ABSTRACT

Title of thesis: EFFICIENT AND ACCURATE STATISTICAL TIMING ANALYSIS FOR NON-LINEAR NON-GAUSSIAN VARIABILITY WITH INCREMENTAL ATTRIBUTES

Ashish Dobhal, Master of Science, 2006

Thesis directed by: Assistant Professor Ankur Srivastava
Department of Electrical and Computer Engineering

For four decades, the semiconductor industry has distinguished itself by the rapid pace of improvement in its products. The principal categories of improvement trends are integration level, Cost, speed, power, compactness and functionality. Most of these trends have resulted principally from the industry's ability to exponentially decrease the minimum feature size used to fabricate integrated circuits. Of course, the most frequently cited trend is in integration level, which is usually expressed as Moore's Law (that is, the number of components per chip doubles every 24 months).

With the aggressive scaling of the recent technologies, process variability is growing. Dealing with variability has become an integral aspect of high performance digital integrated circuits and indispensable for first-time-right hardware and cutting-edge performance [1]. Mainly two reasons can be attributed to this trend. First, critical dimension of the circuit are scaling faster than our control on the manufacturing process, resulting in the proportionate increase in the variability of physical dimensions, such as the effective length of a transistor channel [2]. Second, atomic-scale randomness, e.g. variation in the number of dopant in the transistor channel, is increasing [3]. Under these conditions, it becomes essential for the design tools to account for the uncertainties and to design robust

circuits that are optimized for the device parameter variations.

Traditional design automation techniques (corner based techniques) are not capable of dealing with these fabrication variations. As a result, the design obtained may be sub-optimal or may not even satisfy the design constraints.Thus it becomes imperative to find new methods to handle the variations in the design flow which will be both accurate and fast. The objective is to develop techniques to create robust design that is not prone to the manufacturing variations and which helps in increasing the yield of the ICs.

In this work, we focus on the problem of timing analysis in the presence of fabrication variability. Generally, the timing analysis techniques which handle timing analysis in the presence of variability are called Statistical Static Timing Analysis due to obvious reasons. Unlike current approaches for non-linear, non-Gaussian SSTA [4] which have numerical components, our approach is completely analytical. We also investigate the incremental aspects of SSTA and present (1) a fast yet accurate incremental approach (2) a method to efficiently estimate the expected error injected by incremental SSTA, which can be used to decide when accurate SSTA should be executed and when incremental SSTA would suffice. Our approach (non-incremental) is about 9588 times faster than Monte Carlo whereas an existing state of the art non-linear, non-Gaussian SSTA engine [4] is only 31.3 times faster. Further, the Root Mean Square (RMS) error of both the approaches is comparable w.r.t. Monte Carlo. Our incremental approach is 23 times faster than the proposed accurate SSTA approach. Moreover, our error estimating methodology can accurately capture the trends of error injection due to incremental SSTA.

EFFICIENT AND ACCURATE STATISTICAL TIMING ANALYSIS FOR NON-LINEAR NON-GAUSSIAN VARIABILITY WITH INCREMENTAL ATTRIBUTES

by

Ashish Dobhal

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2006

Advisory Committee:
Professor Ankur Srivastava, Chair/Advisor
Professor Shuvra Bhattacharyya
Professor Gang Qu

ACKNOWLEDGMENTS

This thesis is the culmination of my two year stint at University of Maryland, College Park. During this period, I was supported and helped by many people both professionally and personally. It is a pleasant aspect that now I can express my gratitude for them.

I would like to thank my advisor Dr. Ankur Srivastava who kept an eye on the progress of my work and was always available when I needed him. I would also like to thank Dr Gang Qu with whom I interacted more during my responsibility as his Teaching Assistant. It was a great learning experience. I would also take this opportunity to thank my undergrad professors (Dr. Vimal Singh and Dr. Haranath Kar) who constantly helped me and provided guidance.

My research lab colleagues, Azadeh and Vishal, have been very kind to me. Their suggestions and ideas were very useful in my research.

Throughout my graduate studies, two people, Amit Apte and Bhuvan Middha, have always inspired and motivated me. It was them and my other friends (Chandru, Priyanka, Smitha, Sebestian and Alokika) whose company made these years simply exciting and fun. My apartment-mates (Anshul, Avinash and Piyush) have been very helpful. They deserve a big thanks from my side.

Last but not the least, I would like to thank my family for their support and understanding. It is their love which always keeps me inspiring to do better in my life.

TABLE OF CONTENTS

Chapter 1

Background and Motivation

1.0.1 Deep Sub-micron VLSI design

For four decades, the semiconductor industry has distinguished itself by the rapid pace of improvement in its products. The principal categories of improvement trends are integration level, Cost, speed, power, compactness and functionality. Most of these trends have resulted principally from the industry's ability to exponentially decrease the minimum feature size used to fabricate integrated circuits. Of course, the most frequently cited trend is integration level, which is usually expressed as Moore's Law (that is, the number of components per chip doubles every 24 months).

With ever increasing complexity of the IC designs, the CAD tool needed to design them need to evolve to produce better designs in the shortest possible time. Shrinking device, dimensions, increasing manufacturing and environmental variations has made fast design closure and high yield difficult [5].

In a chip design process, the chip designer operate between several different design activity. In this process, the designer moves between different abstraction levels, transitioning from higher to lower design representation. Design details increases at the lower levels of design representation. Each step or task of the design process, e.g. register-transfer-level (RTL) optimization, logic synthesis, place and route and timing verification, is typically performed by using one or more design-software packages (Electronic Design Automation Tools). To guarantee the integrity and correctness of the design process, different EDA tools required to have an intricate interface between them.

One clear line of demarcation within chip-design tasks is the separation of front-end (logic) and back-end (physical) design activities. The front-end/back-end "barrier" represents an important transition point for the designer, due to the importance of communicating design intent from logical- to physical-design representations. A successful transition from front-end to back-end design is necessary for achieving timing closure – meeting the chip's timing constraints without expensive and time-consuming logic-synthesis/physical-implementation iterations.

Logic synthesis is a critical part of an EDA tool suite. Originally, it was developed in the mid 1980s. Logic synthesis tool converts the design from RTL representation, an architectural design view, to a gate level representation, a structural design view. Apart from converting the design to a different abstraction level, logic synthesis tool also optimizes the design according to the user constraints, e.g. timing or area specifications.

In the logic synthesis process, the user identify a library of gate level logic cells the synthesis tool would target, along with timing or area constraints. The synthesis tool will convert the RTL level representation to a gate level representation which satisfy the given timing/area constraints. The gate level design will only contain gates given in the initially specified library.

The timing of the different paths in the chip is calculated by adding the delays of the gate and the interconnect delays. As at the gate level the routing information is not available, the synthesis tool would use estimates of interconnect delay. Since these delays are only estimates, the actual delay of the net obtained after place and route step almost always comes to be much different than the estimated delays. This poor estimation at logic synthesis level results in the non-compliance of the chip performance to the timing specification [6].

Non-compliance of chip performance to the timing specification required the designer to modify the design at the RT level, re-synthesize, and then redo the chip's layout– expensive operations that took huge chunks of time. Adding to the design problem was the reality that one often needed more than one synthesis/P&R reiteration to meet timing specification–in other words, to reach timing closure. With shrinking time-to-market cycles and profit margins, the need for accurate timing analysis become more relevant.

For almost two decades, conventional static timing has proved to be a reliable and efficient method for timing sign-off of digital integrated circuits. Incremental static timing - fast computation of circuit timing when only a small portion of the design is changed - is a key enabler of circuit optimization during logic synthesis and physical synthesis. In recent years, the static timing paradigm has been enhanced to accommodate such deep sub-micron effects as coupling noise, RC and RLC interconnect models, simultaneous switching and more accurate waveform propagation.

There are three main reasons why the conventional paradigm of static timing analysis is breaking down [7].

1. The first is that critical dimensions are scaling faster than our control of them. Thus, the variability of physical dimensions, such as the effective length of a transistor channel, is proportionately increasing [2].

2. In previous technologies, variability was dominated by the active transistors and gates. It was reasonable to assume that the dominant sources of variation were strongly correlated, and therefore case analysis with relatively few process comers provided high coverage confidence. With recent technology generations, interconnect metalization has shown large variability, too. These sources of variability are relatively uncorrelated to the former, and relatively uncorrelated from one metal

level to another, so the number of significant and independent sources of variation has dramatically increased. Concomitantly, the number of cases or comers required for confident coverage has grown tremendously (exponentially).

3. Across-the-chip Linewidth Variation (ACLV), caused mainly by reticle and proximity effects during lithography [8] and by local density effects, is increasing with each new generation of technology. Of course, temperature and power supply gradients can also be significant across regions of the chip.

The secondary effects like above are becoming more and more pronounced with the decreasing device size. Some of these effects were present in the old technologies also. But now their number and magnitude is increasing. In the old technologies, the timing analysis was performed by analyzing the design on corner cases. With the increasing number of secondary effects, the corner case analysis becomes impractical. Apart from timing analysis getting too burdensome to perform, static timing analysis is also becoming pessimistic and risky at the same time. It is pessimistic due to the bounding nature of the methods involved for the timing analysis (worst casing). And it is risky because it is impossible to perform a timing analysis for all possible set of corners and cases.

The solution to the above described problem is statistical static timing analysis, which is the main topic of our work. Statistical static timing analysis will enable highly accurate timing analysis which will allow us to achieve timing closure at a faster rate and with the less synthesis-P&R iteration. It will also allow us to perform quantitative risk management as statistical timing analysis will provide the probabilistic distribution of the circuit delays. Overall statistical timing analysis will reduce pessimism, improve timing verification turnaround time and will help in increasing parametric yield.

In this work, we try to address the fabrication variability problem in DSM VLSI

Parameters	Nominal Values					3 σ Values				
Years	1997	1999	2002	2005	2006	1997	1999	2002	2005	2006
L_{eff}[nm]	250	180	130	100	70	80	50	60	40	33
T_{ox}[nm]	5	4.5	4	3.5	3	0.4	0.36	0.39	0.42	0.48
V_{dd} [V]	2.5	1.8	1.5	1.2	0.9	0.25	0.18	0.15	0.12	0.09
V_{th} [mV]	500	450	400	350	300	50	45	40	40	40
W [μm]	0.8	0.55	0.5	0.4	0.3	0.2	0.017	0.14	0.12	0.1
H [μm]	1.2	1.0	0.9	0.8	0.7	0.3	0.3	0.27	0.27	0.25
ρ [$\frac{m\Omega}{\Box}$]	45	50	55	60	75	10	12	15	19	25

Table 1.1: CMOS Technology Roadmap

design automation. We will look at the problem of timing analysis under fabrication variability in detail. Timing analysis is one of the most critical tools in design optimization. Under fabrication variability, it becomes hard to accurately estimate the timing of the circuit. We will present techniques for performing timing analysis in presence of fabrication variability.

1.0.2 Fabrication Variability: Sources and Issues

When technology scales down to nanometer, uncontrollable parameter variations will significantly affect the circuit timing performance. As shown in table 1.1, the magnitude of parameter variation cannot scale down as fast as the nominal values, so parameter variation, as a percentage of nominal value, becomes larger and larger.

Circuit timing parameters are not independent to each other. They are, instead, correlated with each other. For example, variations in gate capacitances are not independent

to variations in gate length. Similarly, variations in threshold voltage is not independent to that in the gate length either.

There are two types of orthogonal fabrication variabilities:

Die-to-Die Variations Die-to-die variations, also known as global or systematic variations, represent the parameter variations from chip to chip for the same circuit. These are variations which are imposed on the design by the fabrication process. Within in the same die, there is no variability in the parameters. Such cases can be analyzed using the classic Monte-Carlo or worst case techniques. The fabrication process can characterize the variability as a distribution for such an analysis.

Intra-Die Variations Intra-Die variation represent the parameter variation within the same die but different locations. Due to the way circuits are manufactured, these intra-die variations in different chip locations are also dependent to each other.

The presence of large number of varying parameters makes any form of circuit analysis harder. The variations can be both spatially correlated or independent (random variations). These variations can be in device parameters or in interconnect parameters.

Some trends in the interconnect and device parameter trends are given in the SIA technology roadmap [9]. It is interesting to note that both device and interconnect variability are causing significant variability variations in delay.

The authors in [10] performed an experiment on the above device parameters to study the variations induced by fabrication variability on the design. We note that though the device variations somewhat stabilizes with newer technology nodes, the interconnect variations keep increasing. This fact clearly points out the importance of handling of both interconnect variations and device variations in all VLSI design optimization. The

current state of the art technology are not able to predict these variability with reasonable accuracy. In [7], the author points out:

1. Technology scaling is continuously reducing the physical dimension and the effect of variabilities in such geometries is making the current estimating models very inaccurate.

2. It was assumed earlier that the variations in device parameters are strongly correlated, thereby reducing the number of sources of variability. However, DSM technologies shows the large number of sources of variabilities which are correlated as well as independent. As a result, the number of sources of variabilities have increases making any kind of analysis harder.

Chapter 2

Introduction to Statistical Timing Analysis Engine

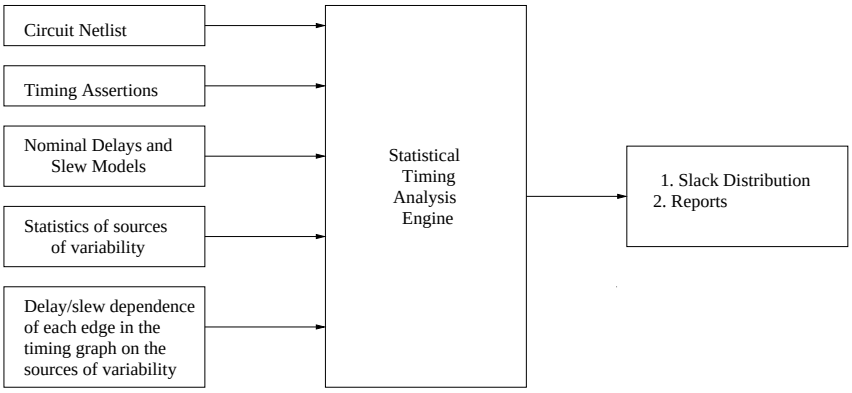

Figure 2.1: Block Diagram of a Statistical Timing Analysis Engine

Conventional static timing analysis program takes circuit as input and constructs a timing graph. The delays and transition times (slew) characteristics of each gate are provided by either of the two ways:

1. By providing Delay models

2. By computing it on fly by transistor level time-domain simulation.

The output of the static timing analysis is the slack at each primary output. The program can also output a timing report which can detail longest (critical) paths, slack at each gate, arrival times etc. In contrast, statistical static timing analysis(2) engine also takes information regarding sources of variations. By modeling the uncertainty in the manufacturing process, the statistics of the sources of variations are produced as result. The

output of the program is the probability distribution of the slack.

A sample output of the statistical timing analysis program is shown in 2. The distribution in this example shows that the parametric yield of the circuit is almost 100% for the slack of -600ps. Whereas the parametric yield drops to almost 0% for the slack of 40ps.

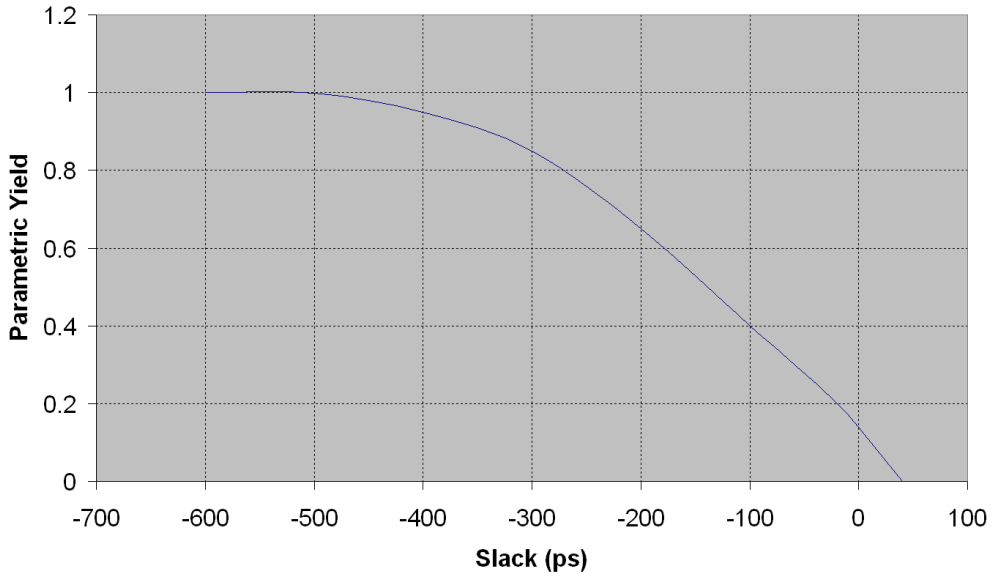

Figure 2.2: Sample Slack Distribution

Having slack distribution information as in figure 2.2 has many benefits:

1. In the case of binned microprocessor products, it allows the prediction of the percentage of chips that will fall in the high-speed high-profit bin.

2. In the case of ASICs, it allows for early decision making on risk management at chip and board level.

Statistical timing analysis engine can work in two different types:

Block Based Statistical Timing Analysis Propagation of arrival time and required time through a timing graph is known as Block based timing analysis. Since block based timing analysis works by performing breath first search on the gates of the circuit, the complexity of this type of timing analysis engine is linear in the size of the gates in the circuit. These types of engine lacks in capturing correlations, e.g. between clock and a data path.

Path Based Statistical Timing Analysis Path based algorithms perform the timing analysis for each path individually. Thus this type of engine can capture correlations better than the block based schemes. As the number of paths in the circuit can be exponential in the number of gates present in the circuit or in the number of inputs in the circuit, special techniques must be employed to make this type of engine tractable.

Regardless of a block or path basis, there are two main numerical methods employed by statistical timers.

2.0.3 Performance-space methods

In this method [11, 12, 13], the problem of timing analysis is seen as finding the probability of the longest of all the paths in the circuit having a delay value q, for all values of q. On a path basis, this problem can be thought of as the integration of an N-dimensional JPDF of path delays, where N is the number of paths in the circuit. Since integration in N-dimensional space will be computationally expensive and might be intractable for all practical purposes, the methods based on this technique do not integrate functions in a N-dimensional path. Instead they use block based methods to propagate the delay quantities across the network.

2.0.4 Parameter-space methods

In this method [14, 15, 16], the circuit requirements such as delay are modeled as constraint in the parameter space. This modeling is done in such a way that on one side of the constraint the circuit is feasible and on the other side infeasible. The intersection of all these constraints defines a feasible region of operation for the circuit. To reduce the dimensionality of the sources of variation, mathematical techniques, e.g. Principal component analysis or SVD-based reduction [17] can be applied. The application of these techniques generally inject very small loss in accuracy.

2.1 Desirable attributes of a statistical timer

This section explains the required attributes of a practical statistical timer.

2.1.1 Correlations

Statistical timing engine should have capability to consider the correlations. This attribute is must for accurate sign-off purposes. A simple example will demonstrate this point. Suppose the chip has 50,000 latches, each with a setup and hold test. Assume that each of the 100,000 tests has a 99.99% probability of passing. If all the tests are perfectly correlated, the parametric yield of the circuit is 99.99%. If the tests are independent, the yield is 0.005%! The real value of yield will be in between these two extremes (99.99% and 0.005%).

There are many kinds of correlations which need to be considered.

1. The first type of correlation is due to re-convergent fan-outs. This type of correlation is produced due to data paths sharing some gates along the way.

2. The second type of correlation arises due to commonality between data and clock.

3. The third and most important source of correlation is dependence on global parameters. As chip-to-chip, wafer-to-wafer and lot-to-lot types of variability is not seen across a single chip, the delay and slew of almost every edge in the timing graph is correlated with every other edge's delay and slew.

4. It is impossible for two adjacent gates to have best and worst case characteristics with respect temperature gradients, power supply gradients and ACLV due to the proximity. Any timer that allow even such variations for greater accuracy is being needlessly pessimistic.

2.1.2 Bounded vs. statistical analysis

Statistical timing analysis engine should be flexible enough to allow specification of analysis of each source of variation as bounded or statistical manner. This will help in a situation where the amount of variation is very small for a variation. That variation can be treated in a bounding manner. The net result of this approach will be reduce dimensionality of the statistical timing analysis.

2.1.3 Slew and load dependence

Delay of each connection/edge in the timing graph is a function of output load and input slew. Input slew and output load also depend on the process variation. The effect of these secondary nature should be taken care of for accurate sign-off. Delay is non-linearly dependent on the input slew and output load. This non-linear dependency of delay over input slew and output load makes the statistical timing analysis more complex. As we must propagate a probabilistic slew at each timing point (gate).

2.1.4 Intra-die variation

A statistical timer should be able to handle intra-die variation. [18] provides a mechanism of modeling the across the chip line variation for the algorithms having a small number of independent sources of variability.

2.1.5 Tail of the slack distribution

The output of a statistical timing engine is a distribution of the predicted slack. The tail of this distribution might be of significant importance based on the application for which circuit will be used. For example, the sign-off criterion on an ASIC may be the -3σ clock frequency. Thus the variance prediction of any algorithm that is used must be very good. Also to have good confidence in such prediction, the accuracy of the modeling and correlations must be very accurate.

Chapter 3

Introduction

Growing sources of variations due to fabrication has caused an increased interest in statistical timing analysis (SSTA). The central idea in SSTA is to capture the variability by modeling delays as distributions and performing timing analysis statistically on these distributions while capturing possible correlations that could exist between gate delays. A lot of recent work in statistical timing analysis tries to consider the process and environmental variabilities in performance analysis. The authors in [19] propose a canonical first order approximate delay model that takes into account both the correlated and independent randomness. A similar strategy is presented in [20]. A moment based approach for capturing correlations is presented in [21]. Further developments in this area try to improve the accuracy of SSTA by extending them to consider the non-linear dependence of gate delay on global sources of variability and the non-Gaussian nature of the variabilities themselves [4], [22], [23] . Most of these approaches model the gate delay as a nonlinear function of global principle components (this helps in capturing correlations). These principle components can have any probability density function and not necessarily Gaussian.

These generalized approaches are an improvement over the traditional approaches that assumed a linear variability model for gate delay, arrival times and Gaussian nature of the variability [19], [20]. But, the disadvantage lies in the fact that most of these approaches need some degree of numerical sampling which can be extremely slow. For example, [22] uses numerical techniques for computing the tightness probabilities which

are an integral part of computing the max of arrival times (please refer to [22] for details). Similarly, the approach in [4] uses numerical sampling to fit a polynomial of arbitrary degree on the arrival time of a gate. Moreover, the incremental aspects of non-linear non-Gaussian statistical timing analysis have not been investigated in detail.

In this thesis, we propose a non-linear non-Gaussian parametric statistical timing analysis engine which overcomes the shortcomings described above. Just like [4], we model the gate delay and arrival times as polynomials of arbitrary degree (depending on the choice of the designer) with global independent principle components as variables (for capturing correlations). Like other block based approaches, we propagate timing information topologically from primary inputs to outputs. At each node we calculate the arrival time of the gate and approximate it using a polynomial in global principle components. This is done by computing a MAX of the input arrival times (each of which are polynomials), approximating the MAX as a polynomial and adding the result with the gate delay itself (again polynomial). Unlike the existing approaches, our MAX operation does not need any numerical sampling and is purely analytical.

Our specific contributions are enumerated below.

1. We develop a generic way of performing the MAX operation on polynomials and of approximating the result back as a polynomial. To this end we present two approaches 1) algebraic 2) probabilistic. Neither of these approaches need numerical sampling. The probabilistic approach is more accurate but slower than the algebraic approach.

2. We develop the theory of incremental SSTA for non-linear non Gaussian case. To this end we propose 1) a method for fast and incrementally generating the new timing information for minor changes in the circuit graph 2) a technique for predicting

the current expected error in the timing estimate given a history of incremental SSTA iterations. This can be used to decide when the incremental SSTA should be executed and when the error is too much and accurate SSTA should be executed.

The techniques presented in this thesis are not limited to any specific degree of the polynomial approximation and are generic. From our experiments we found that our algebraic approach is on average 9588 times faster than Monte Carlo, whereas the numerical approach of [4] is only 31.3 times faster. Both the approaches had similar RMS error. Also, our proposed probabilistic approach results in much smaller errors when compared with the algebraic approach, but the gains in speed up is relatively lesser (average 81 times fast than Monte Carlo). On the average, Our incremental SSTA approach could generate the timing information 23 faster than the more accurate algebraic approach without a significant increase in error. We also propose a methodology for capturing the error injected in the incremental SSTA. This could be used to decide when accurate SSTA is needed and when incremental is sufficient. We found that our approach could accurately capture the trends in the accurate value of the error injected in incremental SSTA.

Chapter 4

Gate Delay and Circuit Modeling

We model the gate delay as a polynomial in independent process parameters, obtained through principal component analysis . This approach is similar to various existing approaches as in [20], [22] etc. If we represent independent process parameters as x,y,z, and w, then the delay of a gate i can be expressed as:

$$D_i = poly(x, y, z, w) \tag{4.1}$$

Where poly(x,y,z,w) means a polynomial in the given parameters. Although in this thesis we will assume all delay and timing characteristic to be of degree two, it should be clear from the explanation that our approach is not limited to degree two polynomials only and can be extended for higher degree polynomials.

We represent gate delay D_i for gate i as a general second order polynomial:

$$\begin{aligned} D_i = {} & c_1x + c_2y + c_3z + c_4w + c_5x^2 + c_6y^2 + c_7z^2 + c_8w^2 + c_9xy \\ & + c_{10}xz + c_{11}xw + c_{12}yz + c_{13}yw + c_{14}zw + c_{15} \end{aligned} \tag{4.2}$$

Each of x, y, z, and w will have an underlying probability density function which is not necessarily Gaussian. Therefore, we can model the non Gaussian aspect of SSTA effectively.

We model a digital circuit with a directed acyclic graph(DAG), G. Gates of the circuit are represented as nodes of the graph and the connections between gates are represented as edges between the nodes of the graph. A delay is associated with every gate.

Figure 1 shows a gate in the circuit with k fan-ins. The arrival time at fan-in i of a gate is represented as A_i. The delay of the gate is represented as D. As we represent gate delay as a polynomial, following the same strategy we would also like to represent the arrival time as a polynomial. Thus the arrival time and the delay of the gate i can be represented by the following equations:

$$A_i = poly(x, y, z, w) \quad \forall i \in fanin \tag{4.3}$$

A dummy source node and a dummy sink node, both having zero delays, are included

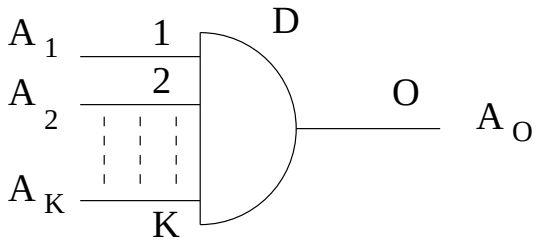

Figure 4.1: SUM and MAX Computation

in the DAG to transform the DAG into a single source- single sink DAG. The dummy source node is connected to all the primary input nodes, nodes having zero fan-ins. In a similar fashion, we connect all the primary output nodes, nodes having zero fan-outs, of the circuit to the dummy sink node. Polynomial modeling of gate/arrival times allows us to capture the non linear dependence of timing on variability

Chapter 5

SSTA Framework

In this section we will describe the algorithm for the calculation of the arrival time of the circuit.

The arrival time of the circuit is the arrival time at the sink node. It is obtained by topologically traversing the nodes of the graph and calculating arrival time at each node. The arrival time at a node is calculated by the following two steps:

1. For every fan-in of the node calculate the **SUM** of the arrival time at that fan-in and the node delay.

2. Find the **MAXIMUM** among all the arrival times computed in the first step.

Step 1 of the above algorithm contains a "SUM" operation and the step 2 consist a "MAXIMUM" operation. As we represent the gate delays and arrival times as polynomials, both of these operations should be defined for polynomials.

5.0.6 SUM operation

SUM operation computes the addition of two polynomials: arrival time at the fan-in of the current node and the node delay. The coefficients of the resultant polynomial are the sum of the corresponding coefficients in arrival time A_i and node delay D. For each fan-in i, the result of the SUM operation is denoted by A_{io}:

$$A_{io} = A_i + D \quad \forall i \in fanin \tag{5.1}$$

As this step does not include any approximations, it is free from any error.

5.0.7 MAXIMUM operation

The MAXIMUM operation finds the maximum of the set of K arrival time polynomials, obtained though the SUM operation (K is the number of fan-ins), and approximate it to a polynomial .

$$A_o = MAX(A_{1o}, A_{2o}, \ldots, A_{Ko}) \approx poly(x, y, z, w) \tag{5.2}$$

We compute the maximum of the K polynomials iteratively. It means that we first compute the maximum of A_{1o} and A_{2o} and then the maximum of the resultant polynomial and A_{3o} and so on. We do this till we find the maximum polynomial for the whole set. Thus, the problem of finding the maximum of K polynomials is reduced to finding the maximum of two polynomials. The MAX operation, as suggested in [4] could be polynomially approximated using regression. This regression was driven by numerical sampling and was therefore very slow. In the next few subsections, we outline our approach for MAX approximation which does not uses any numerical sampling.

Let's suppose there are 2 polynomials A and B whose maximum we would like to find. Then, the maximum of A and B can be represented as:

$$MAX(A, B) = \frac{A + B + |A - B|}{2} \tag{5.3}$$

We can easily get the sum of A and B polynomials by SUM operation. This implies that to obtain the maximum of A and B as a polynomial, we only have to represent $|A - B|$ as a polynomial.

We propose two approaches to approximate $|A - B|$: Algebraic Approximation and Probabilistic Approximation.

Algebraic Approximation

The approximation of $|A-B|$ as a polynomial is done in following three steps. Each step of the algorithm will be explained later in detail.

1. We calculate the range of polynomial $A - B$. Note that since A and B are both polynomials, $A - B$ will also be polynomial. We define "range" of a polynomial as the maximum and minimum values of the polynomial over the entire range of its variables. Based on the range of the polynomial $A - B$, there can be 3 cases:

   (a) The range of the polynomial $A - B$ is positive. In this case, $|A - B|$ simply becomes $A - B$. Thus, no computation is involved for this case.

   (b) The range of the polynomial $A - B$ is negative. In this case, $|A - B|$ simply becomes $B - A$. For case (a) and (b), the algorithm to compute $|A - B|$ as polynomial ends here. As case (a) and case (b) do not make any approximations and do not involve any computation, they do not introduce any error in the scheme and result in high speed up of the algorithm.

   (c) In this case, the maximum value of the polynomial $A - B$ is greater than zero and the minimum value of the polynomial is less than zero. For this case, we perform step 2 and 3.

2. Let's represent $A - B$ by P. As we know the range of P from step 1, we approximate $|P|$ by a higher degree polynomial in "P" using regression. As it would be clear, this step does not require any numerical sampling.

3. Since P (or, A-B) is a polynomial in $x, y, \ldots, w$; a higher degree polynomial in P will also be a polynomial in $x, y, \ldots, w$. At this step, we use regression to approximate higher degree polynomial obtained in the step 2 to a quadratic polynomial in

$x, y, \ldots, w$. Once again, no sampling is required at this step. **Also it should be noted that the use of quadratic polynomial as the final fit for the arrival times is purely our choice and any degree polynomial can be used instead.**

We would like to point out that direct approximation of the mod function into a quadratic is not feasible without numerical sampling. That is exactly why we have an intermediate step (step 2) so that numerical sampling could be avoided.

Now we describe all the three steps of the algorithm in detail:

Step 1: Range Calculation

Given the range of input parameters x, y, z, w, we calculate the range (maximum and minimum value) of the polynomial $A-B$ by evaluating the partial derivative of $A-B$ with respect to x, y, z and w respectively and equating them to zero.

$$\frac{\partial(A-B)}{\partial x} = f_x(x,y,z,w) \;=\; 0 \tag{5.4}$$

$$\frac{\partial(A-B)}{\partial y} = f_y(x,y,z,w) \;=\; 0 \tag{5.5}$$

$$\frac{\partial(A-B)}{\partial z} = f_z(x,y,z,w) \;=\; 0 \tag{5.6}$$

$$\frac{\partial(A-B)}{\partial w} = f_w(x,y,z,w) \;=\; 0 \tag{5.7}$$

Solving the equations 5.4,5.5, 5.6, and 5.7 for x,y,z and w gives a point for maxima or minima of the polynomial $(A-B)$. We compute the value of the polynomials at this point as well as at the boundary surfaces over the range of input parameters . As a result,

we get the range of the polynomial $A-B$. We would like to stress that if we get the range of the polynomial to be either negative or positive entirely, then the algorithm to compute $|A-B|$ will stop at this step.

Step 2: Conversion to Higher Degree Polynomial

Now we will explain the step 2 of the algorithm for finding the $|A-B|$. In this step, we convert $|A-B|$ to a higher degree polynomial by reducing the Root Mean Square (RMS) error of approximating the $|A-B|$ to a polynomial.

We approximate $|A-B|$ by an approximating polynomial, Z, of degree N in P(or, $A-B$) :

$$Z = (q_0 + q_1 P + \ldots + q_N P^N) \approx |P| \tag{5.8}$$

Where $q_0, q_1, \ldots q_N$ are coefficients of the degree N approximating polynomial.

If we represent RMS error of the approximation as R and the range of the polynomial P as r1 and r2. As we know the range of the polynomial P, we can integrate the RMS error in this region instead of performing sampling to achieve the same. Then:

$$R^2 = \int_{r1}^{r2} (|P| - (q_0 + q_1 P + \ldots + q_N P^N))^2 \, dP \tag{5.9}$$

In order to minimize the RMS error of approximation, we evaluate the partial derivative of R^2 w.r.t. each coefficient in the approximating polynomial and equate it to zero as shown below:

$$\begin{aligned}
\frac{\partial(R^2)}{\partial q_0} &= -2\int_{r1}^{r2} (|P| - (q_0 + q_1 P + \ldots + q_N P^N))dP = 0 \\
\frac{\partial(R^2)}{\partial q_1} &= -2\int_{r1}^{r2} (|P| - (q_0 + q_1 P + \ldots + q_N P^N))PdP = 0 \\
&\vdots \\
\frac{\partial(R^2)}{\partial q_N} &= -2\int_{r1}^{r2} (|P| - (q_0 + q_1 P + \ldots + q_N P^N))P^N dP = 0
\end{aligned}$$

Rearranging these equations and writing them in the form of matrix equations as:

$$\begin{pmatrix} \int 1dP & \int PdP & \dots & \int P^N dP \\ \int PdP & \int P^2 dP & \dots & \int P^{N+1} dP \\ \vdots & \vdots & \vdots & \vdots \\ \int P^N dP & \int P^{N+1} dP & \dots & \int P^{2N} dP \end{pmatrix} \times \begin{pmatrix} q_0 \\ q_1 \\ \vdots \\ q_N \end{pmatrix} = \begin{pmatrix} \int |P| dP \\ \int P|P| dP \\ \vdots \\ \int P^N |P| dP \end{pmatrix} \tag{5.10}$$

Thus, we get a matrix equation in the form $XQ = Y$ where we would like to get the value of Q matrix. Note that the each term in the X matrix is of the form:

$$\int_{r1}^{r2} P^i \, dP = \left(\frac{P^{i+1}}{i+1} \right)_{r1}^{r2} \tag{5.11}$$

Therefore, X matrix can be calculated without any numerical sampling unlike [4]. The elements of matrix Y contains the mod function, but because we know the range of P (or, $A - B$), we can calculate this matrix easily. For example, the integration of the last element of the matrix Y can be computed as:

$$\int_{r1}^{r2} P^N |P| \, dP = \int_{r1}^{0} P^N (-P) \, dP + \int_{0}^{r2} P^N (P) \, dP \tag{5.12}$$

$$= -\int_{r1}^{0} P^{N+1} \, dP + \int_{0}^{r2} P^{N+1} \, dP \tag{5.13}$$

Since calculation of matrix X and matrix Y requires no numerical sampling, the matrix Q can be calculated without numerical sampling. Our approach is entirely different from the approach followed in [4] where a sampling based approach was implemented to

get the system of matrix equation for finding the maximum of two polynomials. Our approach is purely analytical and thus performs fast computation of MAX(A,B).

Increasing the degree, N, of approximating polynomial results in greater accuracy but it affects the runtime. For example, for the degree 2 (degree 2 in P, degree 4 in x,y,z,w) approximating polynomial Z, the number of elements in X matrix are $9(3 \times 3)$ whereas for degree 3 approximating polynomial the number of elements are $16(\ 4 \times 4)$. Thus, the complexity of this step increases quadratically with the degree of the approximating polynomial, but at the same time the quality of the solution improves. In our experiment, we found $N = 3$ to be an ideal value. Therefore, the polynomial approximation of $|P|$, Z, is of degree 3 in P and is of degree 6 in $x, y, \ldots, w$, assuming A and B both are order 2 polynomials.

At this step we have a higher degree approximating polynomial which represents $|A - B|$.

Step 3:Converting higher degree polynomial to degree two polynomial

After getting a higher degree polynomial representing $|A - B|$ in step 2, the next step of the algorithm is to convert it to degree two polynomial, so that the polynomial approximation of $|A-B|$ remains quadratic. This conversion is done by reducing the Root Mean Square (RMS) error of approximating the higher degree polynomial to a degree two polynomial. This approach is similar to the approach followed in the step 2.

Though in this thesis we have assumed the degree of the approximating polynomial as two, note that the designer performing SSTA can select different degrees of approximating polynomial (while following the same methodology).

Suppose that the second degree polynomial approximating $|A - B|$ is C, then:

$$C = ax + by + cz + dw + ex^2 + fy^2 + gz^2 + hw^2 + ixy + jxz + kxw + lyz + myw + nzw + o \quad (5.14)$$

So if we represent the RMS error of representing the higher degree polynomial Z (refer equation 5.8) to a second degree polynomial C by R_1 then,

$$R_1^2 = \int (Z - C)^2 \, dx \, dy \, dz \, dw \quad (5.15)$$

Here $\int$ represents $\iiint\int$. Hereafter, the same convention has been followed in the thesis whereas necessary and the context should be clear by the number of principal components ,e.g. dx, present in the equation.

We partially differentiate R_1^2 w.r.t. $a, b, \ldots, o$ and equate it to zero for getting the minimum RMS error:

$$\frac{\partial(R_1^2)}{\partial a} = -2 \int (Z - C)x \, dx \, dy \, dz \, dw = 0 \quad (5.16)$$

$$\frac{\partial(R_1^2)}{\partial b} = -2 \int (Z - C)y \, dx \, dy \, dz \, dw = 0 \quad (5.17)$$

$$\vdots$$

$$\frac{\partial(R_1^2)}{\partial o} = -2 \int (Z - C) \, dx \, dy \, dz \, dw = 0 \quad (5.18)$$

Rearranging these equation and substituting the value of C from equation 5.14 gives:

$$\int Zx \, dx \, dy \, dw \, dz = \int ax^2 \, dx \, dy \, dz \, dw + \int bxy \, dx \, dy \, dz \, dw + \ldots + \int ox \, dx \, dy \, dz \, dw \quad (5.19)$$

$$\int Zy\,dx\,dy\,dw\,dz = \int axy\,dx\,dy\,dz\,dw + \int by^2\,dx\,dy\,dz\,dw + \ldots + \int oy\,dx\,dy\,dz\,dw \quad (5.20)$$

$$\vdots$$

$$\int Z\,dx\,dy\,dw\,dz = \int ax\,dx\,dy\,dz\,dw + \int by\,dx\,dy\,dz\,dw + \ldots + \int o\,dx\,dy\,dz\,dw \quad (5.21)$$

We express these equations as a system of Matrix equations:

$$\begin{pmatrix} \int x^2\,dx\,dy\,dz\,dw & \int xy\,dx\,dy\,dz\,dw & \ldots & \int x\,dx\,dy\,dz\,dw \\ \int xy\,dx\,dy\,dz\,dw & \int y^2\,dx\,dy\,dz\,dw & \ldots & \int y\,dx\,dy\,dz\,dw \\ \vdots & \vdots & \vdots & \vdots \\ \int x\,dx\,dy\,dz\,dw & \int y\,dx\,dy\,dz\,dw & \ldots & \int 1\,dx\,dy\,dz\,dw \end{pmatrix} \times \begin{pmatrix} a \\ b \\ \vdots \\ o \end{pmatrix} = \begin{pmatrix} \int Zx\,dx\,dy\,dz\,dw \\ \int Zy\,dx\,dy\,dz\,dw \\ \vdots \\ \int Z\,dx\,dy\,dz\,dw \end{pmatrix} \quad (5.22)$$

Thus we get a system of matrix equations in the form $XQ = Y$. Again, note that the X and Y matrix can be calculated without using sampling, as shown in step 2. For example, in the X matrix, the value $\int xydxdydzdw$ can be calculated easily since we know the range of the parameters x,y,z,w and they are independent. Similarly in the Y matrix, the value $\int Zxdxdydzdw$, would also be computed analytically since Z is a high order polynomial. Therefore without using any sampling, the regression operation could be

performed. Replacing value of the polynomial C in the equation 5.3 for $|A - B|$, we can get MAX(A,B).

Probabilistic Approximation

In algebraic approach, we have not considered any probability density function for the input variable x, y, z,. . . , w. This may result in significant approximation error if the probability distribution functions of the variables are not uniform. In the probabilistic approach to approximate the $|A - B|$ as polynomial we consider the probability density functions of the variables x, y, z,. . . , w. The basic idea is to perform polynomial fitting in such a way that the error at points which are more probable is lesser compared with improbable points, thereby reducing the overall error.

The probabilistic approximation approach is very similar to the algebraic approximation approach and contains the same 3 steps. We can use the probability density function of the input variables in the second and third steps of the algorithm to improve the quality of the result.

PDF Consideration in Second Step: To include the probability density consideration in the second step of the algorithm, we modify the equation 5.9 to compute $|P| \equiv |A - B|$ as:

$$R^2 = \int_{r1}^{r2} (|P| - (q_0 + q_1 P + \ldots + q_N P^N))^2 \, Pr(P) \, dP \tag{5.23}$$

Where Pr(P) is the probability density function of P, or A-B. Though we know the probability density function of the input variables $x, y, z, \ldots, w$, we don't know the probability density function of P. So in this step, we ignore the probability densities. Thus our algorithm remain same for the second step as compared to the algebraic approach. At the end of the second step we get a higher degree polynomial Z, approximating $|A - B|$.

PDF Consideration in Third Step For the third step of the algorithm, let's assume that the probability density functions of variables x, y, z,. . . ,w are Pr_x, Pr_y, Pr_z and Pr_w. If we represent the RMS error of representing the higher degree polynomial Z to a second degree polynomial C, represented by equation 5.14, by R_2, then,

$$R_2^2 = \int [(Z - C)^2 Pr(x)Pr(y)Pr(z)Pr(w)]\, dx\, dy\, dz\, dw \tag{5.24}$$

Similar to algebraic approach, we partially differentiate R_2^2 w.r.t. to the coefficients of C, i.e a,b,. . .,o respectively, and equate it to zero to get the system of matrix equations:

$$\begin{pmatrix} \int x^2 Pr_x Pr_y Pr_z Pr_w & \dots & \int x Pr_x Pr_y Pr_z Pr_w \\ \int xy Pr_x Pr_y Pr_z Pr_w & \dots & \int y Pr_x Pr_y Pr_z Pr_w \\ \vdots & \vdots & \vdots \\ \int x Pr_x Pr_y Pr_z Pr_w & \dots & \int Pr_x Pr_y Pr_z Pr_w \end{pmatrix} \times \begin{pmatrix} a \\ b \\ \vdots \\ c \end{pmatrix} = \begin{pmatrix} \int Zx Pr_x Pr_y Pr_z Pr_w \\ \int Zy Pr_x Pr_y Pr_z Pr_w \\ \vdots \\ \int Z Pr_x Pr_y Pr_z Pr_w \end{pmatrix} \tag{5.25}$$

Where $\int$ represent $\iiiint$. The principal components $dx, dy, dz, and\, dw$ has been removed for brevity. Once again we get a matrix system of the form XQ=Y. The elements of the X matrix can be calculated as illustrated. $\iiiint x^a y^b z^c w^d Pr_x Pr_y Pr_z Pr_w = \int x^a Pr_x dx \int y^b Pr_y dy \int z^c Pr_z dz \int w^d Pr_w dw$. Knowing all the moments of the random variables x,y,z,w, we can very easily calculate this value. This does not need any sampling. A similar argument would hold for the Y matrix as well. But for some arbitrary PDF for $x, y, z, \dots, and w$ sampling may become necessary. In that case we would suggest to follow the algebraic approach.

Replacing value of the polynomial C in the equation 5.3 for $|A - B|$, we can get MAX(A,B).

Again the probabilistic approximation approach does not use any sampling for calculating the polynomial approximation of $|A-B|$. The analytical evaluation of the arrival time polynomial is a significant contribution of our approach.

5.0.8 Considering Local Randomness

The previous discussion assumed that the gate delay polynomials do not have any local randomness. Latest work on SSTA assumes that such an uncorrelated random component exists and can be represented as follows

$$D_i = poly(x, y, z, w) + R_i r \tag{5.26}$$

where r is an uncorrelated random component (typically modeled as a standard normal variable) and R_i the corresponding coefficient (variance). The approach of considering this component is similar to other existing approaches. In the SUM operation

$$D_1 + D_2 = poly_1(x, y, z, w) + R_1 r_1 + poly_2(x, y, z, w) R_2 r_2 \tag{5.27}$$

which can be rewritten as

$$D_1 + D_2 = poly(x, y, z, w) + \sqrt{(}R_1^2 + R_1^2) r \tag{5.28}$$

where poly is simply the polynomial obtained by the usual SUM operation. The new random component is simply another standard normal variable with exactly the same variance $\sqrt{(}R_1^2 + R_1^2)$ as the sum of $R_1 r_1$ and $R_2 r_2$. In the MAX operation, through regression we can get a polynomial approximation assuming r_1 and r_2 are additional

variables and represent the arrival time in the form of equation 5.27 which can then be further simplified using equation 5.28. Note that this is very similar to the approaches proposed by [19] to consider the local randomness and therefore we do not elaborate this more.

Chapter 6

Incremental SSTA

Physical synthesis programs often require the timing analysis tool to be called many times after one or more changes has been made to the circuit. To deal with the requirement of running the timing analysis many times for small changes in the circuit and providing accurate timing information at a considerable speed, we implemented a incremental timing analysis engine.

In this section, we investigate the incremental aspect of non-linear, non-Gaussian SSTA. As suggested in [19], any change in the gate delay need need to be reflected only in the fan-in/fanout cone of the design. We focused on incremental SSTA from the following perspectives:

1. Given a circuit, assume that we know the associated timing information of the circuit. Now let the delay of some gate changes, then how do we generate the timing information of the circuit, incrementally and as fast as possible.

2. Let us suppose we have made several changes in the circuit, each followed by an incremental SSTA. There will be a point at which there will be too much error accumulated in the timing information of the circuit. At this stage we would like to redo the entire timing information accurately. To this end, we would like to predict when the accumulated error is above a user specified threshold at which point the timing information needs to be redone.

6.0.9 Incremental Timing analysis algorithm

Our incremental timing analysis approach is based on the same general SSTA framework as described in section 5. But as the incremental timing analysis refers to the situations in which only few gate delays have been changed, it is not necessary to perform the SSTA from the source node to the sink node. To this end we start with the gate that is earliest in the topological-ordering whose delay has changed and propagate forward. Similar approaches has been followed in [19] but for linear and Gaussian approximation.

Let's assume that we have a gate whose fan-ins have arrival time A and B. Now assume that the arrival times at the fan-ins are changed to $A + \delta A$ and $B + \delta B$, where δA and δB are change in the arrival times and are also polynomials. So, the current arrival time at the output of the gate, C can be written as:

$$C = MAX(A + \delta A, B + \delta B) \tag{6.1}$$

Using the equation 5.3 for representing the maximum of two polynomial, we can write:

$$\begin{aligned} C &= \frac{A + \delta A + B + \delta B + |A + \delta A - B - \delta B|}{2} \\ &= MAX(A, B) + \frac{\delta A + \delta B}{2} \\ &\quad + \frac{|A + \delta A - B - \delta B| - |A - B|}{2} \end{aligned} \tag{6.2}$$

Now we approximate $|A + \delta A - B - \delta B|$ into a higher order polynomial (as step two), we already have a higher order polynomial for $|A - B|$ which we computed already in previous iteration. Therefore $|A + \delta A - B - \delta B| - |A - B|$ can be represented as a higher order polynomial. At this point instead of performing quad-regression (step3) we perform

a linear regression which is much faster. Note that MAX(A,B) and other components of the above equation are already known in polynomial form. Rest of the timing analysis approach remains the same.

6.0.10 Error Measurement

In order to avoid excessive accumulation of error after many runs of incremental SSTA, we would like to know how does the error accumulation increase from one run of incremental SSTA to another. This can be used to decide when accurate SSTA needs to be executed. To this end, we develop an analytical way of estimating the expected error due to incremental SSTA. Let us suppose we are doing the incremental MAX at a gate. Let the fan-in arrival times be $A + e_A$ and $B + e_B$ where e_A and e_B are the errors. These errors could be generated from incremental MAX of previous iterations or the current iteration. Assume that we know $E(e_A)$ and $E(e_B)$ (the expected values of the errors). Then, we would like to find out expected value of error in the arrival time of the pertinent gate ($E(P_e)$)

$$E(P_e) = E[MAX(A + e_A, B + e_B) - MAX(A, B)] \tag{6.3}$$

Where MAX(A,B) is the accurate value of the arrival time. Note that in the equation 6.3, the only known quantities are $A + e_A, E(e_A), A + e_B$,and $E(e_B)$. Therefore, in order to measure EP_e, we need to generate an estimate for A,B.

Let's suppose:

$$A_{appx} = A + e_A - E(e_A) \tag{6.4}$$

$$B_{appx} = B + e_B - E(e_B) \tag{6.5}$$

Substituting the values of A_{appx} and B_{appx} from 6.4 and 6.5 in equation 6.3, we get:

$$E(P_e) = E\left(\frac{e_A + e_B}{2} + \frac{|A + e_A - B - e_B| - |A_{appx} - B_{appx}|}{2}\right) \tag{6.6}$$

Till this point we have assumed the the approximation error in the MAX function is zero. To include this approximation error in the calculation, the equation 6.6 is modified as:

$$E(P_e) = E\left(\frac{e_A + e_B}{2} + \frac{|A + e_A - B - e_B| - |A_{appx} - B_{appx}| + \epsilon}{2}\right) \tag{6.7}$$

Where ϵ is the approximation error in the MAX operation due to incremental SSTA.

There can be following 4 cases:

1. $A + e_A - B - e_B > 0$ and $A_{appx} - B_{appx} > 0$
2. $A + e_A - B - e_B > 0$ and $A_{appx} - B_{appx} < 0$
3. $A + e_A - B - e_B < 0$ and $A_{appx} - B_{appx} < 0$
4. $A + e_A - B - e_B < 0$ and $A_{appx} - B_{appx} > 0$

As A_{appx} and B_{appx} are the approximation to A and B respectively, for the first case the equation 6.7 can be written as:

$$\begin{aligned} E(P_{e_1}) &= E\left(\frac{e_A + e_B}{2} + \frac{A + e_A - B - e_B - A_{appx} - B_{appx} + \epsilon}{2}\right) \\ &= E\left(e_A + \frac{\epsilon}{2}\right) \end{aligned} \tag{6.8}$$

Similarly, for the second, third and fourth cases: $E(P_{e_2}) = E\left(A_{appx} + e_A - B_{appx} + \frac{\epsilon}{2}\right), E(P_{e_3}) = E\left(e_B + \frac{\epsilon}{2}\right), E(P_{e_4}) = E\left(B_{appx} + e_B - A_{appx} + \frac{\epsilon}{2}\right)$.

Note that expected value of the sum of random numbers is same as the sum of the expected values (this is true even if the random numbers are correlated). Therefore knowing $E(A_{approx}), E(e_A), E(\epsilon)$ (calculation of $E(\epsilon)$ is elaborated subsequently) etc. the above expectations could be calculated easily. Now $A, A_{approx}, B, B_{approx}$ etc. are polynomials. Calculating the expected value of a polynomial in independent parameters with known densities is straightforward, and therefore not elaborated upon. Thus we can easily calculate the above expectations.

For calculating the $E(\epsilon)$, the approximation error of our MAX operation (due to linear regression in incremental mode instead of polynomial), we consider the three steps of our algorithm, i.e. range calculation, conversion to higher degree polynomial and conversion of higher degree polynomial to degree two polynomial. The first two steps of the algorithm are highly accurate. This means that the error introduced in the MAX operation is the error injected during the third step of the algorithm, i.e. the conversion of higher degree polynomial Z to a degree 1 polynomial C (in case of incremental mode). Thus the expected value of epsilon can be written as:

$$E(\epsilon) = E(Z - C) \tag{6.9}$$

Once again since Z-C is a polynomial, its expectation can be easily calculated.

Now, if we assume that the probability of case one, two, three and four are Pr_1, Pr_2, Pr_3, Pr_4 (note that these are joint probabilities). Then,

$$E(P_e) = E(P_{e_1}) * Pr_1 + E(P_{e_2}) * Pr_2 + E(P_{e_3}) * Pr_3 + E(P_{e_4}) * Pr_4 \tag{6.10}$$

Now, calculation of individual probabilities requires computing the joint probability of the form $Pr_1 = Probability(A + e_A - B - e_B > 0 \, and \, A_{appx} - B_{appx} > 0)$. Note that this is equivalent to computing the joint probability on two polynomials. A latest work in

[24] presents a moment matching based approach for calculating these probabilities (for brevity we omit the details). Note that in this calculation we have ignored the conditional expectation.

This completes the description of the methodology to estimate the expected error in the arrival time of a gate, given the expected error in the input arrival time and the inaccurate MAX operation. Now, while performing the incremental SSTA, we can propagate the expected error with each arrival time signal in topological order. Note that the SUM operation does not change the expected error since it is always accurate. After this propagation, we check the expected error at the sink node and decide we need to rerun the accurate SSTA or keep the current estimate from incremental SSTA. Note that this methodology considers the error injected due to both the current run of incremental SSTA and previous runs as well.

Chapter 7

Experimental Results

For experiments, we generated the gate delay polynomials as follows. The gate delay can be represented as a function of V_th as:

$$D_i \propto \frac{C_L \, V_{dd}}{(V_{dd} - V_{th})^{\alpha}} \tag{7.1}$$

Now, we assumed variability in threshold voltage V_{th} for each gate with a variance of 10%. This can be due to the gate length variation which is typically correlated and the dopant fluctuation which is typically uncorrelated. Just for the sake of getting data we assumed that the threshold voltage of a gate could be broken into 4 components

$$V_{th^i} = a_1 \, V_{th_1} + a_2 \, V_{th_2} + a_3 \, V_{th_3} + a_4 \, V_{th_4} + a_0 \tag{7.2}$$

Each of these basic components corresponds to the four corners of the chip. The coefficients are scaled such that 1) the overall variance is 10% and the individual values represent the distance of the gate from the four corners. Two gates that are placed closer to each other will have similar values of the coefficients and therefore their thresholds variability would be correlated. The mean threshold was 0.3V. To generate quadratic gate delay model w.r.t. the four components, we did Monte Carlo sampling to get a large number of data points followed by quad-regression. This gave us gate delay polynomials with the above-mentioned four components as variables. We would like to stress that this was just one way of generating spatially correlated data. Even [4] follows a similar approach. Our technique of course is generic and can be used as long as gate delays are polynomials in global principle components. We implemented our SSTA approach

Bench	M.C.	Our Alg			Sampling based Approach		
	Runtime	Runtime	Speedup	rms Error	Runtime	Sppedup	rms Error
C432	51091	7	7298	0.00435	2550	20	0.00183
C1355	161337	15	10755	0.00246	4744	34	0.00445
C1908	223158	12	18596	0.00436	4717	47	0.00230
C499	167172	16	10448	0.00189	4773	35	0.00100
C880	133981	9	14886	0.83000	4326	30	0.83000
i1	5586	2	2793	0.00040	423	13	0.00100
i2	81245	11	7385	0.00150	2725	29	0.00270
i3	63521	14	4537	0.00140	1616	39	0.00180
Average			9588	0.1070		31	0.1060

Table 7.1: Comparison between Algebraic and Sampling based approach

Benchmark	Monte Carlo	Probabilistic Approach			Algebraic Approach		
	Runtime	Runtime	Speedup	rms Error	Runtime	Sppedup	rms Error
C432	53566	636	84	0.0193	7	7652	0.0699
C1355	171829	2828	60	0.0471	15	11455	0.0732
C1908	238328	2362	101	0.0761	12	19860	0.2687
C499	169607	2549	66	0.0752	16	10600	0.0981
C880	139499	587	237	0.1084	9	15499	0.1084
i1	5646	181	31	0.0011	2	2823	0.1820
i2	80880	1772	45	0.0353	11	7352	0.0683
i3	62888	2767	22	0.0391	14	4492	0.7119
Average			81	0.0502		9967	0.2000

Table 7.2: Comparison between Probabilistic and Algebraic Approach

Bench	Algeb	Incremental		
	Runtime	Runtime	Speedup	rms Error
C432	266	47	5.65	0.0112
C1355	210	56	3.75	0.002
C1908	420	24	17.50	0.005
C499	546	80	6.82	0.003
C880	308	8	38.50	0.010
i1	56	1	56.00	0.001
i2	238	17	14.00	0.007
i3	266	7	38.00	0.012
Average			22.53	0.0064

Table 7.3: Comparison between Accurate and Incremental Approach

in sis [25]. We used an academic placement tool (CAPO [26]) to get a valid placement for each benchmark. The placement information was needed for setting the appropriate coefficients.

We experimented with the following cases 1) We compared our Algebraic approach (explained in section 5) with that of [4] that consists of a numerical approach and with Monte Carlo. In this experiment we assumed that the underlying parameters have a uniform PDF 2) We compared our Probabilistic approach assuming Gaussian probability distribution function for the underlying parameters with Monte Carlo and with algebraic approach 3) We performed the incremental SSTA and compared its error and runtime with that of our accurate SSTA approach.

Table 7.1 shows the experimental results for the comparison of our Algebraic approach with the numerical based max approximation in [4]. The runtime and error are compared with Monte Carlo results. Algebraic, Numerical MAX [4] and Monte Carlo essentially generate three timing CDFs (cumulative distribution functions). We report

the RMS errors between the three CDFs with Monte Carlo as reference. Both our and [4] used quadratic polynomials. It can be seen that our analytical approach is much faster than the current state of the art approach of [4] and the errors compared with Monte Carlo is very similar in both cases. It can be seen that our non-numerical approach gave a speed up of 9588 over Monte Carlo and [4] gate an average speed up of 31.30. Clearly our approach is extremely fast with very comparable RMS errors. It should also be noted that the speedup factor of our approach becomes higher for the larger circuits. This makes our approach of doing non linear non Gaussian SSTA is very accurate and practical.

Table 7.2 compares the probabilistic and algebraic approaches with Monte Carlo as reference, assuming Gaussian nature of the underlying parameters. It can be seen that although the speedups are lesser, the error of the probabilistic approach is smaller when compared with algebraic approach. However the speedup of probabilistic approach still remains significant. The average speedup of Probabilistic approach to Monte Carlo is 81.

Table 7.3 shows the experimental results for our approach for incremental SSTA. In this experiment, we compare our incremental approach with the Algebraic approach. We randomly changed the delay of five gates in the circuit and used the incremental approach to generate the new timing information and also used the Algebraic approach to do the same. In this table we report RMS error and the speed with algebraic approach as the reference. It can be seen that for small changes in the timing circuit, our incremental approach results in better runtime but little increase in error.

As explained in the section 6.0.10, after many changes in the timing graph, the error may become over-overwhelming and we may need to re-run accurate SSTA. We proposed an analytical methodology of fast calculation of the error accumulation due to incremental SSTA after many iterations. Figure 7.1 shows the quality of the proposed methodology.

Essentially, in each iteration we randomly changed the delays of about 5 gates and used incremental SSTA and the proposed method to calculate the expected error. We report the estimated error from one iteration to the other. We also report the accurate expected error calculated by compared the timing CDF generated by incremental approach and the algebraic approach. It can be seen that although, in an absolute sense the predicted and the accurate expected errors are not the same, the trends are still captured. Therefore, the proposed methodology could still be used to decide when accurate SSTA is needed and when incremental SSTA is good enough.

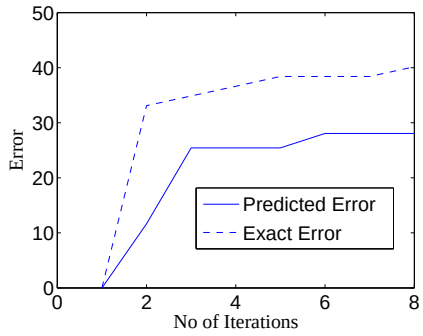

Figure 7.1: Predicted and Exact Error

7.1 Conclusion

In this work we proposed an efficient (non numerical), accurate methodology for performing non linear non Gaussian SSTA. We compared our methodology with the state of the art approaches and report massive gains in runtime with minimal impact on error. We also proposed an incremental technique for performing SSTA for non linear non Gaussian instances.

BIBLIOGRAPHY

[1] C. Visweswariah et al. "First-Order Parameterized Block-Based Statistical Timing Analysis". In *Procs of TAU*, 2004.

[2] Semiconductor Industry Association. "International Technology Roadmap for Semiconductor". 2001.

[3] Y. Taur et al. "25 nm CMOS Design Considerations". In *IEDM*, pages 789–792, 1998.

[4] V. Khandelwal and A. Srivastava. "A General Framework acurate Statistical Timing Analysis Considering Correlations". In *Proc. of DAC*, June 2005.

[5] Azadeh Davoodi. "Predictability-Driven Nano-Scale Design Automation". In *PhD Thesis*, 2006.

[6] V. Khandelwal. "Fabrication Variability Driven VLSI Design Automation for Deep Sub-micron/Nano Technologies". In *PhD Thesis Proposal*, 2005.

[7] C. Visweswariah. "Death, Taxes and Failing Chips". In *Proc. of Design Automation Conference*, June 2003.

[8] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu. "Impact of systematic spatial intra-chip gate length variability on performance of high-speed digital circuits". In *IEEE International Conference on Computer-Aided Design*, pages 62–67, November 2007.

[9] Semiconductor Industry Association. "International Technology Roadmap for Semiconductor". 1997.

[10] Sani R. Nassif. "Modeling and forcasting of manufacturing variations". In *In Proc. Asia South Pacific Design Automation*, pages 145–150, 2001.

[11] J.-J. Liou, K.-T. Cheng, S . Kundu, and A. Krstic. "Fast statistical timing analysis by probabilistic event propagation". In *Proc. Design Automation Conference*, pages 661–666, June 2001.

[12] A. Gattiker, S. Nassif, R. Dinakar, and C. Long. "Timing yield estimation from static timing analysis". In *Proc. IEEE International Symposium on Quality Electronic Design*, pages 437–442, 2001.

[13] M. Orshansky and K. Keutzer. "A general probabilistic framework for worst case timing analysis". In *Proc. Design Automation Conference*, pages 556–561, June 2002.

[14] 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 *In Proc. of Design Automation Conference*, pages 932–937, 2003.

[15] P. Feldmann and S . W. Director. "Integrated circuit quality optimization using surface integrals". In *IEEE Transactions on Computer-Aided Design of ICs and Systems*, pages 1868–1879, December 1993.

[16] J. M. Wojciechowski and J. Vlach. "Ellipsoidal method for design centering and yield estimation". In *IEEE Transactions on Computer-Aided Design of ICs and Systems*, pages 1570–1579, October 1993.

[17] Z. Li, X. Lu, and W. Shi. "An algorithm for process variation reduction based on SVD". In *Proc. IEEE Intemational Symposium on Circuits and Systems (ZSCAS)*, May 2003.

[18] A. B. Agarwal, D. Blaauw, V. Zolotov, S. Sundareswaran,M. Zhao, K. Gala, and R. Panda. "Path-based statistical timing analysis considering inter- and intra-die correlations". In *John Wiley and Sons)*, 1981.

[19] C. Visweswariah et al. "First-Order Incremental Block-Based Statistical Timing Analysis". In *Procs of DAC*, 2004.

[20] H. Chang and S. Sapatnekar. "Statistical Timing Analysis Considering Spatial Correlations Using a Single Pert-Like Traversal". In *Procs of ICCAD*, 2003.

[21] J. Le, X. Li and L. Pileggi. "STAC: Statistical Timing Analysis with Correlation". In *Procs of DAC*, 2004.

[22] H. Chang, V. Zolotov, S. Narayan and C. Visweswariah. "Parametric Block Based Statistical Timing Analysis With Non-Gaussian Parameters, Nonlinear Delay Functions". In *Proc. of DAC*, Jun 2005.

[23] L. Zheng, W. Chen, Y. Hu, J.A. Gubner, C. Chen. "Correlation Preserved Non-Gaussian Statistical Timing Analysis With Quadratic Timing Model". In *Proc. of DAC*, Jun 2005.

[24] A. Davoodi and A. Srivastava. "Efficient Stochastic Pruning for Variability-Driven Dual-Vth Leakage Optimization ". In *Workshop Notes IWLS*, Jun 2005.

[25] E.M. Sentovich, K.J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P.R. Stephan, R.K. Brayton, A.L. Sangiovanni-Vincentelli. *SIS: A System for Sequential Circuit Synthesis*. Memorandum No. UCB/ERL M92/41, Department of EECS. UC Berkeley, May 1992.

[26] A. Caldwell et al. "Can Recursive Bisection Alone Produce Routable Placements?". In *Proc. of DAC*, 2000.