdd pmos L=.090u W=.96u
mp2 O a2 1   vdd pmos L=.090u W=.96u
mp3 O b  vdd vdd pmos L=.090u W=.48u
mn1 2 a1 vss vss nmos L=.090u W=.48u
mn2 2 a2 vss vss nmos L=.090u W=.48u
mn3 O b  2   vss nmos L=.090u W=.48u
.ends

.subckt oai21_c a1 a2 b O VDD VSS
mp1 1 a1 vdd vdd pmos L=.090u W=1.92u
mp2 O a2 1   vdd pmos L=.090u W=1.92u
mp3 O b  vdd vdd pmos L=.090u W=.96u
mn1 2 a1 vss vss nmos L=.090u W=.96u
mn2 2 a2 vss vss nmos L=.090u W=.96u
mn3 O b  2   vss nmos L=.090u W=.96u
.ends

.subckt oai21_d a1 a2 b O VDD VSS
mp1 1 a1 vdd vdd pmos L=.090u W=3.84u
mp2 O a2 1   vdd pmos L=.090u W=3.84u
mp3 O b  vdd vdd pmos L=.090u W=1.92u
mn1 2 a1 vss vss nmos L=.090u W=1.92u
mn2 2 a2 vss vss nmos L=.090u W=1.92u
mn3 O b  2   vss nmos L=.090u W=1.92u
.ends
```

Appendix C. Example Cell Characterization Data

```
gate&sz, ttCl, nom[ps], ssss[ps], s(vt)[ps/mV], s(le)[ps/nm], Cin(um), SlOut[ps]
nand2_a, fo2_tt1, 1.96e+01, 2.29e+01, 8.80e-03, 9.26e-02, 0.48, 3.31e+01,
nand2_a, fo2_tt2, 2.10e+01, 2.43e+01, 9.35e-03, 9.44e-02, 0.48, 3.31e+01,
nand2_a, fo4_tt1, 2.73e+01, 3.18e+01, 1.10e-02, 1.39e-01, 0.48, 5.35e+01,
nand2_a, fo4_tt2, 2.87e+01, 3.32e+01, 1.21e-02, 1.43e-01, 0.48, 5.35e+01,
nand2_b, fo2_tt1, 1.57e+01, 1.84e+01, 7.70e-03, 7.04e-02, 0.96, 2.26e+01,
nand2_b, fo2_tt2, 1.71e+01, 1.98e+01, 8.25e-03, 7.04e-02, 0.96, 2.26e+01,
nand2_b, fo4_tt1, 1.96e+01, 2.30e+01, 9.35e-03, 9.44e-02, 0.96, 3.33e+01,
nand2_b, fo4_tt2, 2.10e+01, 2.43e+01, 9.35e-03, 9.44e-02, 0.96, 3.32e+01,
nand2_c, fo2_tt1, 1.38e+01, 1.62e+01, 7.15e-03, 6.30e-02, 1.92, 1.68e+01,
nand2_c, fo2_tt2, 1.52e+01, 1.76e+01, 7.15e-03, 6.11e-02, 1.92, 1.69e+01,
nand2_c, fo4_tt1, 1.58e+01, 1.86e+01, 7.70e-03, 7.04e-02, 1.92, 2.28e+01,
nand2_c, fo4_tt2, 1.72e+01, 1.99e+01, 8.25e-03, 6.85e-02, 1.92, 2.28e+01,
nand2_d, fo2_tt1, 1.29e+01, 1.52e+01, 6.60e-03, 5.74e-02, 3.84, 1.41e+01,
nand2_d, fo2_tt2, 1.44e+01, 1.66e+01, 7.15e-03, 6.30e-02, 3.84, 1.42e+01,
nand2_d, fo4_tt1, 1.39e+01, 1.64e+01, 7.15e-03, 6.48e-02, 3.84, 1.72e+01,
nand2_d, fo4_tt2, 1.54e+01, 1.78e+01, 7.15e-03, 6.11e-02, 3.84, 1.72e+01,
nor4_a, fo4_tt2, 1.57e+01, 1.76e+01, 3.03e-02, 4.78e-01, 1.08, 1.13e+02,
nor4_b, fo2_tt1, 1.02e+01, 1.14e+01, 1.87e-02, 2.93e-01, 2.16, 7.22e+01,
nor4_b, fo2_tt2, 1.19e+01, 1.33e+01, 1.87e-02, 2.87e-01, 2.16, 7.27e+01,
nor4_b, fo4_tt1, 1.15e+01, 1.29e+01, 2.20e-02, 3.57e-01, 2.16, 8.60e+01,
nor4_b, fo4_tt2, 1.32e+01, 1.48e+01, 2.26e-02, 3.52e-01, 2.16, 8.64e+01,
nor4_c, fo2_tt1, 9.50e+00, 1.07e+01, 1.65e-02, 2.59e-01, 4.32, 6.63e+01,
nor4_c, fo2_tt2, 1.13e+01, 1.26e+01, 1.71e-02, 2.56e-01, 4.32, 6.68e+01,
nor4_c, fo4_tt1, 1.02e+01, 1.14e+01, 1.87e-02, 2.94e-01, 4.32, 7.32e+01,
nor4_c, fo4_tt2, 1.19e+01, 1.33e+01, 1.87e-02, 2.87e-01, 4.32, 7.37e+01,
nor4_d, fo2_tt1, 8.95e+00, 1.01e+01, 1.60e-02, 2.46e-01, 8.34, 6.50e+01,
nor4_d, fo2_tt2, 1.08e+01, 1.20e+01, 1.65e-02, 2.43e-01, 8.34, 6.53e+01,
nor4_d, fo4_tt1, 9.25e+00, 1.04e+01, 1.65e-02, 2.63e-01, 8.34, 6.86e+01,
nor4_d, fo4_tt2, 1.11e+01, 1.24e+01, 1.71e-02, 2.59e-01, 8.34, 6.90e+01,
oai21_a, fo2_tt1, 1.94e+01, 2.26e+01, 9.35e-03, 1.39e-01, 1.44, 2.92e+01,
oai21_a, fo2_tt2, 2.06e+01, 2.38e+01, 9.90e-03, 1.39e-01, 1.44, 2.91e+01,
oai21_a, fo4_tt1, 2.57e+01, 2.98e+01, 1.27e-02, 2.00e-01, 1.44, 4.40e+01,
oai21_a, fo4_tt2, 2.69e+01, 3.11e+01, 1.38e-02, 2.02e-01, 1.44, 4.38e+01,
oai21_b, fo2_tt1, 1.62e+01, 1.89e+01, 7.70e-03, 1.09e-01, 2.88, 2.18e+01,
oai21_b, fo2_tt2, 1.73e+01, 2.01e+01, 8.25e-03, 1.09e-01, 2.88, 2.18e+01,
oai21_b, fo4_tt1, 1.94e+01, 2.26e+01, 9.90e-03, 1.41e-01, 2.88, 2.93e+01,
oai21_b, fo4_tt2, 2.06e+01, 2.39e+01, 1.05e-02, 1.41e-01, 2.88, 2.92e+01,
oai21_c, fo2_tt1, 1.45e+01, 1.70e+01, 7.15e-03, 9.44e-02, 5.76, 1.78e+01,
oai21_c, fo2_tt2, 1.58e+01, 1.82e+01, 7.70e-03, 9.26e-02, 5.76, 1.80e+01,
oai21_c, fo4_tt1, 1.62e+01, 1.90e+01, 7.70e-03, 1.09e-01, 5.76, 2.18e+01,
oai21_c, fo4_tt2, 1.74e+01, 2.02e+01, 8.25e-03, 1.09e-01, 5.76, 2.19e+01,
```

Appendix D. Example 90nm MOSFET Predictive Modelcard

```
* Predictive Technology Model Beta Version
* 90nm NMOS SPICE Parameters

.model NMOS NMOS +Level = 49

+Lint = 1.5e-08 Tox = 2.5e-09
+Vth0 = 0.2607 Rdsw = 180

+lmin=1.0e-7 lmax=1.0e-7 wmin=1.0e-7 wmax=1.0e-4
+Tref=27.0   version =3.1
+Xj= 4.0000000E-08       Nch= 9.7000000E+17
+lln= 1.0000000          lwn= 1.0000000              wln= 0.00
+wwn= 0.00               ll= 0.00
+lw= 0.00                lwl= 0.00                   wint= 0.00
+wl= 0.00                ww= 0.00                    wwl= 0.00
+Mobmod=  1              binunit= 2                  xl= 0.00
+xw= 0.00                binflag=  0
+Dwg= 0.00               Dwb= 0.00
+ACM= 0                  ldif=0.00                   hdif=0.00
+rsh= 7                  rd= 0                       rs= 0
+rsc= 0                  rdc= 0
+K1= 0.3950000           K2= 1.0000000E-02           K3= 0.00
+Dvt0= 1.0000000         Dvt1= 0.4000000             Dvt2= 0.1500000
+Dvt0w= 0.00             Dvt1w= 0.00                 Dvt2w= 0.00
+Nlx= 4.8000000E-08      W0= 0.00                    K3b= 0.00
+Ngate= 5.0000000E+20
+Vsat= 1.1000000E+05     Ua= -6.0000000E-10          Ub= 8.0000000E-19
+Uc= -2.9999999E-11
+Prwb= 0.00              Prwg= 0.00                  Wr= 1.0000000
+U0= 1.7999999E-02       A0= 1.1000000               Keta= 4.0000000E-02
+A1= 0.00                A2= 1.0000000               Ags= -1.0000000E-02
+B0= 0.00                B1= 0.00
+Voff= -2.9999999E-02    NFactor= 1.5000000          Cit= 0.00
+Cdsc= 0.00              Cdscb= 0.00                 Cdscd= 0.00
+Eta0= 0.1500000         Etab= 0.00                  Dsub= 0.6000000
+Pclm= 0.1000000         Pdiblc1= 1.2000000E-02      Pdiblc2= 7.5000000E-03
+Pdiblcb= -1.3500000E-02 Drout= 2.0000000            Pscbe1= 8.6600000E+08
+Pscbe2= 1.0000000E-20   Pvag= -0.2800000            Delta= 1.0000000E-02
+Alpha0= 0.00            Beta0= 30.0000000
+kt1= -0.3700000         kt2= -4.0000000E-02         At= 5.5000000E+04
+Ute= -1.4800000         Ua1= 9.5829000E-10          Ub1= -3.3473000E-19
+Uc1= 0.00               Kt1l= 4.0000000E-09         Prt= 0.00
```

```
+Cj= 0.0015              Mj= 0.72                    Pb= 1.25
+Cjsw= 2E-10             Mjsw= 0.37                  Php= 0.773
+Cjgate= 2E-14           Cta= 0                      Ctp= 0
+Pta= 0                  Ptp= 0                      JS=1.50E-08
+JSW=2.50E-13            N=1.0                       Xti=3.0
+Cgdo=3.493E-10          Cgso=3.493E-10              Cgbo=0.0E+00
+Capmod= 2               NQSMOD= 0                   Elm= 5
+Xpart= 1                cgsl= 0.582E-10             cgdl= 0.582E-10
+ckappa= 0.28            cf= 1.177e-10               clc= 1.0000000E-07
+cle= 0.6000000          Dlc= 2E-08                  Dwc= 0
```