Portland State University

PDXScholar

Dissertations and Theses

Dissertations and Theses

Summer 8-20-2018

Automating Variation and Repeater Analysis in Physical Design of Integrated Circuits

Subrat Mahalik
Portland State University

Follow this and additional works at: https://pdxscholar.library.pdx.edu/open_access_etds

Part of the [Electrical and Computer Engineering Commons](#)

Let us know how access to this document benefits you.

Recommended Citation

Mahalik, Subrat, "Automating Variation and Repeater Analysis in Physical Design of Integrated Circuits" (2018). *Dissertations and Theses*. Paper 4555.

<https://doi.org/10.15760/etd.6440>

This Thesis is brought to you for free and open access. It has been accepted for inclusion in Dissertations and Theses by an authorized administrator of PDXScholar. Please contact us if we can make this document more accessible: pdxscholar@pdx.edu.

Automating Variation and Repeater Analysis in Physical Design of Integrated Circuits

by

Subrat Mahalik

A thesis submitted in partial fulfillment of the
requirements for the degree of

Master of Science

in

Electrical and Computer Engineering

Thesis Committee:

Xiaoyu Song, Chair

Ataur R. Patwary

John M. Acken

Portland State University

2018

© 2018 Subrat Mahalik

Abstract

Rapid advancement and innovation in semiconductor research have continuously helped in designing efficient and complex integrated circuits in miniature size. As the device technology, is aggressively scaling to improve the device performance, the issues related to device interconnects, power, and reliability have become a major concern for the designers. These challenges make the design and validation of ASIC extremely complicated.

The primary idea of this work is to develop automation tools, to be used in the physical design flows to improve the efficiency of the design flow. The first tool named as variation analysis tool automates the on-chip variation modeling used in the post-layout timing closure phase in the physical design flows. The proposed variation analysis tool models three types of variations such as on-chip variation (OCV), advanced on-chip variation (AOCV) and parametric on-chip variation (POCV). The results of the proposed tool have compared with the Synopsys PrimeTime™ results, and the results show average around 98% accuracy compared to the PrimeTime™. The second tool is for automating repeater analysis in the physical design flows. The repeater automation tool can be used to automate the repeater or buffer insertion process, while technology process is changed from one to another. The tool can calculate the best possible repeater distance for any given metal layer and also, the number of repeaters, combinational or sequential for the user given distance and frequency. The accuracy of this script is compared with the repeater insertion based on the synthesis tools and also, the SPICE simulation.

Acknowledgments

First and foremost, I would like to take this opportunity to thank all of my committee members especially Dr. Ataur R. Patwary. This work would not have been possible without his constant guidance, motivation, and support. He has been an excellent source of inspiration and ideas. He has continuously provided me with confidence and having faith in me. His dedication and hard work towards his students inspires me a lot. I am grateful to Dr. Ataur for dedicating his precious time to me and regularly allowing me to meet him after work-hours, providing advice and feedback on research work.

I would like to extend my gratitude to Professor Xiaoyu Song, my thesis committee chair, for his support and help during the period of my thesis work. He had continuously helped and motivated me to continue doing good work while I was struggling.

I would also like to thank Professor John M. Acken for serving on my thesis Defense Examination Committee and providing valuable inputs on thesis draft. He has always dedicated his time to review and provide feedback on the thesis paper carefully. I would also like to thank Dr. Douglas Hall for delivering his constant support and motivation to complete this work. He has always helped me in completing all the formal process required to complete this thesis and also provided valuable advice regarding my future goals.

This work would not have been possible without the constant support of my parents, Judhistira Mahalik and Puspallata Mahalik. Their support and encouragement have continuously helped me to believe in myself and continue doing good work. I would also like to thank all of my family members for their support. I have been fortunate to have

many friends who have constantly cherished me despite my eccentricities. I thank Murali, Mahesh, Sai, Karthik, Ajay, Manoj, Vicky and my brother Sumanth who have always there for me and made my time here memorable and unique.

Table of Contents

Abstract	i
Acknowledgments	ii
List of Figures	vi
List of Tables	viii
Chapter 1: Introduction	1
1.1 Motivation	1
1.2 Contribution of this Thesis	3
1.3 Organization of Thesis.....	6
Chapter 2: Tool for Automating on Chip Variations Analysis in ASIC Design Flow	8
2.1 Introduction	8
2.2 Background.....	9
2.2.1 Inter-Die and Intra-Die Variation	9
2.2.2 Modelling Circuit Timing Behavior	11
2.2.3 Statistical Static Timing Analysis (SSTA)	14
2.2.4 Pre-Silicon Methodologies to Model Process Variations	16
2.3 Proposed Methodologies to Model Process Variations in ASIC Design Flow ...	24
2.3.1 Global OCV	24
2.3.2 Advanced On-Chip Variation (AOCV)	25
2.3.3 Parametric on Chip Variation (POCV)	28
2.4 Experimental Results	34
Chapter 3: Tool for Automating Repeater Analysis Process	44
3.1 Introduction	44
3.2 Background.....	45

3.3 Automated Repeater Analysis Tool.....	47
3.4 Experimental Results	52
Chapter 4: Summary and Future Work	63
References	65

List of Figures

Figure 1. 1 ASIC Design Flow	2
Figure 1. 2 Proposed Tools and Integration in the ASIC flow	6
Figure 2. 1 Classification of Variations	10
Figure 2. 2 Timing Graph Model of a simple Combinational Circuit	12
Figure 2. 3 A combinational Circuit Path	19
Figure 2. 4 Path with OCV constant Derate Factor	19
Figure 2. 5 Updated Cell Delays after Derate Factor is applied	20
Figure 2. 6 Logic Levels / Stage based AOCV	21
Figure 2. 7 Location / Distance based AOCV	22
Figure 2. 8 Propagation of Delay Distribution through timing graph [25]	23
Figure 2. 9 Overview of the OCV Methodology	25
Figure 2. 10 Overview of input derate file.....	26
Figure 2. 11 Overview of proposed AOCV method	27
Figure 2. 12 Overview of Statistical POCV Distance File	31
Figure 2. 13 Overview of the POCV Coefficient File	31
Figure 2. 14 Overview of the Proposed POCV Analysis.....	33
Figure 2. 15 Overview of Experimental flow	34
Figure 2. 16 Variation Analysis Tool User Interface.....	36
Figure 2. 17 Slack Comparison for Path-1 (FSG).....	37
Figure 2. 18 Slack Comparison for Path-1 (FIFO)	38
Figure 2. 19 Slack Comparison for Path-2 (FIFO)	40

Figure 2. 20 Slack Comparison for Path-3 (FSG).....	41
Figure 2. 21 Slack Comparison for Path-3 (FIFO)	42
Figure 3. 1 A simple circuit containing two cells as driver and receiver.....	45
Figure 3. 2 Repeater or buffer insertions design.....	45
Figure 3. 3 Wire delay model of the circuit	46
Figure 3. 4 Repeater model with attacker cells	47
Figure 3. 5 Flowchart for calculating the maximum repeater distance.....	49
Figure 3. 6 Flowchart for calculating the number of repeaters required	50
Figure 3. 7 Repeater Analysis Tool Usage Interface	51
Figure 3. 8 Layout of the repeater automation design	53
Figure 3. 9 Schematic of the repeater analysis design for distance calculation.....	54
Figure 3. 10 Schematic of the repeater analysis design for calculating repeater counts .	55
Figure 3. 11 Comparison of the repeater distances of metal layers when the minimum metal width is used.....	56
Figure 3. 12 Comparison of the repeater distance calculation for different metal layers when width 4X and BUFFX2 cell used	57
Figure 3. 13 Comparison of repeater distance calculations for different metal layers when minimum width and BUFFX4 cell	58
Figure 3. 14 Comparison of the repeater distance calculations for the different metal layers with width 3X and BUFFX4 cell	59
Figure 3. 15 Comparison of the repeater distance calculations for different metal layers when the width 4X and BUFFX4 cell used	60

List of Tables

Table 1 Design Details of FSG and FIFO.....	35
Table 2 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-1 (FSG).....	36
Table 3 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-1 (FIFO)	37
Table 4 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-2 (FSG).....	38
Table 5 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-2 (FIFO)	40
Table 6 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-3 (FSG).....	41
Table 7 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-3 (FIFO)	42
Table 8 Calculated repeater distance table for the minimum width	55
Table 9 Calculated repeater distance table for width 4X using the BUFFX2 cell.....	57
Table 10 Calculated repeater distance table for the minimum width using the BUFFX4 cell.....	58
Table 11 Calculated repeater distance table for the width 3X using BUFFX4 cell.....	59
Table 12 Calculated repeater distance table for width 4X using the BUFFX4 cell.....	60
Table 13 Number of repeaters required calculated by the tool, when various distance values are provided for metal layer M8 with the minimum width	61
Table 14 Number of repeaters required calculated by the tool, when various distance values are provided for metal layer M2, Width 4X	61

Chapter 1: Introduction

1.1 Motivation

Rapid advancement and innovation in the semiconductor research have continuously helped in designing efficient and complex integrated circuits in miniature size. Current modern Application Specific Integrated circuits (ASIC) have billions of transistors and capable of performing quite sophisticated operations. The device technology is aggressively scaling to improve the device performance to meet the demand. As the design becomes more complex and sophisticated day to day, designing and validating these ASICs, have become extremely challenging.

Designing and validating these complex ASICs have been made possible by adopting sophisticated electronic design automation (EDA) tools. These tools are used to model complicated circuits and also to improve the design efficiency by automating the process. Figure 1.1 shows the major phases in the ASIC design flow. Typically, the flow starts with design specifications followed by behavioral descriptions, RTL (Register Transfer Level) design and functional verification. These phases are referred to as the front-end of the design flow, and the backend design flow referred to the phases starting from the logic synthesis flow. Physical design flow begins with the logical synthesis flow, followed by the netlist generation, floorplan, place & route, physical verification and timing closure. Though there might exist many sub-flows between these phases such as logic equivalence checking, timing analysis, and DFT (Design for Test) phases. The following figure shows the major phases in the ASIC design flow.

Figure 1. 1 ASIC Design Flow

ASIC design is getting complicated day by day; this is leading to the requirement of more sophisticated EDA tools. As transistor size continues to shrink, the problems related to the device physical limitation has risen. Some of the challenges in the modern ASIC design are, interconnect related issues such as the signal routing and RC delay optimizations,

process variations, design testability and maintaining high performance while consuming lower power.

Though there are wide varieties of EDA tools available in the market from reputed EDA vendors, these tools are quite complicated to use and require automation. These tools need automation for modeling current design challenges like process variations effect and repeater analysis process in the IC design flow to give the user an early indication about using particular design aspects. The tool will also reduce design time and effort. Current EDA tools use table-based values provided by characterization teams to model variations and perform timing analysis. If there are any significant changes in characterization table, the tool needs to reevaluate the timing for the whole design considering the updated table values. If we have a huge design consisting of millions of gates, this process can be slower. Buffers or repeaters insertion is one of the critical phases in physical IC Design flow. Buffers are used to fix timing and transition violations in the design. If there are any changes in the process and design constraints like the width of metal or metal layers or shielding, this may affect the circuit timing and signal integrity (SI). Therefore, when we have changes in the process constraints, we might need to redo the whole buffer analysis process to recheck the timing and SI criteria.

1.2 Contribution of this Thesis

We have developed two automation tools which can be easily integrated with existing modern EDA tools in the ASIC design flow. These tools are user-friendly and give designer, an early indication of certain design aspects. The primary goal of these tools is to automate the variations and repeater analysis process in physical Integrated Circuit (IC)

design flow to achieve the design process more efficient by reducing manual effort. Primary contributions of this work are following,

- A user-friendly tool has been developed to model process variations in timing analysis phase. The tool models OCV (On-chip Variation), AOCV (Advanced On-chip Variation) and POCV (Parametric On-chip Variations) and can produce results efficiently if there are any updates in the characterization tables or process files which are used to model variations. The user can get analysis data in advance without doing actual analysis using inbuilt or standard tool method, which might take more time.
- Another tool has been developed to automate the repeater analysis process in the IC design flow. The developed tool can calculate the maximum repeater distance a signal can travel considering the maximum transition value limit; it can also calculate the number of repeaters required for a given length of signals based on the cycle time limit or frequency of the design. The tool will perform these calculations if we have changes in the process constraints like metal width or metal layers in an automated way so that user will have an advance idea about the repeater distances and repeater counts without waiting for the full design flow to complete.

The variation analysis tool can be easily integrated into the Synopsys PrimeTime™ EDA tool, and repeater automation tool can be integrated to Synopsys IC Compiler™ tool. Proposed tools and their appropriate integration in the physical IC design flow has shown in Figure 1.2.

Figure 1. 2 Proposed Tools and Integration in the ASIC flow

1.3 Organization of Thesis

This thesis document has organized into four chapters. Chapter 2 gives background and concepts related to process variations modeling in timing analysis phase of the IC design

flow. On-Chip Variation (OCV), Advanced on Chip Variation (AOCV) and Parametric on Chip Variation (POCV) models have been discussed elaborately. This chapter also gives the insight of the design details and algorithm related to the proposed variation modeling tool followed by the design experiments and results. The accuracy of the proposed techniques is compared with Synopsys PrimeTime™ tool.

Chapter 3 discusses details regarding the repeater automation tool. Importance of repeaters or buffers insertion in the current design flow is also discussed. The overview and design details of the proposed model are provided. Experimental setup and results for the repeater analysis are discussed.

Chapter 4 is the final chapter of this thesis, briefly provides the summary of the proposed work and future possibilities for extending this work.

Chapter 2: Tool for Automating on Chip Variations Analysis in ASIC Design Flow

2.1 Introduction

As the semiconductor industry continues to strive with Moore's law and process technology continues to scale, the transistor densities on a single die doubles in approximately every two years [1]. As the process technology continues to shrink in nanoscale CMOS circuits, we have started facing challenges related to physical limitation of devices. Among other significant challenges, the process variation plays one of the major roles in the semiconductor manufacturing industry. Process variations also affect the design performance, reliability, and yield.

Timing analysis of design ensures that the design will meet the required timing criteria and it will operate reliably at the specified clock speed. In traditional timing analysis approach, specific or fixed process conditions are considered, and analysis is performed respectively. As the variability in process parameters increases, it also affects in determining the circuit timing behavior. It would be too pessimistic if only deterministic corner approach is considered for accounting the timing behavior of the circuit with intense process variations. The designer needs to keep in mind about the various process corners such as the slow corner, fast corner, best-case and worst-case conditions. Therefore, process variation has become one of the critical challenges in the static time analysis (STA) in the past few years [2].

In this thesis work, we have proposed three pre-silicon approaches that would be used to model process variations while performing the timing analysis. Proposed methodologies can be easily integrated into the modern ASIC design flow. The first approach is based on

the fixed corner-based derate method, and later techniques handle, systematic and local variations based on the statistical approach. Results of the proposed methodologies have been compared with the Synopsys PrimeTime™ tool results, and the results show reasonably good accuracy. The proposed methods could be seamlessly integrated into the modern EDA tools that would help to model process variation effortlessly and also user doesn't need to have the advanced tool license from the EDA tool vendors to model these variations.

2.2 Background

Process variations can be defined as the changes in the physical characteristics of devices. Figure 2.1 shows the classification of process variations. In the circuit design point of view, process variations can be categorized into two types. These are intra-die variation and inter-die variation [3]. These two variations can be again sub-categorized as wafer-to-wafer, die-to-die, lot-to-lot and with-in-die variations.

2.2.1 Inter-Die and Intra-Die Variation

Inter-die variation is defined as the variations in the process parameter with the similarly manufactured dies. The variations might be observed in wafer-to-wafer, lot-to-lot or from die-to-die. Variations could exist on the same wafer or different wafers or may be on different lots. Generally, lot-to-lot and wafer-to-wafer variations are more random in nature [6]. Intra-die variation is defined as the variations that occur within the same die or chip. Intra-die variations are generally considered more spatially distributed and location dependent. One simple example of intra-die variation is the gate length of the device. The gate length of the device could be smaller or larger on the same die. Intra-die and inter-die

variations can be again classified into two types, random variation, and systematic variation.

2.2.1.2 Systematic and Random Variation

Systematic variations account for the universal part of the process variations. In the systematic variation, process device parameters like oxide thickness, gate length, width and doping concentration can vary equally for all the transistors. So systematic variation can be spatially correlated. Systematic variations can be modeled by performing a comprehensive analysis of the layout during the manufacturing process. Example of systematic process variation includes chemical or mechanical polishing, various lithography effects, etc. [4][5]. Random or non-deterministic process variation is entirely random in nature, and these variation does not show any correlation, and so we consider the probability while modeling random variation. The example such as random dopant concentration fluctuation and line-edge roughness are random variations.

Figure 2. 1 Classification of Variations

Spatial variation effects arise mostly due to the logical gates proximity to each other within the die. More closely spaced devices tend to exhibit many similar variations than the device which is located far apart [7]. On the other hand, random variation does not depend on the location of the device. Variations in the gate length, gate width, oxide thickness, channel doping and interconnect are some of the examples of the random variations. The variations observed in the device interconnect geometry also impact the interconnect parasitic (Resistance-Capacitance (RC)), which might also affect the device performance.

2.2.2 Modelling Circuit Timing Behavior

Timing verification process is quite a complex task in the circuit design. Static Timing Analysis (STA) and Dynamic Timing Analysis (DTA) are two techniques widely adopted for verifying the circuit's timing behavior. In the Dynamic Timing Analysis (DTA), the circuit timing behavior is calculated by introducing input test vectors. DTA performs logical simulation by considering the input vectors. The timing coverage will depend on the quality of the test vectors so that maximum timing path can be covered. Generally, DTA based simulation timing results are more input vector values dependent and it can also verify the functionality of the design. DTA is more appropriate use for the design consists of multiple clock domains. DTA is simulation based, therefore achieving maximum coverage by applying simulation test vectors is a challenging task, and it is also noticeably slower. Therefore, Static Timing Analysis (STA) is a more viable option and has been widely adopted for modeling circuit timing in the industry from past few decades over DTA.

2.2.2.1 Timing Graph Model

A digital circuit can be represented using the Directed Acyclic Graph (DAG). Suppose we have a timing graph G , 'V' represents nodes and 'E' represents edges such as $G(V, E)$. Each input and output pair of a gate in the circuit can be represented as an edge in G , and each signal line in the circuit can be expressed as a node in 'V' [4]. Figure 2.2 illustrates the circuit timing topological model.

Figure 2. 2 Timing Graph Model of a simple Combinational Circuit

2.2.2.2 Static Timing Analysis (STA)

Unlike the Dynamic Timing Analysis (DTA), Static Timing Analysis (STA) does not perform logic simulation; hence it does not require any input vectors. Therefore, it checks delays for all the paths in the circuit like Critical Path, Data Path, and Clock Path and even detects False Path with respect to timing constraints and calculates circuit timing failures. STA calculates the circuit timing using formal and mathematical equations. In contrast to DTA, STA does not verify the functionality of the design. This approach is much faster

and simpler to integrate into the tool; hence this has been popular in the industry for many years [8][9].

2.2.2.3 Corner Based Static Timing Analysis

Process variation is one of the critical issues in the ASIC design. In traditional STA, we use deterministic or corner-based approach to model variations. In the corner-based STA, device parameters like the oxide thickness, gate length, temperature, and voltages are assumed as systematic and uniformly applied for all the devices in the circuit. Hence, the corner-based STA can model global variations by considering multiple corner files. STA assumes process parameter variations to be fixed while performing the analysis. In this technique, generally, two different corners are considered for each parameter variations. These are typically referred as best case and worst case. For a single parameter variation, we have two cases. Similarly, for two parameter variations, we would have four corners, and it continues.

As parameter variations increases, the number of corners required for the timing analysis also increases exponentially. Therefore, for 'n' number parameter variations, we need the 2^n number of corners for the timing analysis. The above technique is highly inefficient and bottlenecks performance in the corner-based STA. To restrict the number of combinations, we can consider only one corner, i.e., worst-case corner for each parameter variations. However, it would be highly pessimistic. Statistical Static Timing Analysis (SSTA) was proposed to account process variations more accurately.

2.2.3 Statistical Static Timing Analysis (SSTA)

Deterministic STA was a simple and straightforward approach to predict the circuit timing behavior. However, as the number of process parameters is increasing at lower node size, this approach suffers from the performance and accuracy issues. Statistical Static Timing Analysis (SSTA) has recently emerged as one of the practical approaches to reduce pessimism over corner based or deterministic static timing analysis to model process variations. SSTA uses the statistical model of delay propagations to overcome accuracy and speed issues observed in corner based static timing analysis. SSTA considers variation parameters as random variables with the statistical distribution.

SSTA considers the random process variations of within die or local variation as the probability distribution function, instead of fixed values and manages to model the parameter variations more accurately. It propagates both fall and rise delays, unlike traditional STA which propagates only the fixed delay value.

2.2.3.1 Existing SSTA Related Techniques

The typical corner-based approach is more pessimistic to model variations at lower process node. New SSTA techniques have evolved to mitigate issues faced in modeling the local variations. Numerical integration method is a technique in which, the yield of the circuit is computed for a particular delay by performing the numerical integration over the process parameter space [10]. This method has the high level of accuracy; however, it is very expensive to implement in practice, as the runtime is quite high for a circuit with the larger number of critical paths.

Another popular technique is the Monte-Carlo simulation method. In this method, regions of the probability are first identified, and samplings are taken for the particular regions instead of statistically sampling entire sample space. Samples are chosen by taking the Probability Density Function (PDF) of device parameters. Simple deterministic static timing analysis is performed for each sample to calculate the circuit delay [12,13]. Monte-Carlo method can handle the variations in a better manner and performs faster than the numerical integration method. However, this method also suffers from runtime issues, since this method uses the traditional deterministic static timing analysis (DSTA).

2.2.3.2 Probabilistic Analysis Method

The probabilistic method considers the arrival time, gate delays and slack as the probabilistic model with random variations, unlike the Monte Carlo and integration method where we consider sample space enumerations. Mostly addition and maximum (comparison) operations are performed in this approach for propagating the delay values. There are two types of probabilistic analysis method, Path-Based Approach, and Block-Based Analysis.

2.2.3.3 Path-Based Approach

In the path-based algorithm, critical paths are generally identified, and probability analysis is performed over these paths to determine circuit delay distribution. By performing the sum (add) operations over all the edges, the delay distribution is calculated. Finally, the final circuit delay is calculated by taking the maximum statistical operations of all the path delays [14] [15]. For calculating delays over all the paths, it considers the approach similar to the breadth-first search (BFS). The shortcoming of this method is identifying all the

critical paths and finding delays over all the edges. As the circuit size increases, the complexity increases exponentially with respect to the critical paths. While considering the paths, few critical paths could be missed, and this could affect the accuracy.

2.2.3.4 Block Based Approach

The block-based approach follows an algorithm similar to the depth-first search (DFS) to traverse the circuit in a topological way or in a hierarchical manner. In this method, each node is considered as blocks. At each node, the arrival time is added to the edge delay by performing the sum (add) operation, and the delay values are propagated. The final arrival time is calculated by taking the maximum (compare) operations of these two arrival times at each node. As the circuit size increases, the runtime and complexity increase linearly using this approach, unlike the path-based approach. Path-based and block-based approach have individual merits and demerits. The path-based approach is more accurate but suffers from higher computation time, whereas the block-based approach tends to show less accuracy over the path-based approach.

2.2.4 Pre-Silicon Methodologies to Model Process Variations

Since past five decades, the process technology in the semiconductor industry continues to scale, and we have reached from 10 μm to the current 5nm node. As the technology node continues to scale, the modern ASICs design flow has also evolved quite rapidly. Current ASIC design flow starts with the capturing the design specifications, followed by collecting behavioral descriptions of the design, developing RTL model, functional design verification, logic synthesis, gate-level netlist generation, floor-planning, place & route, physical verifications, timing closure and finally sign off. These are some of the critical

phases in the ASIC design flow, but in reality, we might have hundreds of miniature sub-flows.

Apart from other various challenges in the flow, the timing closure phase is considered as one of the most critical phases in the ASIC design flow. Timing closure issues are increasing rapidly at lower technology node and no longer can be ignored. The timing closure phase ensures that the predefined logic in the design meet the required timing criteria so that the chip can operate reliably at the specified clock rate.

At the lower CMOS technology node, the designer continues to face problems such as significant variations in the gate delay, net delay, voltage, and temperature, etc. A designer needs to come up with the advanced tools to model and account for these issues in the early phase of the design. Generally, to model process variations at the timing closure phase, we have been introduced majorly three types of techniques in the modern EDA tools. These are On-Chip Variation (OCV), Advanced On-Chip Variation (AOCV) and Statistical timing analysis method for handling variations [18]. The statistical approach has been named accordingly by respective EDA tool vendors. Synopsys® uses the term Parametric On-Chip Variation® (POCV) for statistical timing analysis, whereas Cadence® refers to it as the Statistical On-Chip Variation® (SOCV) [19]. These two methods are currently industry standards for modeling variations and timing analysis in the modern ASIC design. In the following sections, details regarding the OCV, AOCV, and Statistical based OCV have been discussed.

2.2.4.1 On-Chip Variation (OCV)

In OCV analysis, we follow the conservative approach to predict the timing behavior and accounts systematic or non-random variations. In OCV, derating factors are used to slow or speed up certain cells or nets in the design. So basically, two conditions are considered and referred as, maximum condition and minimum condition. For set-up timing analysis, the maximum delay value is generally applied to the data-path, and minimum delay value is applied to the clock-path, whereas in case of hold analysis we apply maximum delay for clock-path and minimum delay for data-path. Specifying maximum and minimum conditions for the design can be done using several ways. The max and min delay values can be specified using the Standard Delay Files or can be defined in the technology library as best case and worst-case conditions called as delay variations in OCV [20]. If best-case and worst-case options are selected, we account the maximum delay values for the data-path under worst-case conditions and minimum delay values for the clock-path under best-case conditions.

We can also provide a constant global derating factor value, that will be applied for data-path and clock-path. If the constant derating value option is selected, two constant derate factor values are provided, one as early and one as late. These constant derate values are applied to all the clock paths and data-paths respectively in the design. The problem with the global fixed derating based OCV is, it adds additional pessimism to the analysis. All the data-paths and clock-paths might not behave similarly, and it is pessimistic to assume the delay values as one global fixed value for all the data-paths and clock-paths cells in the design.

Figure 2. 3 A combinational Circuit Path

Figure 2. 4 Path with OCV constant Derate Factor

After the derating factor is applied, the cell delays will change based on the respective derating factor. Figure 2.3 illustrates the delay of the path before applying the derate, and Figure 2.4 shows the changed delay values after derating factors are applied. The new cell delays are calculated using the following equations [21], and final values are shown in figure 2.5.

$$delay_{new} = original_delay + [(derate_factor - 1) * abs(original_delay)]$$

If the original cell delay is a positive value, then the equation can be written as follows.

$$delay_{new} = original_delay * derate_factor$$

If the original cell delay is a negative number value, then the equation will change as below.

$$delay_{new} = original_delay * (2 - derate_factor)$$

Now if we apply the OCV constant derate factor of late 1.05, then the delays will change as shown below for the setup analysis.

Figure 2. 5 Updated Cell Delays after Derate Factor is applied

Generally, OCV derates factor values are provided by the process manufacturing foundries, and values might differ based on the corners and drive strength of the device. The advantages of fixed on-chip variation-based analysis are that it is simple and straightforward analysis approach [22]. By just considering constant derate value, we account for the variations in the whole design. However, it is too pessimistic to consider fixed global derate values for all the paths in the design and also at the lower node; it would increase costs and might also affect the performance of the design [23][24].

2.2.4.2 Advance on Chip-Variation (AOCV) or Stage Based OCV

Though OCV is a simplistic approach to model variations, it is not highly accurate, and it also adds additional pessimism to the analysis. We cannot just assume that all the cells in a path would behave similarly and consider a single global derate factor value for the whole design is a challenging task. Therefore, to model variations realistically, random variations, systematic variations, locations and logic stage parameters should be taken into consideration. Advanced On-Chip Variation (AOCV) considers the logic levels and locations of the design while modeling variations. The delay distribution of cells and nets are varied depending upon the locations and device load. This would be the more realistic approach to model systematic and random variations and also lead to reducing timing slack. In AOCV, we generate the derate values that are the function of logic levels or depth

(number of combinatorial cells in the path) and locations of the design. From statistical analysis, we know that, as the logic depth increases on a path, the random variations tend to decrease. Therefore, we would likely to observe less random variations as the logic level increases.

Similarly, as the distance increases or if the cells are placed far apart in the design, we would likely to observe more systematic variations and vice versa. In AOCV, instead of defining single global derate value, variable derate factors, based on the distance (locations), and logic levels are generated and provided in a tabular format to the tool. The appropriate derate factor values for the particular cell is selected and applied, while accounting process variations.

Figure 2. 6 Logic Levels / Stage based AOCV

In level or stage based AOCV, the derating factor is a function of the logic levels or depth of the clock or data path. As shown in Fig 2. 6, the clock-path has logic depth or levels of 2, and the data-path has the logic depth of 5. In the level based AOCV, the derating factors are used based on the logic levels of the path [18]. Statistical analysis can be performed

considering the SPICE model, and the derating factor can be calculated at each cell stage of the path.

Figure 2. 7 Location / Distance based AOCV

Similarly, in location-based AOCV, the physical locations of the path, after the placement is used to model the variations. The location of the path is calculated by calculating the bounding box diagonally that covers all the instances, such as cells and nets on the clock-path and data-path. As shown in Fig 2.7, location-based AOCV derating factors are considered based on the distance of the path. Here, a timing path is a combination of both the data-path and clock-path.

2.2.4.3 Parametric On-Chip Variations

AOCV models the variation using derate factors which are the function of the location (distance) and logic levels (stage). Though the method is more appropriate for modeling variations than the traditional OCV, it still shows inaccuracy in the graph-based analysis [25]. As we know, the golden approach would be the statistical static timing analysis

(SSTA) for the path, but it requires the statistical library characterization and parasitic effects variations extraction process.

Figure 2. 8 Propagation of Delay Distribution through timing graph [25]

Parametric on Chip Variation (POCV) handles the local random variations without requiring the statistical library characterizations. As we could see from Figure 2.8, it considers the instance delays as a function of the random variable, that is specific to that particular instance. Therefore, the instance delay is parameterized based on the random variable [25]. It uses the statistical single single-parameter derating value for modeling random variations. POCV also handles the systematic variations by considering location-based derate factors similar to the AOCV. The new cell delay after the POCV can be defined as below,

$$delay_new = delay_original + delay_var * P$$

delay_original is the original delay of the cell without applying any derate, delay_var is one standard deviation value of the delay distribution of the cell and P is the standard random variable $N(0,1)$. Once these parameters are obtained, we could calculate the delays at each point of the path and propagate delays to calculate arrival and required time for the whole path.

2.3 Proposed Methodologies to Model Process Variations in ASIC Design Flow

As we have seen, modeling process variations are one of the critical tasks for the STA engineers and the challenges are continuously increasing at lower technology node. Even user needs to have an advanced version of the EDA tools license from the commercial EDA vendors to run and analyze these advanced methods. Configuring analysis environment for variation analysis using the EDA tool environment is a complicated process. We have introduced three methodologies inspired from the OCV, AOCV and POCV techniques to model process variations in an automated way. The proposed techniques have been integrated into one of the commercial EDA tools, and the results have been compared with one of the commercial EDA tool results. Following sections explain details related to proposed methodologies.

2.3.1 Global OCV

In global derate value based OCV approach, derate factors should be provided for modeling variations. Generally, we get these derate factors values from process characterization teams. Once the derate factor numbers have been provided, the tool uses these derate factors and compute the new delay values of cells on the clock and data path. Based on the

cell delay values, final arrival and required time of the path is calculated and slack is computed respectively.

Figure 2. 9 Overview of the OCV Methodology

2.3.2 Advanced On-Chip Variation (AOCV)

Unlike global or fixed derate based OCV, in AOCV analysis derate factor is applied as a function of distance and logic levels of the path. When a path is referred here, it might be data-path, clock-path or combination of any of these. Basically, we provide the derate factor values in the tabular format through a side-file and using this side-file the derate values are applied respectively.

```

version:1.0
object_type: design
rf_type: rise fall
delay_type: cell
derate_type: late
object_spec: FibGen
voltage: 1.2
depth: 10 11 12 13 14
distance: 10000 20000
table: 1.15 1.14 1.13 1.12 1.11 \
          1.16 1.15 1.14 1.13 1.12

```

Figure 2. 10 Overview of input derate file

As we could observe from the Fig 2.10, when distance value increases left to right, the derate value increases from top to bottom of the table. This is because, as the distance increases the random variations tend to increase. Also, from the figure we could see as the logic levels or depth value increases from left to right, derate values decreases from left to right. This is because of the systematic variation, as the logic levels increase variations tends to cancel out. Suppose the logic depth of the path is 11, distance bound value is 10000, then the derate factors that would be selected from the table as '1.14'. Above table is considered for a late type path and the format for early type would be vice versa. then the derate factors that would be selected from the table as '1.14'. Above table is considered for a late path and the format for early path derate would be vice versa.

Figure 2. 11 Overview of proposed AOCV method

Figure 2.11 shows the flow chart for the proposed AOCV method for modeling variations while performing timing analysis. The new cell delay is calculated using the following formula.

$$delay_new = delay_nominal * derate_factor$$

Total derating factor that is applied to a timing arc, which is the product of AOCV derate factor and guardband derate factor. Guardband derates consists of factors such as IR drop values, tool error values, and margins, etc. This guardband derate factor has no impact outside the AOCV analysis. Similarly, incremental derate value is used to adjust the derate value of objects like cells or nets. Suppose for a cell, original derate is x, the incremental derate value is specified as y, then the final derate would be (x+y) and (x-y) for early and late derate. If the guardband and incremental timing analysis option is selected, then the delay is computed as below.

$$delay_new = (delay_nominal * (aocv_guardband * derate_factor + incr_value))$$

In this approach instead of providing a single derate factor value, multiple derate values based on distance and depth are provided using the side file. The appropriate derate value is selected based on the logic depth and distance value of the path. Once the derate factor is selected, the new delay is computed using the derate factor and then arrival time, the required time is calculated respectively. After the arrival and required time are obtained, the slack value is computed by taking the difference of arrival and required time.

2.3.3 Parametric on Chip Variation (POCV)

In POCV method two side files are provided as the input. One for the coefficient value, which is computed by taking the ratio of delay variation value and the nominal delay value. In the other file, we specify the distance based derate factor values. This is used to model the spatial variations or location-based variation [25].

$$Coefficient = \frac{\sigma (delay\ variation)}{\mu (nominal\ delay)}$$

After coefficient value and distance based derate factor values are obtained, we could also provide the guard band value and incremental value to make the analysis more conservative, because by adding these two factors we could account values related to the IR drop, tool-related error or any final adjustment to derate value. After obtaining these values as inputs, we calculate the mean value for each cell in the timing path. The cell delay is calculated using the following formulas. These equations are extracted from the Synopsys POCV application note [25].

$$Mean_Cell = delay_nominal * (POCV\ guardband * POCV\ distance\ derate + Incremental\ derate)$$

$$\begin{aligned} Sigma_Celldelay \\ = delay_nominal * (POCV\ guardband * POCV\ coefficient \\ * POCV\ coef\ scale\ factor) \end{aligned}$$

Final “Mean” value is calculated by just adding the previous stages mean values until that stage. For N number of stages,

$$Mean = Mean_{cell1} + Mean_{cell2} + \dots + Mean_{Cell\ N-1} + Mean_{Cell\ N}$$

Final sigma or sensitivity is calculated using Root Sum Square of all the sigma cell delay values until that stage. For N number of stages, sigma is calculated as,

Sigma

$$= \sqrt{\sigma_{celldelay(1)}^2 + \sigma_{celldelay(2)}^2 + \dots + \sigma_{celldelay(N-1)}^2 + \sigma_{celldelay(N)}^2}$$

Incremental delay value at the particular timing point is calculated using the following formula,

$$Incr_value = Mean \pm K * \frac{(Sigma_{celldelay})^2}{Sigma_{Slack}}$$

$Sigma_{Slack}$ is calculated using the following equation,

$$Sigma_{Slack} = \sqrt{Sigma_{Arrival}^2 \pm Sigma_{Required}^2}$$

Add or subtract operation is performed based on the value, if it's positive then add operation is performed, if its negative then subtract operation is performed.

$$Sigma_{Required}^2 = Sigma_{Capturepath}^2 + Sigma_{crpr}^2$$

Where,

$$Sigma_{crpr}^2 = Sigma_{commonlaunchpath}^2 + Sigma_{commoncapturepath}^2$$

Generally, the sigma delay of a path is calculated as the Root Sum Square (RSS) of all cell delays on the path.

```
version: 4.0  
ocvm_type: pocvm  
object_type: design  
rf_type: rise fall  
delay_type: cell net  
derate_type: late  
distance: 10000 20000  
table: 1.11 1.12
```

Figure 2. 12 Overview of Statistical POCV Distance File

```
version: 4.0  
ocvm_type: pocvm  
object_type: design  
rf_type: rise fall  
delay_type: cell net  
derate_type: late  
object_spec: FibGen  
coefficient: 0.05
```

Figure 2. 13 Overview of the POCV Coefficient File

POCV considers the derate values using side files. Though, Liberty Variation Format (LVF) file can be used for slew table in POCV, in our case coefficient-based side file has been considered. Figure 2.12 and Figure 2.13 show the overview of both side files in which, coefficient value and distance based derate factors have been provided. The distance derate value is selected based on the distance bounding box calculated for the path, and coefficient value is considered based on the value specified in the coefficient field. The coefficient

value considered as the variation factor while calculating sigma for the path in the POCV analysis. Suppose the distance value is calculated as 20000 units for the path, then the derate factor is chosen as '1.12'. Figure 2.14 shows the proposed POCV analysis flow. As seen from the Figure 2.14, the derate files are provided to the tool, then bounding boxes for cells and nets are calculated for the path. After the distance calculation, mean and sigma for the path are computed. Finally, the timing slack for the path is calculated.

Figure 2. 14 Overview of the Proposed POCV Analysis

2.4 Experimental Results

In this section, obtained results of the proposed techniques have been discussed and compared with the results of the commercially available EDA tool such as Synopsys PrimeTime™. RTL code for the considered designs is developed using Verilog HDL, then synthesized using the Synopsys Design Compiler®. We are using Synopsys technology library for analysis in this experiment. Backend steps like Placement and Routing etc. are done using the Synopsys IC Compiler™. After post layout netlist generation, the parasitic extraction is performed and the ‘Standard Parasitic Exchange Format (SPEF)’ files are obtained from the Synopsys IC Compiler™. SPEF files are used for distance calculations while performing the AOCV or POCV analysis. Proposed variation analysis techniques are developed using Tool Command Language (TCL) programming language and compiled directly on the inbuilt PrimeTime™ shell.

Figure 2. 15 Overview of Experimental flow

Fig 2.15 shows the proposed analysis flow. PrimeTime™ is chosen as the primary framework, and the developed variation analysis tool is provided to the PrimeTime™ along the other required derate side files and design related files.

For analysis purpose we have considered two designs, first design is a 64bit Fibonacci Sequence Generator (FSG), which generates the 64-bit Fibonacci sequences and second design is an asynchronous First Input First Output (FIFO) design which consists of two clocks. Both synthesis and post-layout netlist generations are done using the Design Compiler™ and IC Compiler™ for these designs. Total six paths, three from each design are selected. These three paths are combinations of the longer and shorter paths. In a long timing path, there might exist a large number of cells in the paths, and short paths will have lesser cells.

Design Name	Design Area	Total Cells Count
FSG	4271.67	634
FIFO	8510.02	1742

Table 1 Design Details of FSG and FIFO

Design details like the area and cell count for both FIFO and FSG design have been given in Table 1 above. FIFO design is comparatively bigger design than the FSG design, and it also has multiple clocks. Design area of the FIFO is almost twice the area of the FSG design, and the same applies for the cell count too.

First, timing analysis is performed for the timing paths using PrimeTime™ inbuilt OCV, AOCV and POCV technique. Later developed variation analysis tool is provided to the PrimeTime™ and analysis is done for the different paths of both FSG and FIFO design using proposed OCV, AOCV, and POCV model. Finally, the accuracy of the obtained results is compared and discussed.

```

pt_shell> variation_analysis -help
Usage:
variation_analysis # Reports OCV, AOCV and POCV analysis for the given timing path
  -from <-from>      ( Required, From pin/port of the timing Path)
  -to <-to>          (Required, To pin/port of the timing Path)
  -delay_type <-delay_type>
                      (Required, 'max' = setup analysis, 'min' = hold analysis)
  -mode <-mode>
                      (Required, 'OCV' for OCV analysis,'AOCV' for AOCV analysis,'POCV' for POCV analysis)
  [-debug_mode <-debug_mode>]
                      (Optional, 'true' to turn on debug mode, 'false' to turn off debug mode, default is set to true)
pt_shell>

```

Figure 2. 16 Variation Analysis Tool User Interface

After the developed tool is integrated into the Synopsys PrimeTime™ environment, the user can use the command ‘variation_analysis -help’ to get information regarding the usage of the tool. Figure 2.16 shows the interface of output help message displayed by the tool when help is needed by the user.

Design	Path Details	Path Depth	Path Cell Distance	Path Net Distance	Proposed Method Slack	PrimeTime ™ Slack	Accuracy
FSG	Path-1 OCV	NA	N/A	N/A	0.85548	0.85549	99.99 %
	Path-1 AOCV	1	8392.07	7794.39	0.87264	0.87265	99.99 %
	Path-1 POCV	1	8392.07	7794.39	0.87078	0.87122	99.94 %

Table 2 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-1 (FSG)

Table 2 shows the comparison of slack value calculations for path-1 FSG design. Path-1 of the FSG design is a small path as the depth of the path is quite small. The table also shows the path depth, cell and net distances calculated using the developed tool for the path-1. As

seen from the Figure 2.17, the accuracy between PrimeTime™ calculated the slack value and the developed tool calculated slack values are above 99% for all the analysis mode.

The slack values are measured in terms of the delay unit.

Figure 2. 17 Slack Comparison for Path-1 (FSG)

Design	Path Details	Path Depth	Path Cell Distance	Path Net Distance	Proposed Method Slack	PrimeTime™ Slack	Accuracy
FIFO	Path-1 OCV	N/A	N/A	N/A	0.69383	0.69382	99.99 %
	Path-1 AOCV	3	19996.6	15246.7	0.73633	0.73722	99.87 %
	Path-1 POCV	3	19996.6	15246.7	0.74108	0.74156	99.93 %

Table 3 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-1 (FIFO)

Table 3 shows the comparison of the slack values calculated by PrimeTime™ and developed tool, for path-1 of the FIFO design. The analysis is done for three variations mode OCV, AOCV, and POCV respectively. Path depth is 3, and as seen in Figure 2.18, the accuracy of all these analysis modes are above 99%.

Figure 2. 18 Slack Comparison for Path-1 (FIFO)

Design	Path Details	Path Depth	Path Cell Distance	Path Net Distance	Proposed Method Slack	PrimeTime™ Slack	Accuracy
FSG	Path-2 OCV	N/A	N/A	N/A	0.34450	0.34451	99.99 %
	Path-2 AOCV	10	22272.9	45479.8	0.48989	0.49237	99.50 %
	Path-2 POCV	10	22272.9	45479.8	0.49403	0.49423	99.94 %

Table 4 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-2 (FSG)

Figure 2. Slack Comparison for Path-2 (FSG)

Table 4 and Table 5, show the slack comparison values of path-2 for both FSG and FIFO design respectively. As we could observe from the table path-2 of the FSG design is the relatively larger path and has the depth value of 10, while the FIFO design has path depth value of 6. As we could observe from Figure 2.19 and Figure 2.20, OCV, AOCV and POCV slack accuracies are considerably good for both FSG and FIFO design. Slack values are measured in terms of the delay unit. The calculated slack value using proposed AOCV method is 0.5881 delay unit, compared to PrimeTime™ slack which is 0.5908 delay unit for path-2 of FIFO design. The accuracy is above 99% for all these analysis modes when compared with the PrimeTime™ calculated slack values.

Design	Path Details	Path Depth	Path Cell Distance	Path Net Distance	Proposed Method Slack	PrimeTime™ Slack	Accuracy
FIFO	Path-2 OCV	N/A	N/A	N/A	0.51722	0.51723	99.99 %
	Path-2 AOCV	6	23202.3	22354.3	0.58881	0.59080	99.66 %
	Path-2 POCV	6	23202.3	22354.3	0.62249	0.62456	99.66 %

Table 5 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-2 (FIFO)

Figure 2. 19 Slack Comparison for Path-2 (FIFO)

Design	Path Details	Path Depth	Path Cell Distance	Path Net Distance	Proposed Method Slack	PrimeTime™ Slack	Accuracy
FSG	Path-3 OCV	N/A	N/A	N/A	0.43470	0.43471	99.99 %
	Path-3 AOCV	9	15025.64	23244.2	0.56091	0.56519	99.24 %
	Path-3 POCV	9	15025.64	23244.2	0.53515	0.53906	99.27 %

Table 6 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-3 (FSG)

Table 6 and Table 7 show the slack comparison data of the path-3 of both FSG and FIFO design. Path-3 has the depth of 9 for FSG and 3 for FIFO design respectively. As seen from Figure 2.21 and Figure 2.22, slack calculated by AOCV and POCV is considerably improved over OCV calculated slack.

Figure 2. 20 Slack Comparison for Path-3 (FSG)

Design	Path Details	Path Depth	Path Cell Distance	Path Net Distance	Proposed Method Slack	PrimeTime™ Slack	Accuracy
FIFO	Path-3 OCV	N/A	N/A	N/A	0.229060	0.229061	99.99 %
	Path-3 AOCV	3	12024.01	10560.40	0.26539	0.265401	99.99 %
	Path-3 POCV	3	12024.01	10560.40	0.26920	0.267810	99.48 %

Table 7 Slack Comparison of PrimeTime™ vs. Proposed Method for Path-3 (FIFO)

As seen from Table 7, the net and cell distances are calculated as 10560.4 and 12024.01-unit distance for path-3 of the FIFO design. We could also see from Figure 2.22 POCV slack has been improved slightly from AOCV slack, as we know POCV uses statistical approach over AOCV.

Figure 2. 21 Slack Comparison for Path-3 (FIFO)

We compared and analyzed results for three different timing paths taken from both the FSG and FIFO design respectively. We observed that the AOCV and POCV show significantly better slack values compared to the OCV calculated slack. Results achieved from the proposed methodologies show that the accuracy of proposed techniques is pretty compelling compared with the PrimeTime™ computed results. The results show that the proposed methods have approximately above 99% accuracy compared to the PrimeTime™ inbuilt variation analysis tool.

Chapter 3: Tool for Automating Repeater Analysis Process

3.1 Introduction

As process technology advances and node size continues to shrink, reducing the interconnect delay is one of the critical issues in the Deep Sub-Micron technology (DSM) designs. Apart from logic optimizations and process variations issues, the interconnect optimization is one of the major hurdles. As CMOS technology continues to scale, modern process technology uses multiple metal layers for interconnect. Most of the interconnect related optimizations are done in the placement and route phase of the ASIC design flow.

Buffer or repeater insertion technique is one of the simple yet effective methods to handle the interconnect delay issues to improve the design performance and timing criteria [26], [27]. Hence this technique is extensively used in the current ASIC designs to solve the interconnect delay issues and improve the performance [28], [29], [30]. However, the designers need to be aware of the delay criteria while inserting the buffers. When the buffers or repeaters are inserted, the drive strength of the buffer cells, routing, the interconnect distance and metal layers properties play the crucial role in the interconnect delay calculations. If there are any changes in the metal layers width or routing, then the interconnect delays would be affected. If the buffer is inserted at a farther distance, then the routing distance might dominate the interconnect delay, and it could negatively impact the delay. Therefore, the user should be aware of the maximum repeater distance while inserting the buffers, so that it wouldn't adversely affect the interconnect delay. We have developed a tool, which automatically calculates the maximum repeater distance without violating the slope delay criteria and also updates when there are changes in the metal

layers. The tool also calculates the number of repeaters required when the user distance is provided.

3.2 Background

Figure 3.1 shows the typical buffer or repeater modeling structure. As we could see from the Figure 3.1, there exist two cells which are connected by a wire 'w,' the length of the wire is considered as ' l .' The first cell is referred to as the driver cell and the second cell as the receiver cell. The RC delay of the wire depends on the wire length ' l .' RC delay of the wire will increase proportionally to the square of the wire length ' l .'

Figure 3. 1 A simple circuit containing two cells as driver and receiver

We know that the resistance and capacitance values depend on the wire length ' l ,' and these values increase with wire length ' l .' Therefore, the RC delay increases with ' l^2 '. One way of reducing delay is by splitting the wire into different segments and inserting buffers or repeaters to drive the delay actively. By this method, the RC delay will be reduced with minor increment in gate delay, as new buffers are added. The overall delay will increase linearly with respect to the wire length ' l .' Figure 3.2 demonstrates the above technique.

Figure 3. 2 Repeater or buffer insertions design

Generally, inverters are used as the repeater cells, as the inverter combination gives the best performance. When the repeaters are added, there would be a slight increase in the delay and overall design area. If the distance between the repeaters is too high, then it adds more wire (RC) delay and otherwise, if the distance between these repeaters is small, then the delay will be dominated by the inverters size or area. Therefore, these two things should be kept in mind while inserting the repeaters to obtain the best performance. Equivalent circuit model for the one segment of the repeated wire is shown in Figure 3.3. As seen in the figure, the resistances and capacitances are reduced as the length of the wire is reduced.

Figure 3. 3 Wire delay model of the circuit

Referring to *Weste and Harris* book [31], Using Elmore delay method, assuming l as length of the wire, N as number of segments, gate capacitance C , Resistance R , diffusion capacitance Cp_{inv} , wire resistance R_w , wire capacitance C_w and repeater size as unit size of W times, the delay of the repeated wire is calculated as below [31].

$$t_{pd} = N \left[\frac{R}{W} \left(C_w \frac{l}{N} + CW(1 + P_{inv}) \right) + R_w \frac{l}{N} \left(\frac{C_w}{2} \frac{l}{N} + CW \right) \right]$$

By performing the differentiation operation on the above equation with respect to N and W , gives the best length of the wire between the repeaters or buffers.

3.3 Automated Repeater Analysis Tool

As we have seen, to obtain the optimal performance while inserting the repeaters, negotiations of the distance and device size should be made. As greater repeater distance causes the increase in the RC delay and the larger repeater cell causes overall gate delay to increase, attention should be paid to these criteria while inserting the repeaters. We have introduced a tool which will calculate the maximum repeater distance without violating the delay criteria for various process constraints like width, spacing, and metal layers. The tool also calculates the number of repeaters required when the user given distance is provided and can be integrated into the Synopsys IC Compiler™ tool.

Figure 3. 4 Repeater model with attacker cells

As seen from Figure 3.4, the distance between the driver and repeater cell is calculated and referred to as repeater distance. For more realistic conditions, the crosstalk effect is modeled. As we could see, the attacker cells are placed above and below the repeater cell to model crosstalk. The coupling capacitance is modeled by placing the wire very near to each other. The main idea is to calculate the distance between the driver cell and repeater

cell. The routing length calculated between these cells is considered as the final repeater distance.

In the beginning, the repeater cell type, location, and metal layer type are provided. Given repeater cell with the appropriate drive strength is placed initially at the respective location. Delay condition is given as input to the tool. To model the design more realistically we have considered the crosstalk and coupling capacitance effects. Attacker cells are placed above and below the repeater cells.

Once the repeater cell is placed, routing is performed using the mentioned metal layers. After routing is done, the parasitic extraction is performed, and timing information is updated. The calculated delay value of the wire after inserting the repeater is compared with the given delay value. If the calculated value is less than the specified value, then the repeater cell is moved to the next location. Once again routing, parasitic extraction and delay calculations steps are performed. After comparing the calculated delay value with the specified value, if the condition is satisfied, the above steps are repeated again until the delay condition has violated. The final routing distance without violating the delay criteria is considered and taken as the final repeater distance. Design rule checking (DRC) condition is considered. While moving the cell or routing operation, if there is any DRC violation then routing is stopped and the repeater distance until that location is considered, and tool points as DRC violations have occurred.

Once different metal layers or repeater cells with drive strength are provided, the above steps are performed, and final repeater distance is computed for the respective metal layer

and drive strength. The idea is, if there are updates in the process file like routing layer, width, the maximum repeater distance can be calculated conveniently.

Figure 3.5 shows the flow of the maximum repeater distance calculation method. In the repeater distance calculation algorithm, the location of the repeater cell is constantly changed and also the routing, DRC and delay calculation is performed continuously. If there is no DRC and delay criteria violations, repeater cell location is continuously changed until there are any DRC or the delay condition is violated, and final repeater distance is computed.

Figure 3. 5 Flowchart for calculating the maximum repeater distance

Once initial cell location, delay criteria, and metal layer data are given to the IC Compiler™, the repeater cell will be placed at the respective location and routing, parasitic extraction would be performed. After the parasitic extraction is performed, the delay is calculated for the newly placed repeater. Now the cell location is changed, and the above steps are performed until the core boundary area has been reached, or if there are any DRC violation or delay condition violation, the operation is stopped.

Figure 3. 6 Flowchart for calculating the number of repeaters required

Figure 3.6 shows the flow for calculating the number of repeaters required to travel distance given by the user. Now when the user provides repeater distance, the tool can calculate the total number of repeaters necessary for the given distance. If the cycle time is violated while inserting the combinational repeaters, the tool inserts a sequential repeater. In this case, a simple flop is considered as the sequential repeater. The final distance value is calculated, and output data is written to a comma separated file (CSV) file.

```
icc_shell> repeater_analysis -help

Usage: repeater_analysis  # Calculates maximum repeater distance and number of repeaters required
for given delay condition and distances

[-width <-width>]
    (Optional, Specify the width of the metal, Default is taken as minimum width)
-metal <-metal>      (Required,Specify the metal layer to be used)
[-buf_lib <-buf_lib>]  ( Required, Specify the buffer libray cell to be used as repeater)
[-write_csv <-write_csv]
    (Optional, Writes the calculated data into CSV files, Default is 'false', Enter 'true' to enable)
[-distance <-distance>]
    (Optional, Specify the distance (micron) to find out number of repeaters required to travel)
[-net_name <-net_name>]
    (Optional, Specify the repeater net name of the repeater, default name is 'repeater_net')
[-cell_name <-cell_name>]
    (Optional, Specify the repeater cell name of the repeater, default name is 'repeater_cell')
icc_shell>
```

Figure 3. 7 Repeater Analysis Tool Usage Interface

Figure 3.7 shows the user interface window for repeater automation tool. As seen from the figure user can specify metal width using ‘-width’ option, if nothing is specified, then the default width is considered. The user also has options to specify repeater cell name and repeater cell net name using ‘-net_name’ and ‘-cell_name’ options. The ‘-write_csv’ command writes the analysis data into a CSV file. When ‘-distance’ option is selected, the

tool performs the number of repeaters calculation for the given user distance. The user also has options to specify metal layer and buffer library to be used as a repeater cell.

3.4 Experimental Results

The repeater analysis automation tool is developed using the Tool Command Language (TCL) running in the Synopsys IC Compiler™ (ICC) shell environment. All the experiment steps are performed on a Linux based machine running the RedHat Linux. Synopsys Design Compiler™ 2012 and IC Compiler™ 2016 version are used in this experiment. A sample RTL design consists of a few inverters, and a few flip-flops are modeled using the Verilog HDL. Then the developed RTL design is synthesized using the Synopsys Design Compiler™. Synopsys education technology library is used for synthesis and auto place and route (APR). The synthesized netlist from the Design Compiler™ (DC) and other required input files are provided to the Synopsys IC Compiler™.

Floorplan, placement, pre-route, clock tree synthesis (CTS) and routing are done using the IC Compiler™ tool. After the routing has performed, the attacker cells are placed nearer to the driver cell to model the crosstalk and coupling capacitance. After specifying the required inputs correctly, the developed script is provided to the ICC. The user can specify the metal layers, metal width, repeater library cell type, user distance and delay condition to be used inside the script. The tool can write repeater analysis data into a Comma Separated File (CSV) file. Once the repeater analysis data is calculated, final delay information for the path is noted. The post-layout netlist and extracted parasitic files are given to the Synopsys PrimeTime™ for further analysis and SPICE simulation. Once these files are given to the PrimeTime™, the required SPICE simulation library files are also

given to the PrimeTime™. After all the necessary files are set in the PrimeTime™, the spice netlist for the given path is generated using the ‘write_spice_deck’ command. Output SPICE files from the PrimeTime™ are further fed into the Synopsys H-SPICE™, and the SPICE simulation is performed for the path. Then the SPICE simulation delay data is compared with the IC Compiler™ data and analyzed. Above procedures are followed for various metal layers, widths, and distances. Finally, the obtained results are compared and analyzed for multiple input constraints.

Figure 3. 8 Layout of the repeater automation design

Figure 3.8 illustrates the layout view of the repeater analysis design for calculating the maximum repeater distance. As we could see from the figure, a total of four cells is placed near to the repeater cell. Two cells are positioned above, and two cells are placed below

the repeater cell. This is to model crosstalk effects, and these cells act as attacker cells. The repeater cell is placed in the middle of these cells and the repeater cell position constantly changes until the slope delay condition is violated or if there exist any design or routing related errors.

Figure 3. 9 Schematic of the repeater analysis design for distance calculation

Figure 3.9 shows the schematic of the repeater analysis design for calculating maximum repeater distance and Figure 3.10 shows the schematic of the design for calculating the number of repeaters required to meet user given distance condition. As seen from Figure 3.8, the highlighted cell is the repeater cell, and net connected to the input of the repeater cell is referred to as the repeater net, and the maximum distance is calculated for this net. AS we could see from Figure 3.10, the highlighted cells are the repeaters inserted in the

path, when the distance condition is given. In this case, there is a total of ten repeater cells required to reach the user-specified distance for the given delay condition.

Figure 3. 10 Schematic of the repeater analysis design for calculating repeater counts

Table 8 shows the calculated repeater distances for different metal layers and the transient time comparison between the ICC® and SPICE simulations. The metal width for all the metal layer is taken as the minimum width 0.056 unit here, and the BUFFX2 is used as the library cell for the repeater cell here. BUFFX2 has a drive strength of two.

Width = Minimum Width					
Metal Layers	Calculated Repeater Distance	Repeater Cell Lib	Transition Time ICC	Transition Time SPICE	Difference (%)
M2	377.52	BUFFX2	0.1019	0.1643	37.9 %
M3	387.30	BUFFX2	0.1010	0.1670	39.5 %
M5	392.28	BUFFX2	0.1010	0.1675	39.7 %
M7	394.13	BUFFX2	0.1012	0.1673	39.5 %
M8	397.18	BUFFX2	0.1014	0.1684	39.7 %

Table 8 Calculated repeater distance table for the minimum width

From the table 8 metal layers refer to the metal layers used for routing, here a various combination of metal layers are considered for calculation. The second column shows the

obtained repeater distance results in distance unit for the respective metal layers and repeater cell. The third column consists of the combinations of various repeater cells with different drive strength. Last three columns show the transient time values comparison for the repeater cell between the ICC® and SPICE simulations. The transient value calculated by the ICC is 0.1019 unit, while SPICE simulated value is 0.1643 unit for metal M2 using the BUFFX2 cell.

Figure 3. 11 Comparison of the repeater distances of metal layers when the minimum metal width is used

Figure 3.11 shows the repeater distance calculation for various metal widths. As seen from the figure, the repeater distance increases gradually when the metal layer is increased or when the higher metal layer is used for the routing. At the higher metal layer, the resistance and capacitance values are improved. As a result, the repeater distance increases.

Metal Width = 4X					
Metal Layers	Calculated Repeater Distance (Micron)	Repeater Cell Strength	Transition Time ICC	Transition Time SPICE	Difference (%)
M2	297.39	BUFFX2	0.1012	0.1673	39.5 %
M3	307.17	BUFFX2	0.1009	0.1660	39.2 %
M5	312.21	BUFFX2	0.1008	0.1668	39.5 %
M7	317.17	BUFFX2	0.1020	0.1688	39.5 %
M8	317.17	BUFFX2	0.1014	0.1682	39.7 %

Table 9 Calculated repeater distance table for width 4X using the BUFFX2 cell

Figure 3. 12 Comparison of the repeater distance calculation for different metal layers when width 4X and BUFFX2 cell used

Table 9 shows the calculated repeater distances when the width 4X, i.e., width is increased to 4 times the minimum width. As seen from Table 9, here same library cell BUFFX2 is used for the analysis and has a drive strength of two. As seen in Figure 3.12, the width of the metal increased, the repeater distances are decreased compared to the Table 9 data. When the width of the metal is increased, the capacitance values tend to increase. As a

result, repeater distances are decreased. We observed from our analysis that the capacitance values were dominance in the delay calculation.

WIDTH = Minimum Width					
Metal Layers	Calculated Repeater Distance	Repeater Cell Lib	Transition Time ICC	Transition Time SPICE	Difference (%)
M2	382.51	BUFFX4	0.1015	0.1789	43.2 %
M3	392.45	BUFFX4	0.1018	0.1803	43.5 %
M5	397.33	BUFFX4	0.1017	0.1807	43.7 %
M7	397.58	BUFFX4	0.1012	0.1800	43.7 %
M8	397.83	BUFFX4	0.1012	0.1799	43.7 %

Table 10 Calculated repeater distance table for the minimum width using the BUFFX4 cell

Figure 3. 13 Comparison of repeater distance calculations for different metal layers when minimum width and BUFFX4 cell

Table 10, Table 11 and Table 12 show the repeater distance calculations for different metal layers and metal widths using the BUFFX4 as library cell. BUFFX4 cell has a drive

strength of 4, in contrast to BUFFX2 which has a drive strength of 2. The width of 3X is referred to as the metal width increased by three times of the minimum width. As we could see from the Figure 3.13, 3.14 and 3.15 the, repeater distance increases when the higher metal layers are used. This is due to the improvement in the resistance and capacitance values. However, when the width is increased, the distance values are decreased, this is due to the dominance of capacitance values in the higher width metal in our experiment.

WIDTH = 3X					
Metal Layers	Calculated Repeater Distance	Repeater Cell Drive Strength	Transition Time ICC	Transition Time SPICE	Difference (%)
M2	347.37	BUFFX4	0.1015	0.1879	45.9 %
M3	357.35	BUFFX4	0.1014	0.1894	46.4 %
M5	362.35	BUFFX4	0.1011	0.1900	46.7 %
M7	367.36	BUFFX4	0.1016	0.1913	46.8 %
M8	367.56	BUFFX4	0.1016	0.1913	46.8 %

Table 11 Calculated repeater distance table for the width 3X using BUFFX4 cell

Figure 3. 14 Comparison of the repeater distance calculations for the different metal layers with width 3X and BUFFX4 cell

WIDTH = 4X Micron					
Metal Layers	Calculated Repeater Distance	Repeater Cell Lib	Transition Time ICC	Transition Time SPICE	Difference (%)
M2	297.52	BUFFX4	0.1016	0.1750	41.94 %
M3	307.33	BUFFX4	0.1013	0.1766	42.63 %
M5	312.33	BUFFX4	0.1012	0.1774	42.95 %
M7	317.34	BUFFX4	0.1020	0.1790	43.01 %
M8	317.57	BUFFX4	0.1019	0.1790	43.07 %

Table 12 Calculated repeater distance table for width 4X using the BUFFX4 cell

Figure 3. 15 Comparison of the repeater distance calculations for different metal layers when the width 4X and BUFFX4 cell used

Table 13 shows the repeater analysis results when user given distance is provided. The tool calculates the number of repeaters required when the distance is provided. As seen from the table, for covering 4800-unit distance, the total calculated number of repeaters is 13.

For this distance, total 12 combinational repeaters required and one sequential repeater is required. As it would violate the cycle time condition, the sequential repeater is used.

Width = Minimum Width						
Metal	User Given Distance	Combinational Repeater Distance	Combinational Repeater Delay	Combinational Repeaters Required	Sequential Repeaters Required	Total Repeaters Required
M8	4800	397	0.107	12	1	13
M8	4200	397	0.107	10	1	11
M8	3600	397	0.107	10	0	10
M8	3000	397	0.107	8	0	8

Table 13 Number of repeaters required calculated by the tool, when various distance values are provided for metal layer M8 with the minimum width

As seen from the Table 13, when distance values are decreased, the number of repeater counts are also decreased. To cover 4200-unit distance, we require total 11 repeaters, out of which ten are combinational and one sequential repeater. Similarly, to cover 3600-unit distance, the calculated number of repeaters are 10 in total. We don't need any sequential repeaters here, as cycle time is not violated.

Width = 4X						
Metal	User Given Distance	Combinational Repeater Distance	Combinational Repeater Delay	Combinational Repeaters Required	Sequential Repeaters Required	Total Repeaters Required
M2	4800	298	0.102	16	1	17
M2	4200	298	0.102	14	1	15
M2	3600	298	0.102	11	1	12
M2	3000	298	0.102	10	1	11

Table 14 Number of repeaters required calculated by the tool, when various distance values are provided for metal layer M2, Width 4X

As seen from Table 14, we could observe that when maximum distance covered by a single combinational repeater is decreased from 397-unit distance to 298-unit distance, the total number repeaters required to cover the given distances are increased. Comparing Table 13 and Table 14 data, to cover 4800-unit distance, using metal M2 and width 4X, which is four times of minimum width, we would need a total of 17 repeaters. If we use metal M8 instead of M2 and reduce width to minimum width, we would only require 13 repeaters to cover the distance.

Chapter 4: Summary and Future Work

This work describes two simple and convenient automation tools, variation analysis tool and repeater automation tool. These tools can be used in the integrated circuits design to automate variation analysis modeling and repeater analysis in the physical design flow to reduce the manual effort.

In Chapter 2 we discussed background related to process variations in the timing analysis phase and also the proposed variation analysis automation techniques. We also tested the proposed variation analysis tool by considering two different designs. Proposed variation analysis tool results have been compared with the Synopsys PrimeTime™ results by considering three different timing paths from two different designs, and the results show above 98% accuracy compared to the PrimeTime™ results.

Chapter 3 gives the background related to repeater analysis and automation process in the IC design flow. We also discussed our proposed method for automating the repeater analysis process. The proposed repeater automation tool can calculate the best possible repeater distance for any given metal layer. It also calculates the number of repeaters required for the user given distance and frequency. Analysis has been done using the repeater analysis tool with various metal layers and width. We also analyzed repeater distances based on these process constraints. The accuracy of this script is compared with the repeater insertion based on synthesis tools and also, the SPICE simulation.

In future, it would be interesting to model and test the variation analysis tool in a design which has common path pessimism problem and the repeater analysis tool can be tested in design with different shielding options and crosstalk effects into the considerations.

References

- [1] G.E. Moore, “Cramming More Components onto Integrated Circuits,” In Proceedings of IEEE, vol. 86, pp. 82–85, 1998.
- [2] S. Nassif, “Delay variability: sources, impacts and trends,” in Proceedings of IEEE International Solid-State Circuits Conference. Digest of Technical Papers (Cat. No.00CH37056), pp. 368–369, 2000.
- [3] P. Gupta, “Toward a systematic-variation aware timing methodology.” In: Proceedings of the IEEE design automation conference, pp 321–326, July 2004.
- [4] E.E. Rabab, M. Ali, and H. Hamed, “Analysis and Design of Networks-on-Chip under High Process Variation,” Springer [ISBN 978-3-319-25764-8], PP 57-67, 2015.
- [5] S.S. Sapatnekar, “Timing,” Kluwer Academic Publishers, Boston, MA, 2004.
- [6] D. Boning and J. Cliung, “Statistical metrology - measurement and modeling of variation for advanced process development and design rule generation,” in Proceedings of International Conference on Characterization and Methodology for ULSI Technology, pp. 395-404, March 1998.
- [7] J. Watts, N. Lu, C. Bittner et al. “Modeling FET variation within a chip as a function of circuit design and layout choices.” In: Proceedings of the nanotech workshop on compact modeling, pp 87–92, 2005.
- [8] N. Maheshwari and S.S. Sapatnekar, “Timing Analysis and Optimization of Sequential Circuits,” Kluwer Academic Publishers, 1999.
- [9] J. Bhasker and R. Chadha, “Static Timing Analysis for Nanometer Designs: A Practical Approach,” Springer [ISBN 978-0-387-93819-6], 2009.

- [10] D. Boning and J. Cliung, "Statistical metrology - measurement, and modeling of variation." International Conference on Characterization and Metrology for ULSI Technology, NIST, Gaithersburg, March 23-27, 1998.
- [11] D.M. Andrade, A. Calomarde, and JA. Rubio, "A comprehensive compensation technique for process variations and environmental fluctuations in digital integrated circuits," In Proceedings of the IEEE international Midwest symposium on circuits and systems, Aug. 2010.
- [12] A. Agarwal, D. Blaauw, and V. Zolotov, "Statistical timing analysis for intra-die process variations with spatial correlations," International Conference on Computer Aided Design, 2007.
- [13] J. Jess, K. Kalafala, S. Naidu, R. Otten, and C. Visweswariah, "Statistical timing for parametric yield prediction of digital integrated circuits," in Proceedings of DAC, pp. 932–937, 2003.
- [14] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equation of state calculations by fast computing machines," J. Chem. Phys., vol. 21, no. 6, pp. 1087–1092, Jun. 1953.
- [15] D. Blaauw, K. Chopra, A. Srivastava, and L. Scheffer, "Statistical timing analysis: From basic principles to state of the art," In Proceedings of IEEE Trans. Computer. - Aided Design Integer. Circuits Syst., vol. 27, no. 4, pp. 589–607, Apr. 2008.
- [16] A. Agarwal, D. Blaauw, V. Zolotov, S. Sundareswaran, M. Zhao, K. Gala, and R. Panda, "Statistical delay computation considering spatial correlations," Design

- Automation Conference, 2003. In Proceedings of the ASP-DAC 2003. Asia and South Pacific, pp. 271 – 276, Jan. 2003.
- [17] C. Amin, N. Menezes, K. Killpack, F. Dartu, U. Choudhury, N. Hakim, and Y. Ismail, “Statistical static timing analysis: How simple can we get?” In Proceedings of Design Automation Conference, pp. 652–657, 2005.
- [18] S. Walia, “PrimeTime® Advanced OCV Technology,” April 2009.
- [19] B. Bautz and S. Lokanadham, “A Slew/Load-Dependent Approach to Single-variable Statistical Delay Modeling,” In proceedings of TAU Workshop, 2014.
- [20] Incentia Design Systems Inc, “Advanced On-chip-variation Timing Analysis for Nanometer Designs,” www.incentia.com.
- [21] Synopsys Inc, “PrimeTime® User Guide 2016”, 2016.
- [22] P.S. Fuang and N.M. Mahyuddin, “Implementation Study of Path-Based AOCV Model on Pessimism Reduction for 20nm Technology”, IEEE International Conference on Control System, Computing and Engineering, pp. 28 - 30 November 2014.
- [23] B. Daniel, “Oh, The pain of timing closure at 20nm”, Electronic Design Europe, 2010.
- [24] G. Petrosyan, S. Abovyan, and T. Harutyunyan, “Modelling on-chip variations in digital circuits using statistical timing analysis,” In proceedings of Design Test Symposium, pp. 37-39, 2010.
- [25] Synopsys Inc, “Parametric On-Chip Variation (POCV) Application Note”, Feb. 2017.
- [26] J. Lillis, C. K. Cheng, and T. T. Y. Lin, “Optimal wire sizing and buffer insertion for low power and a generalized delay model,” in Proceedings of ICCAD, pp. 138-143, 1995.

- [27] C. C. N. Chu and D. F. Wong, "A new approach to simultaneous buffer insertion and wire sizing," in Proceedings of ICCAD, pp. 614-621, 1997.
- [28] C. J. Alpert, M. Hrkic, and S. T. Quay. "A fast algorithm for identifying good buffer insertion candidate locations," in Proceedings of ISPD, pp. 47-52, 2004.
- [29] L. P. P. P. van Ginneken, "Buffer placement in distributed RC-tree networks for minimal Elmore delay," in Proceedings of IEEE ISCAS, pp. 865-868, 1990.
- [30] A.B. Kahg, S. Muddu, E. Sarto, and R. Sharma. "Interconnect tuning strategies for high-performance ICs," in Proceedings of DATE, pp. 471-478, 1998.
- [31] N. Weste and D. Harris, "CMOS VLSI Design: A Circuits and Systems Perspective," Addison-Wesley [ISBN:978-0321547743], pp. 211-237, March 2012.