Smart and Efficient Multi-scenario SoC Timing Closure and ECO Generator

by

Aditi Sharma

Under the Supervision of
Dr. Sujay Deb
Rangarajan Ramanujam

Indraprastha Institute of Information Technology Delhi
June, 2015

Smart and Efficient Multi-scenario SoC Timing Closure and ECO Generator

by

Aditi Sharma

Submitted
in partial fulfillment of the requirements for the degree of
Master of Technology in Electronics & Communication
Engineering with specialization in VLSI & Embedded Systems

to

Indraprastha Institute of Information Technology Delhi
June, 2015

Abstract

Signoff timing analysis is still considered as a critical element in the SoC design flow. With the advancements of the leading edge technologies towards the deep sub-micron realms, the performance of a multi-million transistor SoC is challenged by the aggravation of process variations. Traditionally, the performance of an integrated circuit (IC) is characterized by the clock frequency at which it operates, and is evaluated considering only worst-case gate delays. This is a pessimistic approach, which highly underestimates the performance of the design in deep sub-micron. This pessimism drives the engineers to further optimize the design, eventually increasing the design cost drastically, as analysing such a design would require consumption of more resources, adding to increase in the number of iterations. With the aggravation of process variations, the main contributor to the delay of the circuit is dominated by the interconnect delays. Hence, the number of scenarios required to analyse and fix the design is enormous and ever increasing. Such a pessimism hampers with time-to-market window of product, which in worst case may be missed too.

In world of large number of multi-scenarios, the engineering change order (ECO) becomes inevitable to achieve design timing closure with a low respin cost. This concern is attributed to the fact that, firstly, performing ECOs is an experimental process requiring an expert for each mode. The involvement of huge manual effort utilising their own respective scripts leads to significant amount of iterations to achieve its final goal. Secondly, timing violations under multi -scenario design exists even after the detailed routing. In the absence of such an implementation system, that can handle fixing of violations in multi-scenario design, without the involvement of manual effort, causes non-convergence of timing closure, making the design prone to the ping-pong effect. This becomes a bottleneck to the design cycle time. Additionally, this approach requires large memory usage and licensed CPU resources which are already very limited.

This work presents a hierarchical approach to target the fixing of timing violations, gaining significant cycle time in overall timing closure. A complete solution for true hierarchical timing and crosstalk delay signoff is presented in this work. It explores the gaps in the current approach and provides a

novel solution to address them. It discusses the smart and efficient Timing Closure Methodology which first identifies the Dominant Scenarios and then prioritizes the violations for fixing, in a manner which eliminates the danger of ping-pong effect, where fixing one set of violations can create hundreds of new violations. This iterative approach seamlessly integrates the steps of timing closure thereby reducing 90% of the manual effort.

Certificate

This is to certify that the thesis titled **Smart and Efficient Multi-scenario SoC Timing Closure and ECO Generator** being submitted by Aditi Sharma to the Indraprastha Institute of Information Technology Delhi, for the award of the Master of Technology, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree/diploma.

June, 2015

Dr. Sujay Deb
Department of _____
Indraprastha Institute of Information Technology Delhi
New Delhi 110 020

June, 2015

Rakesh Gulati
Consumer Product Division, ST Microelectronics
Greater Noida,201308

June, 2015

Rangarajan Ramanujam
Consumer Product Division, ST Microelectronics
Greater Noida,201308

Acknowledgements

This thesis would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost, my sincere gratitude to *Dr. Sujay Deb*, Assistant Professor, IIIT Delhi, for his continuous support, motivation, immense knowledge. His guidance has helped me in all the time of research and writing of the thesis.

Rangarajan Ramanujam, Group Manager, CPD, ST Microelectronics, for accepting me as an intern in his team. I am grateful to him for his expertise, kind concern and consideration regarding my academic requirements.

Rakesh Gulati, Staff Engineer, CPD, ST Mircoelectronics, for his patience and steadfast encouragement, his constructive criticisms at different stages of my work were thought-provoking and they helped me focus on my ideas.

I am deeply grateful to *Mohita Batra*, Senior Design Engineer, CPD, ST Microelectronics, for the long discussions that helped me in understanding the technical details of my work. I am also thankful to her for her unfailing support as my thesis adviser and for having faith in me.

Apurva Chaure, Staff Engineer, CPD, ST Microelectronics, for sharing his valuable insights at each stage of this work and for being accommodating to my queries.

Last but not the least, my family and dear friends. None of this would have been possible without their love. And the one above all of us, the omnipresent God for giving me the strength to plod on, thank you so much everyone.

Contents

List of Figures

List of Tables

List of Abbreviations

ASIC Application-Specific Integrated Circuit
BEOL Back End Of Line
Cmax Maximum interconnect capacitance
Cmin Minimum interconnect capacitance
CMOS Complementary MetalOxideSemiconductor
CTS Clock Tree Synthesis
DMSA Distributed multi-scenario analysis
ECO Engineering Change Order
FEOL Front End Of Line
FF Fast nmos fast pmos
FS fast nmos slow pmos
IC Integrated Circuit
IP Intellectual Property
LVT Low Threshold Voltage
MOS MOSFET
NMOS n-channel MOSFET
NVP Number of Violating Points
OCV On-chip Variations
PMOS p-channel MOSFET
RC Resistance Capacitance
RCmax Maximum interconnect resistance capacitance
RCmin Minimum interconnect resistance capacitance
RTL Register Transfer Language
SF slow nmos fast pmos
SoC System on Chip
SS slow nmos slow pmos
STA Static Timing Analysis
TT typical nmos typical pmos
WNS Worst Negative Slack

Chapter 1

Introduction

Designing of a System-on-Chip (System on Chip (SoC)) involves integration of intellectual properties (Intellectual Property (IP)) on the chip in a hierarchical manner. As predicted by the Moore's law, the number of transistors integrated on a die would double every 18 months. With this exponentially growing number of transistors along with the decreasing feature size, the design complexities increase manifold. This advent of technology has shifted the design paradigm from logic dominated to interconnect dominated, wherein the delay due to the parasitic effects of the nearby interconnects predominates the gate delay. This, along with the increase in clock rates pose big challenges for timing and crosstalk delay signoff. It has been observed that accurate timing closure can consume upto 60% of the design time [2] becoming a critical step in signoff flows of all semiconductor companies. Also, existence of multiple corners due to variations in manufacturability, timing closure becomes all the more very complex. If the design under such circumstances is not closed properly, defects occurs in later design stage or design change orders are issued, and the chip needs to modified by photomask changes. This incurs huge expenses.

Even though designers make great effort in improving the timing closure in the physical design stages, however they still face heavy timing violations in the post routing stages too. Analysing and fixing these violations over multi scenarios requires manual intervention causing an increase in the number of iterations as shown in Figure 1.1, creating delays in timing closure. This also leads to emergence of the ping-pong effect, which is fixing a set of violations in one scenario, creates other types of violations in other scenarios. For the entire design convergence and timing closure, especially under current multi-corner multi-mode design, some more efficient methods need to be introduced in order to improve the timing closure in the last mile to timing signoff.

The major challenges faced by the timing signoff today can be summarised as below:

1. The three key issues are: Speed, capacity, closure with respect to fixing violations (Engineering Change Order (ECO)).

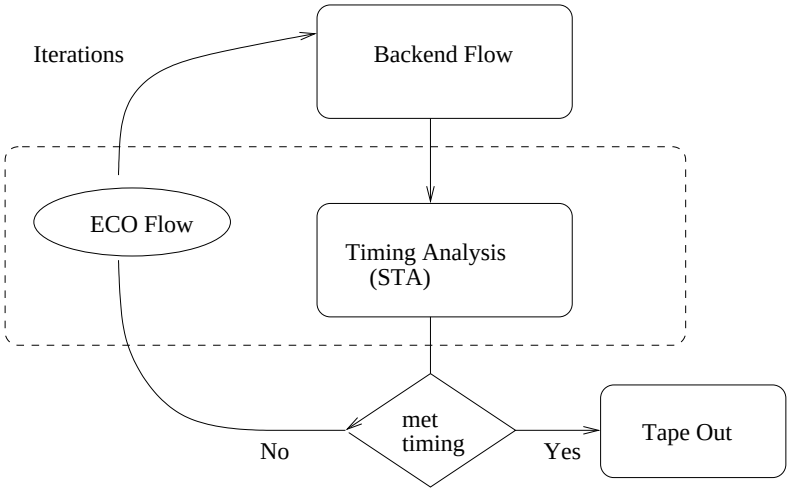


Figure 1.1: Steps of SoC design flow

2. Increasing design variations from foundry perspective, increases the number of views to be considered for analysing and fixing, accurately fixing in the right context to prevent creation of new violations.

Thus, with emergence of each new technology, a holistic approach is required which takes into the account multitude of variation issues to achieve accuracy without compromising with the performance. An approach which can fix the violations at signoff with less respins and lesser cost too. This work addresses the above mentioned dire needs by providing deterministic automated solutions improving the cycletime before reaching decent yield.

To appreciate the work, following section delves further, focussing on the caveats in timing closure today.

1.1 Timing Closure And Its Caveats

Timing closure is the process of efficiently modifying the design in order to meet the optimization and timing constraints satisfying the insatiable need for speed of the SoC. The various steps of an SoC design flow from the netlist to tape-out includes Register Transfer Language (RTL) synthesis, floorplanning, routing, timing analysis, (ECO), signoff.

This work primarily focuses on the ECO Timing Closure which is primarily an exhaustive process of fixing the violations. The pressure of achieving timing closure with significant cycle time reduction is a major challenge designers are facing today.

Following subsections address the dominant concerns, needed to be tackled for improvement in ECO timing closure thereby improving the time to tape-out.

1.1.0.1 Poor Convergence

Every new process technology node introduces variations in the design parameters in terms of process, voltage and temperature, collectively called as a *corner* (PVT). The effect of these variations impact the overall performance of the chip and if not addressed, can render the system inoperable.

As the number of corners and thus scenarios (number of modes * number of corners) keeps increasing, the number of violations also increases, as it may happen that a path maybe setup critical in one corner and hold critical in any other. Thus analysing and fixing the design on all scenarios impedes the timing closure process, making the convergence difficult.

1.1.0.2 Limited Resources

In the current semiconductor market, the signoff tools have high license fees and long runtime. Owing to cost and budget constraints, the resources are limited. This hampers the delivery of complex designs in shorter time frames.

1.1.0.3 Ping-Pong Effect

With the creation of new corners, new set of violations can be created, making it difficult for the designers to close the design after every stage of violation fixing. This effect, of introducing violations in a single or multiple corner(s) while fixing a set of violations in any other corner, is called the *Ping-Pong Effect*. This effect cannot be predicted, as a result of which, the number of iterations required to completely fix the violations in all corners increases, impacting the ECO timing closure.

1.1.0.4 Runtime

A typical ECO flow handles multiple tools for Static Timing Analysis (STA), place and route, ECO. This multi-tool solution requires intimate knowledge of the process technology and its associated tools. These engines also produce their own custom timing violation reports. The correlation issues stemming from difference between the timing engines along with the growing number of scenarios leads to explosive increase in STA runs. As a result, the violations may not decrease monotonically, additionally requiring more number of iterations.

1.1.0.5 Manual Effort

Currently under multi- scenario design, absence of an efficient implementation system, that can handle the mode and process variations, leads to blind pessimist engineering change order loops. Therefore, there is no systematic approach to fix the violations, its more experimental. Design teams consists of experts working on each mode, manually analysing the functionality and process variations effects. They generate their own scripts to perform ECO, consuming large memory space. As a result, the count of iterations is enormous, and manual effort involved in generating ECO scripts, hand calculations spent in debugging, is overwhelming taking as long as 10 months to tape-out.

1.1.0.6 Sensitive Clock Tree

The clock tree is one of the most sensitive parts of the design. Typically, the skew generated between synchronous clocks is budgeted. If the skew does not lie in the budget range, timing violations could be introduced on some paths, which may go unnoticed. Modifying the clock tree in the later design stage, is expensive, and also affects the timing of the design.

1.1.0.7 Signal Integrity Issue

In the design flow, circuit performance is evaluated on the basis of worst case gate delays. However, with the larger process variations, higher clock rates, and increasing interconnect densities, the effect of parasitic coupling capacitances become dominant factors, causing signal integrity issues. Thus, crosstalk effects result in significant delay and noise variations in data propagation becoming another major concern for delayed schedules and chip failures.

1.2 Inter And Intra Chip Variations

Scaling of Complementary MetalOxideSemiconductor (CMOS) technologies to nanometer range is increasing the device parameter variations, such as oxide thickness, effective channel length, and width [3], affecting the high performance of the designs [4] . Thus, there is a dire need to incorporate these parameter variations not only for analysing but also for fixing. This section provides an overview of parameter variations and its effect on the increase in the number of views required to cover the process space.

Earlier, at large process nodes such as 180nm and 130nm, it was acceptable to perform timing analysis and ECO only at worst(slow) case and fast(best) case for a combination of process, voltage and temperature. The design variability effects were

seldom looked upon. However, in the recent years, the contribution to variability has lead to significant increase in the number of variations not just between the wafers but also across it, as mentioned in [5].

This section explains the reasons of growing number of scenarios (variations) and its effect of timing closure.

1.2.1 Variation Sources

1.2.1.1 Variation in Supply Voltage

Every standard cell in the chip is connected to voltage rails via interconnects of finite resistances. These interconnects may have different lengths resulting in difference in resistance values. Along with this, the inductance effects also come into picture becoming another cause of the voltage drop. This affects the voltage, the standard cell receives. Local effects ,that is on chip variations, of device voltage variation includes IR drop, ground bounce, slew variations due to crosstalk etc. These effects affect the delay characteristics of the device. As the delay of the cell is dependent on the saturation current, voltage drop due to resistance in the supply voltage, inflects the propagation delay of a cell. A higher voltage reduces the propagation delay of the cell, improving its speed.

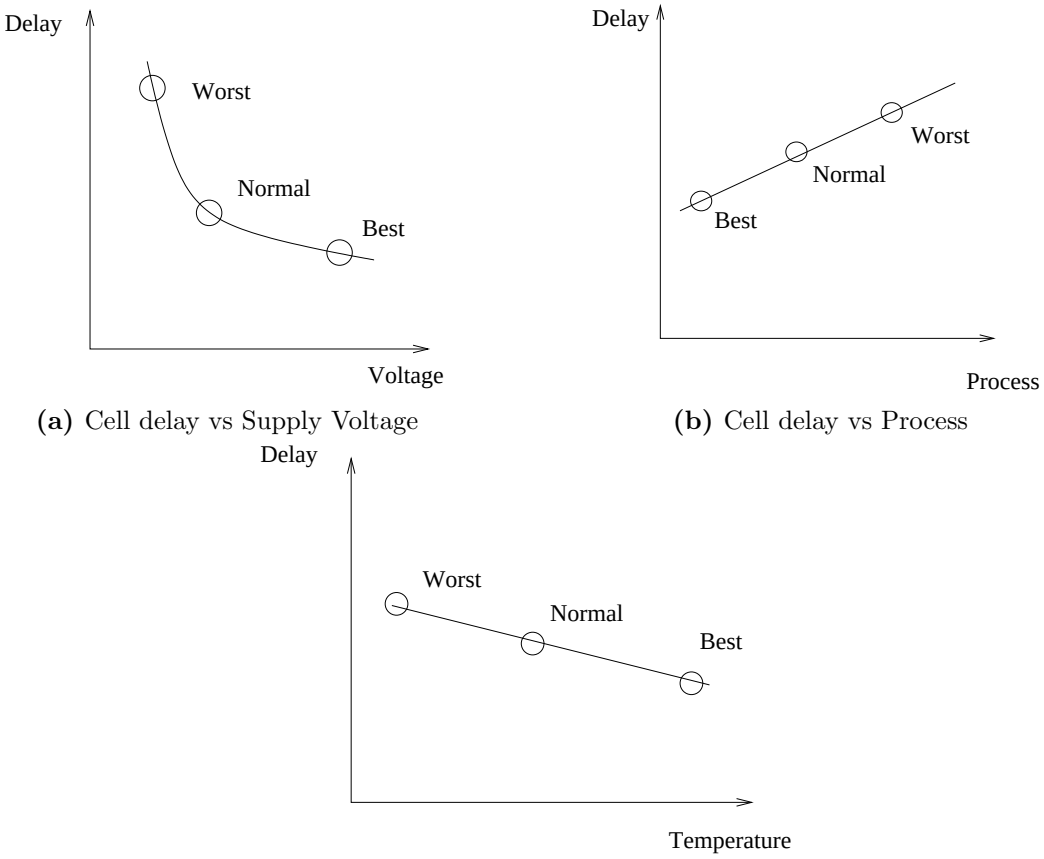
1.2.1.2 Variation in Process

Process variations arise due to deviations in the semiconductor fabrication process. This is because there are millions of transistors packed together on the same chip, and hence geometries of transistors(channel length and width) may vary from one another. Process variations are due to variations in manufacturing conditions such as pressure, dopant concentration. Inter-die variations include geometric and material variations. Geometric variations such as channel length and width variation, oxide thickness variation, result in threshold voltage and leakage variations. On the other hand, intra-die variations include device and interconnect geometry variations. These deviations impact the interconnect parasitics (Resistance Capacitance (RC)) resulting in performance, and signal integrity degradation.

1.2.1.3 Variation in Temperature

Temperature varies continuously across the chip during its operation. This is mainly due to the power dissipation in the MOSFET (MOS) transistors caused by switching, short-circuit and leakage power. This affects the threshold voltage of a transistor. A higher temperature decreases the threshold voltage(temperature inversion) which indeed increases the current of the cell and hence decreases the propagation delay. Also, since some parts of the chip maybe densely packed, creating regions of hot spot.

On the basis of the description provided above, there are three kinds of manufacturing process models. These are slow, fast, typical. As also explained, cell delays are affected by the variations in the manufacturing process and environmental parameters creating best, worst, typical behaviours, as already explained in the previous sections. The variation of trend of cell delay with process, voltage and temperature is shown in Figure 3.3.



(a) Cell delay vs Supply Voltage

(b) Cell delay vs Process

(c) Cell delay vs Temperature

Figure 1.2: Cell delay Trend with PVT variations

1.2.2 What is a Scenario

- **Mode**: A mode is defined by the operational variability of a design such as, a set of clocks, supply voltages, and timing constraints and libraries. A chip can be operated in multiple modes, such as functional mode, test mode, standby modes etc.

- **Corner** : A corner defines the differences due to variations in process, voltage and temperature. These are defined as a set of libraries supplied with the process

kit. There are two types of corners

- Front End Of Line (FEOL) corners: Front end of line is the first phase of manufacturing process where wafer based devices are formed, such as the transistors, which is required to form a fully isolated CMOS elements. Depending on the doping concentrations and carrier mobilities, three types of corners exist, typical,fast and slow. A corner has 2 letter designation, first part corresponding to n-channel MOSFET (NMOS), and second part corresponding to p-channel MOSFET (PMOS). As an example, an Fast nmos fast pmos (FF) corner corresponds to Fast-NMOS Fast PMOS. A total of 5 such corners exist- FF, slow nmos slow pmos (SS), fast nmos slow pmos (FS), slow nmos fast pmos (SF), typical nmos typical pmos (TT). A representation of such is shown in Figure 1.3.

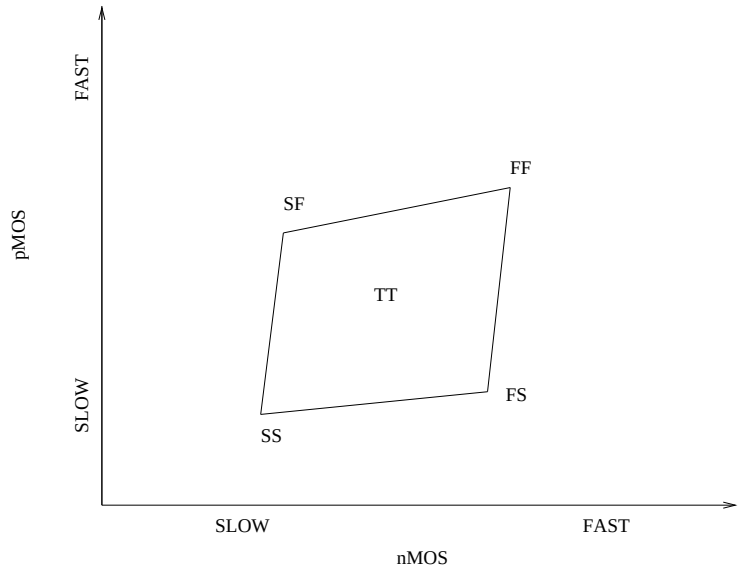


Figure 1.3: Design Corners

- Back End Of Line (BEOL) corners: Back end of line is the last stage of the manufacturing process where metal interconnects are added. Due to On-chip Variations (OCV) as explained in the previous section, the delay due to coupling effects caused by the nearby interconnects play a major role in determining the overall delay. Hence, the interconnect variations from maximum resistance(minimum cross-sectional area) to maximum capacitance(minimum cross-sectional area), accounts for the new set of corners: Maximum interconnect capacitance (Cmax), Minimum interconnect capacitance (Cmin), Typical, Maximum interconnect resistance capacitance (RCmax), Minimum interconnect resistance capacitance (RCmin).

A combination of a mode and a corner is referred as a *Scenario*. The purpose of

scenarios is to ascertain that the device, at any operating condition, meets the timing constraints for its proper functionality.

1.2.3 Effect on Timing Closure

As nanometer geometries move into the 65nm and 45nm realms, the variations of process, temperature, voltage increases, the number of scenarios required for checking the performance of the design also increases, becoming one of the main causes of chip failures and delayed schedules.

The number of scenarios for 28nm, are calculated as follows:

- Voltage: best, worst: 2 corners
- Temperature: best, worst : 2 corners
- Process FEOL: fast-fast, slow-slow, typical : 3 corners
- Process BEOL: Cmax, Cmin, RCmax, RCmin : 4 corners

Therefore, the total number of corners is equivalent to 48. This combined with the number of modes,2 in this case (functional, test), the total number of scenarios reaches 96. Such sky-rocketing number of scenarios makes ECO timing closure increasingly difficult.

1.3 Conceptual Overview

The main focus of this section is to bring forth the main causes of timing violations along with providing a brief overview of the concepts required for a better understanding.

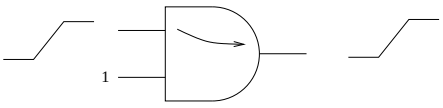
A. Static timing analysis: Timing analysis refers to analysing the design for timing violations. The two methods used for timing analysis are the dynamic timing analysis and STA. The former validates the design by verifying its functionality as well as the timing. This is done by simulating the design for various input stimulus. This process is extremely exhaustive in contrast to static timing analysis, where the design is validated statistically for timing and does not depend on the input stimulus. Thus, for the timing analysis of Application-Specific Integrated Circuit (ASIC), containing millions of gates, STA is a faster and simpler approach. The purpose of STA is to validate if the design can safely operate at rated speed without any timing issues [6], [7]. Following subsections provide a brief overview of few STA concepts.

B. Timing paths: There can be multiple paths through which a logic can propagate to the desired destination point. The point in the design from where the path starts is called a *startpoint*. This is the point from where the data gets launched by a clock or where the data is expected to be available at specific time. A startpoint can either be a clock pin of a sequential element or an input port of

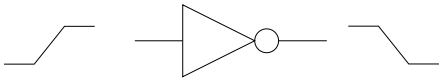
a synchronous design Similarly, the point in the design where the data gets captured by a clock edge or the point where the path ends is called an *endpoint*. An endpoint can either be a register data input or an output port of a synchronous design. Thus, a timing path is defined as a sequence through a design which starts at a valid startpoint, passes through some combinational logic and ends at a valid endpoint. [7] [6].

C. Path groups: For the purpose of ease of analysing the design, all the timing paths are sorted into separate groups, called *Path Groups* based on the same endpoint clock.

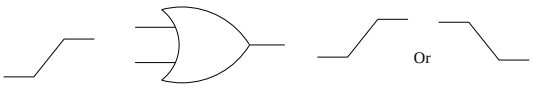
D. Timing Arcs and Unateness: A combinational cell such as an and gate or a NAND gate, may have multiple input pins. Each input pin has its own timing arc describing how the output changes with changes in the input transition. The existence of such multiple timing arcs have their own transitions driving the output transition. Thus, a timing arc is said to have a timing sense. This sense of a timing arc is referred as *Unateness*. There are three types of unateness : *positive unateness*, where a rising/ falling transition at the input pin causes a rising/ falling transition at the output. The second type of unateness is the *negative unateness*, where input and output have different transitions. The third type of unateness is a *non-unateness* timing arc, where the transition behaviour at the output with respect to input cannot be predicted. Examples of three types of unateness are shown in Figure 1.4



(a) Positive unateness



(b) Negative unateness



(c) Non unateness

Figure 1.4: Types of Unateness

E. Common path pessimism: Due to variations of device parameters on chip as

explain in section 1.2, a cell in a path can have two types of delays, min delay for short path considerations and max delay for long path considerations. At times for verifying the correctness of the design, it is required that max delays along one is considered and min delays along the other path. In such a case, if there exists a cell which lies in a path common to both the cells, max delay of this cell is considered for max delay path and min delay of the cell is considered for min delay path. However, a cell cannot have two types of delays at the same time and hence this conflict must be taken into consideration for timing verification. The difference between the delays at a common point is called *clock re-convergence pessimism*.

F. Timing violations due to clock skew: Clock skew is defined as the difference in the arrival times of the clock(s) signal(s) at launch and capture flop across the chip. There are two types of clock skew causing setup and hold violations.

i. *Positive clock skew*:if the clock reaches startpoint flop later than endpoint flip-flop , then data launched at startpoint flip-flop gets extra time equal to skew to reach endpoint flip-flop. However, hold slack suffers. The skew calculated in such a case is called *positive clock skew* and is given by eq.1.2

$$HoldSlack = T_{cq} + T_{comb} - T_{skew} - T_{hold} \tag{1.1}$$

ii. *Negative clock skew* : Similarly, if clock reaches endpoint flip-flop earlier than startpoint flip-flop, the skew is called *negative clock skew*. In this case, launch flop gets lesser time to launch the data, hence setup slack becomes critical, on the other hand, hold slack is relaxed. This is expressed by eq 1.1.

$$SetupSlack = T_{cycle} - T_{setup} - T_{skew} - T_{launch} - T_{combo} \tag{1.2}$$

where, T_{cycle} is the clock period, T_{setup} is the setup time of the capture flop, T_{skew} is the clock skew, T_{launch} is the max time taken by clock in the launch path, T_{combo} is the combinational delay in the data path, T_{hold} is the hold time of the launch flop.

G. Timing violations due to signal integrity issue:With manufacturing and environmental parameter variations cropping up, becoming a major concern to be tackled for eliminating timing related issues, this section deals with the role of crosstalk in signal integrity issues resulting in timing violations.

Crosstalk effects become prominent when global interconnects become thinner and come closer in nanometer designs. The closeness of interconnects induce parasitic coupling capacitances on adjacent wires. This results in switching activity on an interconnect creating undesirable effects on coupled signals. This affects the operation of the chip. Apart from the higher routing density of the chip as one

of reasons for crosstalk effects, frequency of operation and lower supply voltage too play a significant role.

The affected signal is called the victim net and the affecting signal is termed as an aggressor net. The charging of the two capacitances (ground capacitance and coupling capacitance) due to switching activity of the aggressor nets affects the timing of the interconnect. Crosstalk delay effects can be broadly classified by the two following types:

- *Positive crosstalk delta*: In this scenario, the aggressor and victim nets switch in the opposite directions at the same time. So, the charge on the coupling capacitance changes by $(2 * C_c * V_{dd})$. This results in additional delay which is termed as positive crosstalk delta. In other terms, the data now reaches the endpoint later than steady state case. Such a case is considered for hold analysis. Figure 1.5 illustrates the case of positive crosstalk delta when both the nets undergo transitions in the opposite directions at the same time.

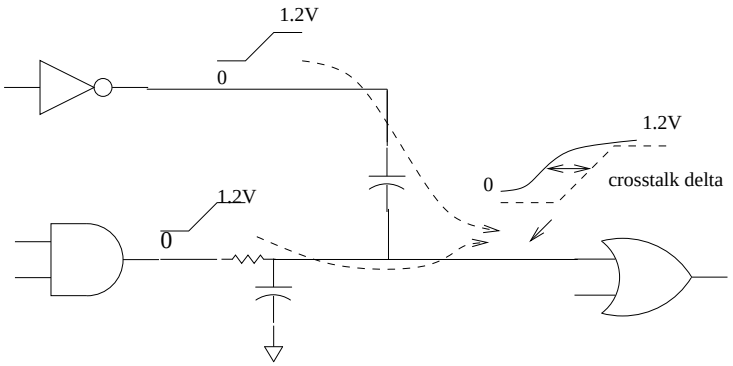


Figure 1.5: An example of positive crosstalk delta

- *Negative crosstalk delta*: In this scenario, the aggressor net switches in the same direction as that of the victim net and at the same time, that is if both the nets have identical slew, the cumulative charge required to be charged by the victim driver cell is less than or equal to $C_g * V_{dd}$. Thus, there is reduction in the delay of the net contributing to the path. In other terms, the data now reaches the endpoint earlier than steady state case. This reduction in delay termed as negative crosstalk delta. Such a case is considered for hold analysis. Figure 1.6 illustrates the case of negative crosstalk delta when both aggressor and the victim undergo rising transition at the same time.

H. ECO: ECO, is an effective technique for fixing circuit functionality and timing problems after the placement stage and routing stage [8], after STA has been performed. It is the process of inserting logic into the designs netlist. One of the methods is to perform gate level netlist ECO, which is a tedious task involving

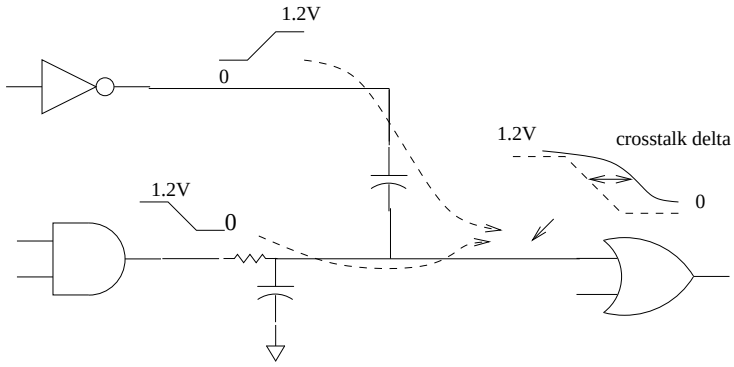


Figure 1.6: An example of negative crosstalk delta

huge manual effort. The need is thus to automate the ECO implementation process in an effective way to reduce burden and modifications to be done in the design stages, placement and routing. This work proposes such a solution which is discussed in the next chapter.

1.4 Motivation and Solution

The current methodology requires implementing the ECOs using multiple tools in the design flow, that can handle multiple modes and multiple corners simultaneously. These tools generate their own reports. Eventually, there exists multiple divergent reports for the design teams to figure out how to fix the violations. The number of iterations involved in this process is far too many, potentially threatening the design through the ping pong effect. This problem of handling of ECOs gets compounded with each new process node. Meeting production deadlines, in such a scenario, becomes a challenging goal. There is a dire need for a more proactive management of timing violations followed by ECOs to reduce the likelihood of chip failure. In absence of such tool, designers face unpredictable iterations during signoff ECO loop. This work aims to circumvent the performance limitations of the traditional flow by proposing a novel solution which significantly reduces the runtime with better ECO coverage. This is achieved by the task of endpoint filtration at each stage. It seamlessly integrates the steps of signoff flow from extracting the worst case scenarios, prioritizing the violations, analysing and fixing violations in an iterative manner.

1.5 Thesis Organisation

The thesis has been organised as follows: chapter 2 provides highlights the inefficiencies of the current methodologies through the related work while providing the solution to tackle the issues. It is followed by the proposed methodology in chapter 3. The results of

the proposed method and its discussion is provided in chapter 4 followed by conclusion and future work in chapter 5.

Chapter 2

State of the Art

This chapter highlights the work related to this study, in use by the industry and the academia, followed by explaining the details of the issues existing in the current methodology along side mentioning the intent of this work, the need of the hour.

2.1 Related Work

Olivier Coudert, through [9], exposes the problems of timing closures. It includes signal integrity, design variable dependencies, clock and power/ground routing, and design signoff, capacity. It emphasis on the need of tool which can provide a signoff flow solution to reduce the turn around time.

Various robust tools exist like the Integrated Circuit (IC) compiler which is capable of timing driven placement and routing. However, new timing violations arise following the placement and routing stage by adding new logic to the design to meet the functionality requirement, as mentioned in [10], [11]. The latter paper discusses the main reasons for it, one of them being the emergence of the number of scenarios because of which the designers over or under constrain a certain design block, that is they either miss out on some set of constraints or they over constrain with an effort of making it operable at higher frequencies than required in order to reuse in other chips. There exist yet another issue, which is the correlation between various tools. As addressed in the same paper, multiple scripts to deal with timing closure violations for multiple scenarios for ECO guidance exist, however they are not sophisticated enough to take care of the ping-pong effect.
Due to complexity of the design and exponential size of the corner space, the design is usually analysed for a set of corners, as is pointed out in [12], expecting it to cover the design for all worst case scenarios. The design teams find it difficult to scale their scripts to cope with the large designs and dozens of scenarios [11]. [12] paper proposes an automated methodology for finding out the worst case scenarios based on gate and interconnect delays, however such scenario extraction is done only for performing STA and not for ECO stage.

Enough research work has been done to improve hierarchical analysis in the design flow from RTL, placement and routing stages. But post route analysing and fixing for achieving timing accuracy is a lesser explored domain. This causes bottlenecks In the design leading manifold increase in ECO iterations. This has been pointed out in [13]. The purpose of this paper is to present the challenges and the gaps associated with hierarchical timing analysis and crosstalk signoff analysis. It also provides a comprehensive solution to address this issue by introducing the benefits of another tool, Modelscope. However, increasing the number of tools would increase the divergence between the tools at various stages creating bulky reports to analyse. None of the tools have utilised the best capabilities of Primetime alone, neither a complete methodology is proposed to reduce the divergence between the tools. Interest is only shown to compare the divergence in the timing reports between the tools for understanding whether their design is over constrained or under constrained. This aspect of current industry situation is analysed in [14]. In [8], An effective methodology for functional changes and removing timing violations is proposed by utilising the techniques of buffer insertion and gate sizing. Their results show that their approach fixes the timing violation paths with functional changes in lesser runtime. But, since all the violations are targeted at once, the runtime is still huge and may lead to ping-pong effect when applied to all the scenarios. Also, the analysis and fixing stages for functional changes are performed separately which may produce divergent reports when integrated for complete signoff flow. In addition to this, excessive buffer insertion and gate sizing increase the area, and leakage. This requires large number of modifications to be made during placement and routing stage, which poses a threat of introducing other violations.

[15] focuses on reducing runtime of the overall STA runs. Since ECO causes the propagation delays, hence STA is performed after each ECO stage to ensure minimal violation changes. With the increase in the complexity of the design, the STA execution time also increases. It is thus an inefficient approach to perform STA on full chip. This paper proposes incremental STA algorithm in which it utilises the information available from formal verification, which is a stage which identifies the modifications done after ECO stage. On the basis of the information available in the form of net, cell delays, fanin of each pin and parasitics, the circuit is divided in cones, and STA on only those cones are performed which are expected to be affected. This is an efficient method for small designs, and hence it is not recommended. Also, signal integrity issues have not been taken care of, else the complexity will increase.

2.2 Current Methodology and Need of the Hour

In the entire flow of the design stages of an SoC, the most critical stage is to perform the ECO. It is very lengthy process, and if not handles efficiently, can lead to design failure or the adversely affect the timing closure.

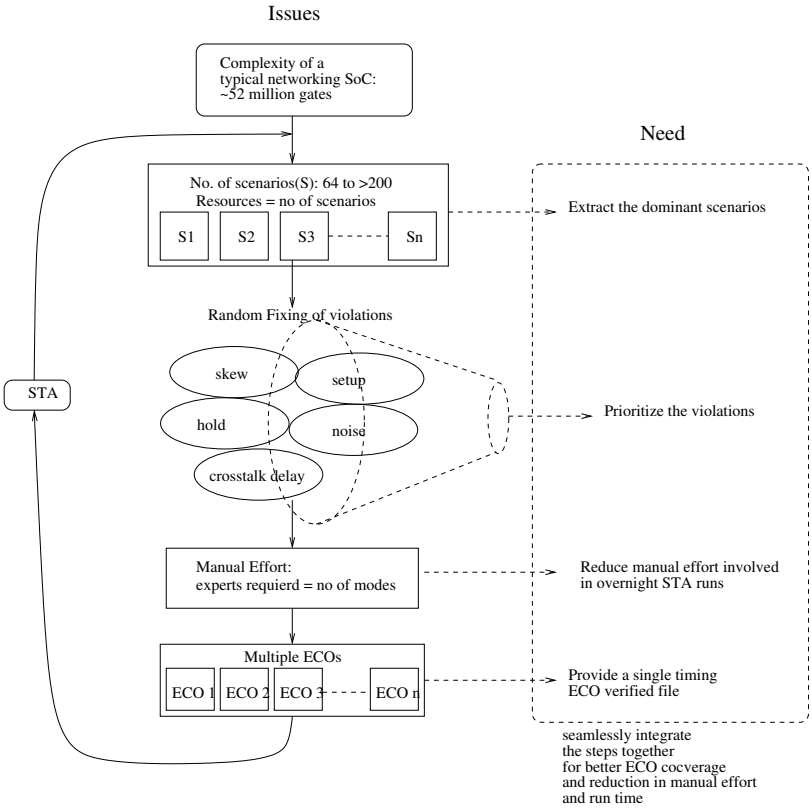


Figure 2.1: Current methodology issues vs required solution

Figure 2.1 shows the inefficiencies of the current methodology along with the respective desired method to counter the existing issues. For a complex networking chip of 52million gates, operating at 28nm technology node, the number of scenarios may lie in the range of 64 to above 200. Performing STA and ECO on such huge number of scenarios leads explosive increase in the total runs requiring access to equivalent number of resources and licenses which are already very limited. On the other hand, the effort involved in manually analysing and selecting the worst case scenarios is unimaginable, which ultimately leads to performing ECO on just extreme scenarios. This work targets to reduce the number of runs by analysing and selecting the number of worst case scenarios, completely eliminating the manual effort involved.

The next in line is fixing violations. The current method involves *random* picking of timing violations for fixing. This random fixing of violations further increases the STA+ECO runs. This is because randomly fixing the violations, without considering the impact of fixing other violations(which may create new type of violations), can lead to viscous ECO loop. This combined with the huge count of scenarios, causes non-convergence of timing closure. The end result is, the design is closed with some

violations still remaining.

To counter this effect, however, the manual effort is divided among the designers, with each designer targeting a particular mode. Although, they generate their own custom scripts, but still the violations and in such a case the man effort involved is enormous. And yet, fixing the violation process is random. This process is, thus, prone to the ping-pong effect, where randomly fixing one type of violation by a designer working on a particular mode, creates other violations in other scenarios. This process is unpredictable.

Also, as illustrated in Figure 2.1, since the process of ECO is performed in discrete steps, this generates multiple redundant ECO files in the end. It is thus clear that the current ECO methodology is non-convergent. This convergence results in a major dependency on manually fixing the violations, which is extremely time consuming. The ECO changes are inefficient, as they are performed by hand (error prone) which ultimately leads to performing a lot of modifications during the post routing stage, which is extremely expensive.

The target of this work is to thus *seamlessly integrate* STA with ECO, contrary to the current discrete process, to reduce the number of STA Runs eliminating the manual effort involved. The theory explained in the previous sections, explains the causes of timing violations. The work utilises those concepts in *identifying the dominant scenarios*, understands the importance of fixing violations by *prioritizing* the type of violations on filtered endpoints, unlike on all as done in the current methodology, to improve the ECO coverage. Each step of the proposed work is explained in detail in the next chapter 3.

Chapter 3

Proposed Methodology for SoC Timing Closure

To overcome the current experimental based method of performing ECO wherein, each expert is assigned a mode to fix violations in all the scenarios of that mode. However, since this is a manual process, the methodology involved in fixing the violations is not systematic. Any particular type of violation maybe targeted without realising that it may worsen the situation too, it may introduce the ping-pong effect, affecting the runtime.

Also, the current focus targets all the violating endpoints. However, performing ECO fix on filtered endpoints only can significantly reduce the number of violations without the need of considering all the violating endpoints.

This chapter introduces a novel methodology, which extracts the dominant scenarios, prioritizes the type of violations, filters the violating endpoints at each stage and thus improves the runtime with minimum involvement of manual effort. It also explains the step by step issue faced with the current methodology along with discussing the significance of proposed step of the methodology.

3.1 Methodology Overview

Referring to Figure 3.3c , the first step of the methodology is to identify and extract the dominant scenarios. The inputs provided to the first step are shown in Figure 3.2 The next step follows launching of Primetime Distributed multi-scenario analysis (DMSA) on the dominant scenarios. The tasks of prioritizing the type of violations and filtering of the violating endpoints follows next. Since the clock tree gets frozen in the later design stage, it is the holy grail of the design. Violations arising out of the network needs to be tackled first. Hence Analysing of clock skew, on the dominant scenarios, is considered to be of the highest priority among other types of violations. This step also

filters out the endpoints for the next step which is crosstalk analysis.

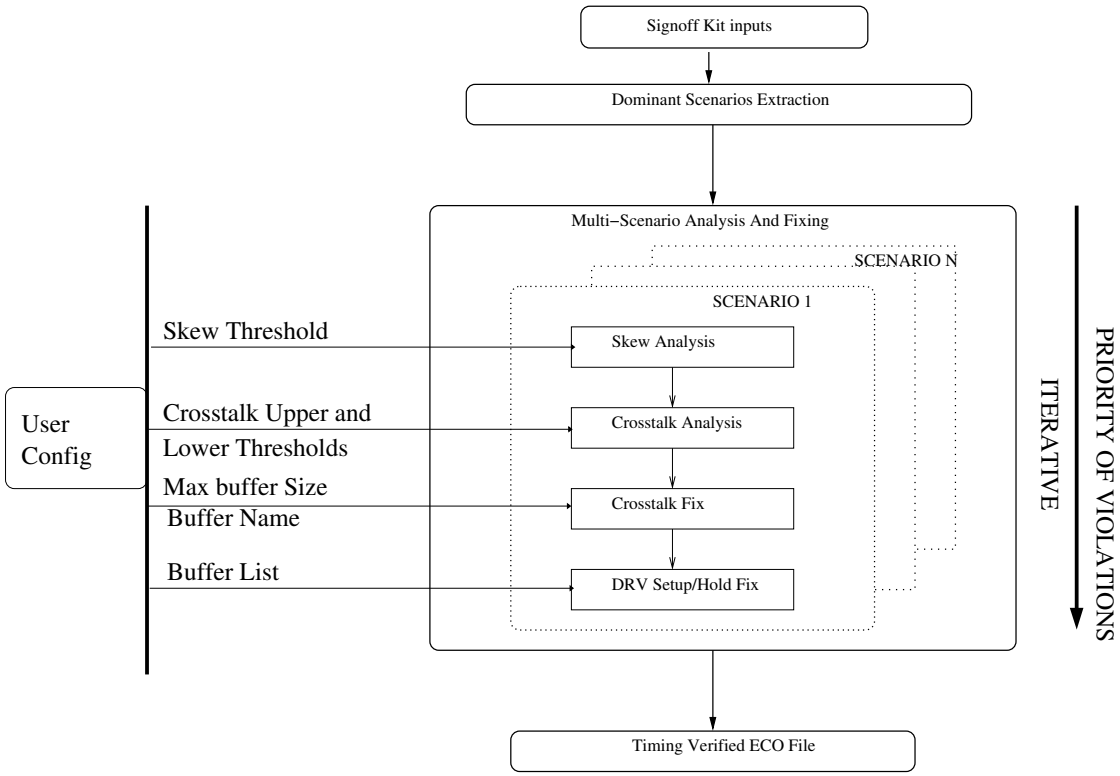


Figure 3.1: Methodology overview

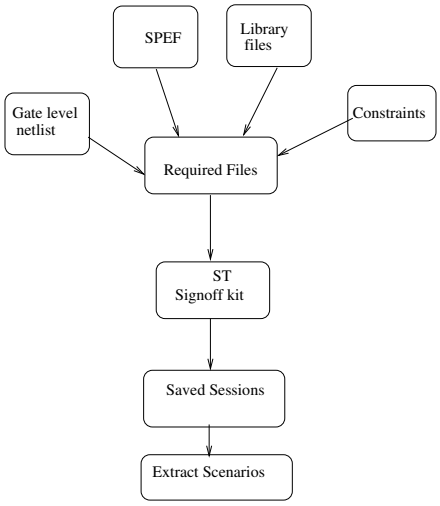


Figure 3.2: Inputs to dominant scenarios extraction step

Primetime does not automatically fix the violations arising due to crosstalk. Hence, the next step of the flow is to fix the crosstalk on the endpoints filtered from the previous step. Till this point, it has been observed that maximum setup and hold violations have been fixed. Hence, the last step of the flow is to fix the setup and hold violations on the remaining filtered endpoints from the crosstalk fix flow. The three fix steps namely, crosstalk fix ,setup and hold fix, forms a part of the ECO. Also various kinds of reports are generated for users better understanding.

It is be noted that, user reconfigurability is provided at each step of the flow which provides the user access to control the ECO. Also various kinds of reports are generated for user's better understanding.

3.2 Integrated Steps of Methodology

3.2.1 Dominant Scenario Extraction

With the existence of enormous number of scenarios, timing analysis and performing ECO fix are one of the critical stages of the digital integrated circuit design. Since it is impossible to consider fixing of all the violations in all the scenarios, the violations are fixed only in the worst-case scenarios, assuming it to cover all violations in the remaining scenarios. However, this method is pessimistic as the occurrence of violations in the scenarios is unpredictable and also figuring a way of finding worst-case scenarios is difficult. At present, picking out the right set of worst-case scenarios from all the scenarios, does not follow a systematic methodology. This task relies on the experience of the experts, design and process engineers, as mentioned in [16].

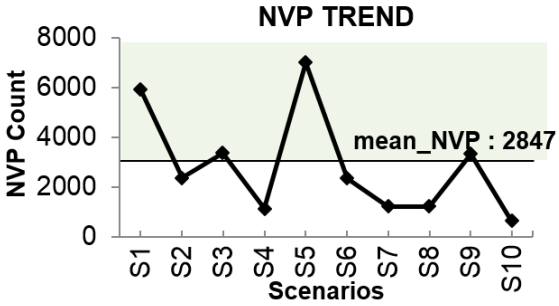
Care also needs to be taken while choosing the worst-case scenarios, as a path may be setup violating, having max delay type, in one corner and hold violating, having min delay type, in the other. This work proposes an efficient way of extracting the dominant scenarios or the worst-case scenarios considering both the delay types, min and max.

Generally dominant scenarios are found on the basis of worst delay types. That is the scenario having maximum number of violating endpoints is considered a worst-case scenario, similarly, other scenarios can be picked in the similar fashion. However, set of scenarios found using this method can often miss other scenarios which may be critical too, to be considered as a worst-case scenarios. For example, from the Table 3.1, on the basis of number of violating endpoints (Number of Violating Points (NVP)), that is the scenario consisting of maximum number of violating endpoints (having worst violating slack), scenarios S1 and S2 can be considered as a set of dominant scenarios. However, this misses the scenario S3 which has the Worst Negative Slack (WNS) value among all the scenarios.

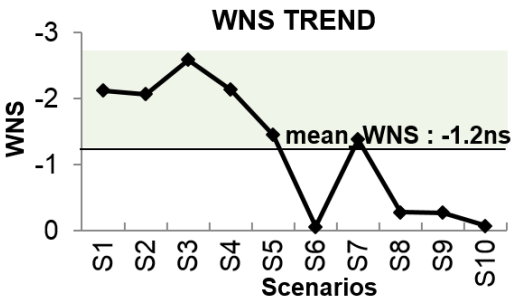
Table 3.1: Worst-case delay scenarios

Scenarios	Number of violating points	Worst Negative Slack(ns)
S1	127	-1.2
S2	100	-0.7
S3	53	-3.5
S4	69	-0.1

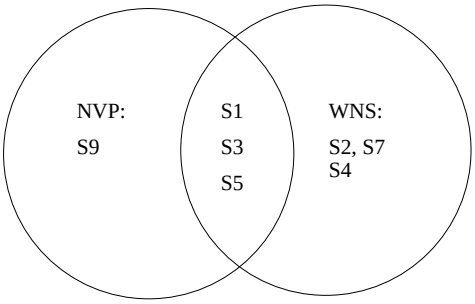
This approach extracts the set of dominant scenarios on the basis of NVP and WNS for each delay type and for each mode, ensuring that fixing the violations in these carefully chosen set, efficiently reduces or eliminates the number of violations in other scenarios too.



(a) Dominant scenarios extraction on the basis of NVP criteria



(b) Dominant scenarios extraction on the basis of WNS criteria



(c) Selected dominant scenarios

Figure 3.3: Dominant scenarios extraction per mode per delay type on the basis of NVP and WNS criteria

NVP count and WNS value is found out for each corner per mode per delay type. NVP count here refers to the number of violating group-endpoint combinations for that particular scenario. WNS value refers to the group-endpoint combination having the worst negative slack value among all the group-endpoint combinations. From all the

Table 3.2: Dominant scenarios extraction

SCENARIOS	NVP COUNT	WNS(ns)
Delay : Min, Mode: FuncC Corner : slow, 40C, cC	5894	-0.95
Delay : Max, Mode: Test Corner : fast, 125C, cC	234	-2.58
Delay : Max, Mode: FuncC Corner : slow, 40C, RC	1130	-0.74
Delay : Min, Mode: FuncC Corner : fast, 40C, RC	2834	-0.97
Delay : Max, Mode: FuncC Corner : slow, 40C, cC	6745	-2.13

NVP counts and WNS values available, mean value is calculated for both of them, respectively. Thus, scenarios with NVP count above the mean NVP, forms one set. Similarly, another set is formed for WNS criteria. The scenarios lying in the intersection of the two, forms the set of dominant scenarios as shown in Figure 3.3. The example shown in this figure is found for each delay type and for each mode. Table 3.2, represents an example of finding a set of dominant scenarios. Therefore,

- for delay type: Min, mode: FuncC, the dominant corner is slow, 40C, cC, chosen on the basis of NVP criteria.
- for delay type: Max, mode: Test, the dominant corner is fast, 125C, cC, chosen on the basis of WNS criteria.
- for delay type: Max, mode: FuncC, the dominant corner is slow, 40C, cC, chosen on the basis of both NVP criteria and WNS criteria.

3.2.2 Skew Analysis

The dominant scenarios extracted in the previous section, are launched for concurrent analysis and fixing of the design. Timing violations in deep sub-micron technologies occurs not only due to process variations but also because of the high switching networks, which is the clock network in this case, operating at high frequencies. Thus, clock nets acts as aggressors causing crosstalk affects on nearby data path and variations in clock skew and thus worsening the slack. Thus, this becomes the 1st process of the methodology.

Although many works have been focused on the clock skew optimization [17] [18] [19] mainly during the physical design stages, placement and routing. However, timing violations still exist, in a multi-scenario environment, even after detailed routing, as focus is on the Clock Tree Synthesis (CTS) and little attention is given to timing information on the data path [20].

There is yet another concern, while analysing clock skew. Firstly, since it is not advisable to touch the clock network in this later design stage (after post CTS), and since the focus is to eliminate timing violations from the data paths only, clock skew arising in the clock network needs to be ignored and reported back to constraint developers,

and and that of only the data paths needs to be considered. Secondly, clock skew can either be negative or positive, however, too negative and too positive can have adverse effects. Clock skew being too positive can lead to a setup time violation, where the data reaches the capture flop too late, relative to the clock pulse. This limits the clock frequency. On the other hand, excessive negative skew may create a race condition causing hold time violation that is the data propagated at the output of one register gets captured by another in the same clock cycle [21]. Thirdly, achieving zero skew for proper functioning of the high speed design would mean that all the flip-flops would toggle at the time exhibiting peak power which is undesirable.

Thus, to mitigate the aforesaid issues, a permissible clock skew threshold is defined. This threshold value is asked by the user for making it more user configurable. Since the calculation of skew is different for both the setup and the hold cases, the value of skew is considered to be violating if it is either negative, as explained earlier in chapter 1, or is less than a particular positive value. Hence, this Value is the skew threshold value. This value ensures that there is no excessive positive skew and takes necessary action of the violating skew, which is described below.

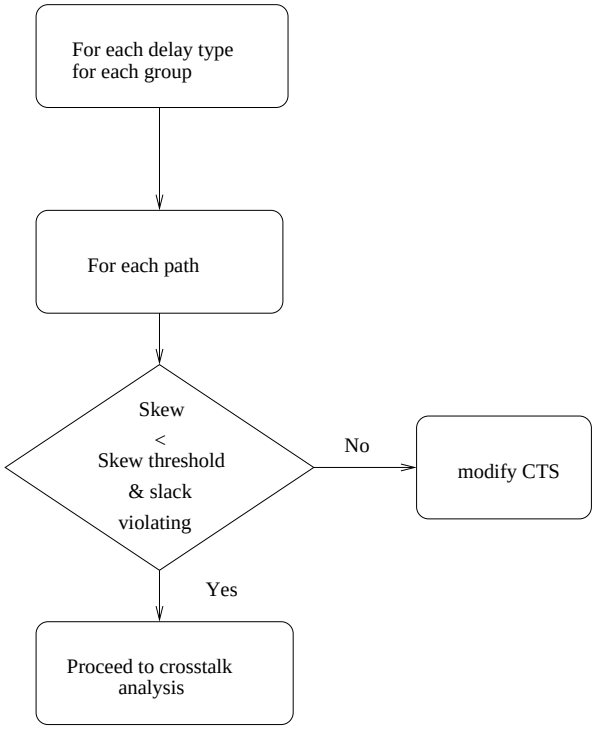


Figure 3.4: Skew analysis algorithm

The algorithm is an iterative process, it moves per delay type then group-wise, that is for each clock group. Now, each clock group has multiple paths. So, for each group,

the algorithm moves one level down the hierarchy, that is, now it works on each path. For each path, it finds the launch and capture clocks and thus their respective latencies, startpoint latency and endpoint latency, for calculating the skew. if the clocks have some common path, it is also taken into account while calculating the skew as shown in the equations below.

$$skew\_hold = endpoint\_latency - startpoint\_latency + crpr \tag{3.1}$$

$$skew\_setup = startpoint\_latency - endpoint\_latency - crpr \tag{3.2}$$

In both the equations 3.1, 3.2, common re-convergence pessimism(crpr)is removed, as it is an inaccuracy occurring between two clock paths sharing a common path segment, arising due to on chip variations, as explained in 1. In 3.2, for setup skew, maximum delay along the launch clock path is considered and minimum delay along the capture clock path. Similarly, for hold skew calculation in 3.1, minimum delay along the launch clock path is considered, and maximum delay along the capture clock path.

After the skews of the paths for a particular group has been calculated, the task of endpoint filtering takes place. The endpoints of the violating paths, having slack lesser than 0 and skew less than the skew threshold, are considered for the next stage of the flow which is the crosstalk analysis. Hence, the two deciding factors, the skew threshold and the slack, defines the range limiting the number of endpoints to a small set. This set is believed to have either excessive negative skew or excessive positive skew. The other endpoints, are segregated path wise and group wise to be given to the constraint developers. The basic algorithm is shown in Figure 3.4 .

3.2.3 Crosstalk Analysis

The cause of violating skew, on the violating paths (having slack lesser than 0), is considered to be crosstalk. This can be attributed to the interaction of signals or interconnects with the clock network causing coupling effects to it, affecting its skew and latencies. Change in clock latencies with respect to the ones predicted earlier, prior to routing, causes timing violations. The desire is to have minimum clock skew due to crosstalk. Prior work have used specialised balanced routers to equalise the length of the wires at any level of the clock tree [22]. However, this technique was ignorant to the effects of coupling, often yielding to undesired clock skew.

For the set of violating paths, filtered from the previous stage, the algorithm, as shown in Figure 3.5 now analyses each of these paths. Each path has multiple nets which may or may not have crosstalk on them. This is decided on the basis of delta delay, that is the delay caused on the victim nets by the aggressor nets. This delta

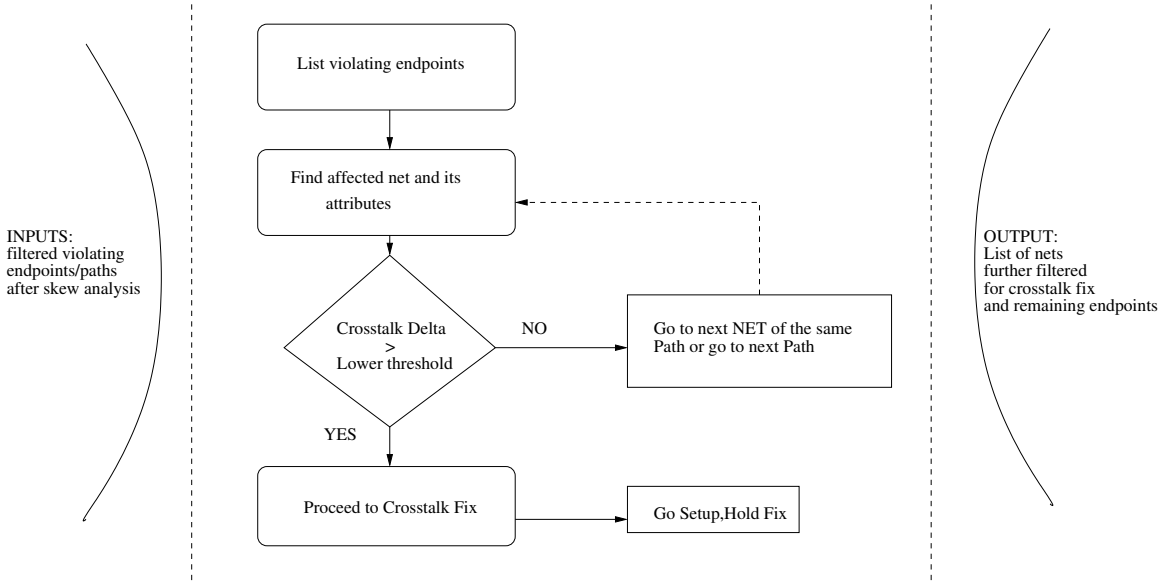


Figure 3.5: Crosstalk analysis algorithm

delay is found for each of the nets depending on the type of transition(rise of fall) and type of delay(min or max). If this delta delay value is greater than the threshold value, which can be set by the user (or default set value), then that net is considered to be a bottleneck for the design causing significant timing violations. A list of such crosstalk affected nets is created, to be considered for the fixing process.

The paths which have violating skew and violating slack but does not have significant crosstalk on them are filtered for setup and hold fixing process stage.

3.3 Engineering Change Order Fix

3.3.1 Crosstalk Fix

The list of crosstalk affected nets extracted after crosstalk analysis needs to undergo crosstalk fixing process to eliminate timing violations arising out of it. An internal loop is initiated on this list. From the list, the worst affected crosstalk net is picked up, based on the crosstalk delta on it. To reduce or eliminate the crosstalk on a net, the drive strength of the victim driver needs to be improved. The drivers of the aggressor nets are not downsized, as it is more prone to inducing timing violations on non critical paths. Hence, working on the victim net for reducing signal integrity issue is a better option. Based on the level of crosstalk effect on this victim segment, out of the three crosstalk fix ECO techniques, one is used. The three delay optimization techniques are:

- **VT Swapping** : It is known that the speed of a circuit is governed by how fast

the capacitance on the output of the cell charges or discharges. The threshold voltage dictates the transistor switching speed, that is it determines what is the minimum voltage required to charge the capacitor. Higher the threshold voltage, higher the oxide thickness, more time the capacitor takes to charge, to reach to that level, decreasing its speed. Thus to reduce the delta delay on the net, the gate delay or the cell delay of the driver cell needs to be reduced, which is done by swapping the cell with its equivalent Low Threshold Voltage (LVT) cell available in the libraries. Switching to LVT cell, having low oxide thickness, ensures that the time taken by the cell to switch its output, when the input arrives, decreases, increasing its speed [23], [24], [25] as given by, making it less prone to effect of transition of the aggressor nets on itself.

VT swapping is the preferred method to be considered for fixing as changes can be made with no modification during placement and routing. However, decreasing the threshold voltage also jeopardises the net for leakage related problems. At low threshold voltages, the power increases with the increase in transient current.

- **Driver Upsizing** : Another technique of reducing the crosstalk delay is upsizing the victim driver cell [26], [27], [28] for improving the output drive of the victim cell. Upsizing the cell means, increasing the width of the transistor which effectively decreases the resistance of the cell resulting in increasing the (dis)charging current. From the Elmore Delay calculation of a cell [25], [26], shows that a cell can be replaced by its equivalent RC network and its delay (τ) is given by eq. 3.3

$$\tau = 0.69 R_{int} * C_{eff} \tag{3.3}$$

where τ is the delay of the cell, R_{int} total internal resistance of the cell, C_{eff} is effective internal and load capacitance and S is the scaling factor.

$$Inmos_{sat} = \mu * C_{ox} * W * (V_{gs} - Vt)^2 * (1 + \lambda * V_{ds})/L \tag{3.4}$$

$R_{int} \propto 1/Width\ of\ the\ transistor$, thus, by scaling the transistor when the output load is fixed, that is increasing its width decreases the resistance and increases the drive current as can be observed from eq. 3.4 which is a saturation current equation for nmos ($Inmos_{sat}$) , W is the width of the transistor, L the channel length, C_{ox} is the oxide capacitance. So, Upsizing the cell improves the output transition, reduces the propagation delay time of the cell improving the drive strength. However, from eq. 3.5 [25], increasing the width of the cell also increases the gate and parasitic capacitances, which provides larger load in the fanin of the cell. Also, Upszing the cell requires re-routing, and so is preferred second to VT Swapping.

$$C_{gate} = C_{ox} W L \tag{3.5}$$

where C_{gate} is the gate capacitance.

- **Buffer Insertion** : This is an effective technique for interconnect driven timing optimization. The interconnect delay is directly proportional to the square of the interconnect length. Inserting buffers in smaller segments, makes interconnect delay linearly proportional to the wire length [29]. It reduces the elmore delay of the nets by driving net capacitances through smaller resistances, thereby improving the slack of the path. However, it increases the area drastically along with requiring modifications in the re-routing stage.

  Table 3.3, showcases the effect of inserting buffer in min data paths of an SoC, consisting of 9 million gates, in order to increase the delay of the path for reducing the effect of crosstalk from adjacent effective aggressors and hence removing hold type violation from this path by improving its slack. In both the cases, Path 1, Path 2, fixing the worst affected net by inserting a single buffer improves the slack of the path.

Table 3.3: Pathwise crosstalk fixing by inserting buffer

Paths	Slack	Improved Slack	Delta	Improved Delta
Path1	-.0032ns	0.0055ns	-0.0630ns	0
Path2	-0.0152ns	0.0176ns	-0.0618ns	0

To consider the effect of negative crosstalk delay causing a hold violation, as explained in chapter 1, if the crosstalk delta on the worst picked net is greater than the upper bound threshold, which is user configurable, the third technique, buffer insertion, is used. The buffers are inserted at equidistant points on this segment. The buffer to be inserted is taken as an input from the user. If the crosstalk delta of the worst affected net is in the range of upper and lower crosstalk thresholds, its details and its driver cells details are extracted for choosing from the two ECO techniques for the fix, VT swapping and upsizing. The details of the net include the driver cell, transition type, delay type, slack of the path. And that of the driver cell includes its unateness, input slew, output load. These details of the driver cell helps in choosing the collection of alternative cells, cells having the same functionality, available for sizing the driver cell. The behaviour of each cell from this collection is noted when put under same conditions as that of the driver. Based on the drive strengths either VT swapping is chosen or upsizing(upsize to the maximum size available except for the don't touch and don't use cell list).

The alternative cell with the best input slew is chosen for replacement. Its slew value is compared against the path slack value. If its lesser than that, the driver cell

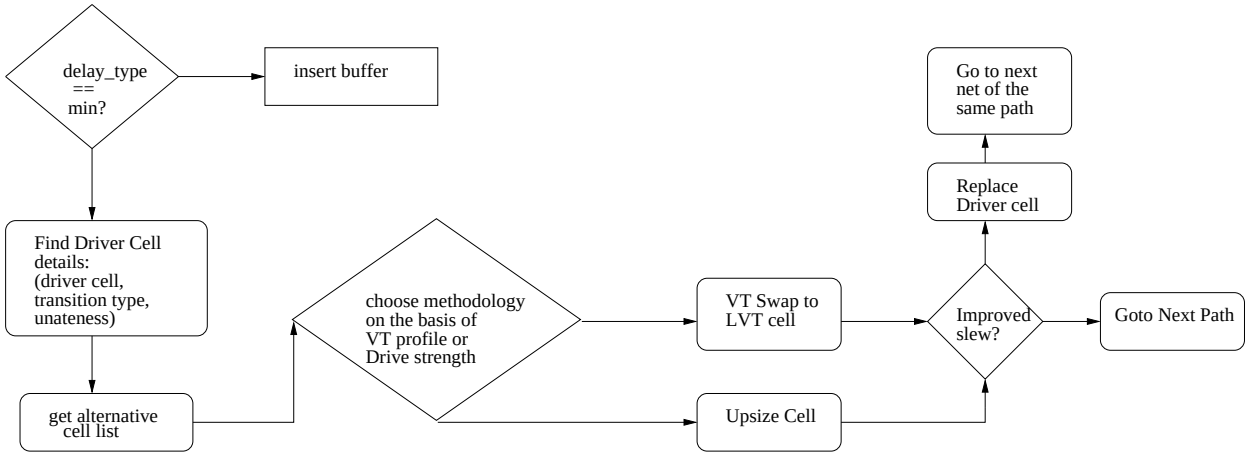


Figure 3.6: ECO: crosstalk fix algorithm

is successfully replaced with this alternative cell and the value equivalent to path slack value is updated for the next inner iteration. The slew value is compared with slack value as the ultimate goal is to eliminate the timing violation induced due to crosstalk. After successful replacement, the next worst net from the list is considered. This inner iteration, of finding the worst affected net, its details and its driver cell details followed by replacement, over the list of nets of a particular path continues till either the slack of the path significantly improves or the list gets exhausted. In such a case, the next path of a group is taken into consideration. And after all the paths of a group have been analysed, the next group is considered. The flow of ECO crosstalk is shown in Figure 3.6.

3.3.2 Setup Fix and Hold Fix

At this stage, maximum violations arising due to crosstalk have been eliminated. There can be setup and hold violations arising solely out of bad skew, positive and negative clock skew respectively, as explained in chapter 1. Those timing violations are focused upon, in this step. It is seen that setup is a harder problem to fix, fixing setup can introduce hold violations. On the other hand, chances of fixing hold violations creating new setup violations are bleak. In addition care for such a case is done by carefully setting hold margin while fixing setup violations and setting setup margin while fixing hold violations. For fixing setup violations, only size cell or VT swapping is considered, based on the methodology described in the previous section. The purpose is to reduce the delay of the data path as compared to the delay of the clock path. Inserting buffers in the clock path, is not considered wise option, as although this may improve the slack of a particular path but, may disturb the operation of the design. Multiple iterations for fixing are performed until no more violations can be fixed at this point.

As for fixing hold, iterative process of inserting buffers is done in the data path, with the methodology described in the previous section.

3.4 3 Stage Filtering Process

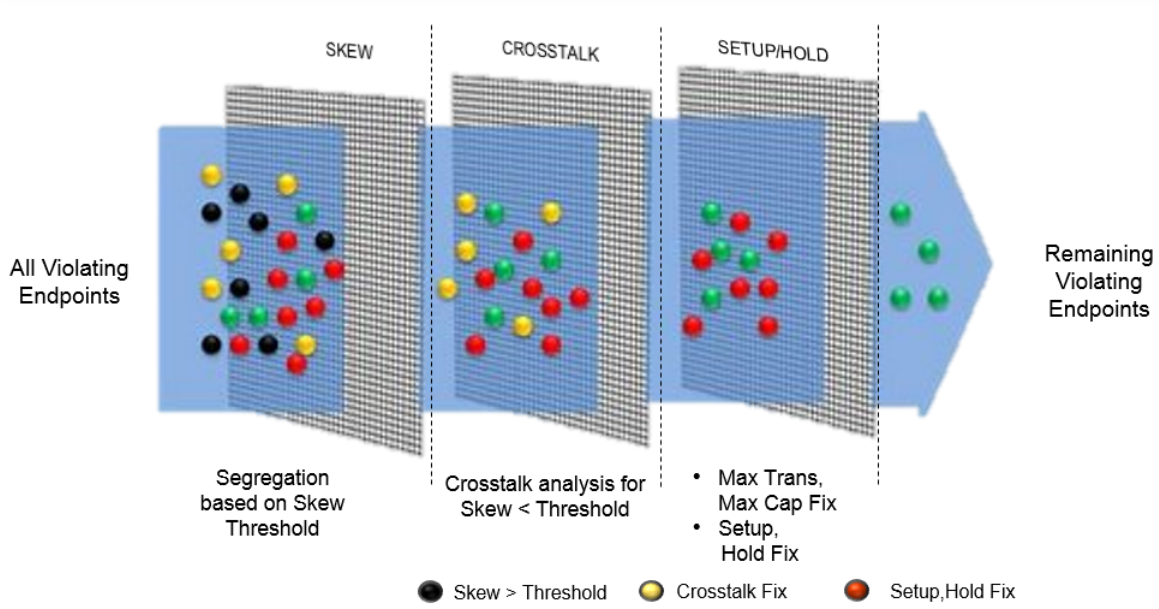


Figure 3.7: 3 Stage filtering process [1]

In this section, an overview of how and when the endpoints have been filtered is shown in Figure 3.7. The intent of endpoint filtering process is to reduce the number of iterations. As explained in the previous sections, the worst case endpoints have been picked from all the scenarios, and then filtering takes place at each stage of the flow. The worst case endpoints have filtered after process in order to ensure that performing fix on these worst case endpoints will subsequently fix violations on low priority ones. The endpoints are filtered in a prioritized manner to eliminate the effect of the *ping-pong effect*. Thus, it improves the overall runtime and also ECO coverage.

Chapter 4

Results And Discussions

In this chapter the results obtained have been presented and discussed. It firstly provides the experimental setup on which the trials have been performed. This is followed by results obtained at each step of the methodology starting with the extraction of dominant scenarios, followed by crosstalk ,setup, hold fixing and finally concludes with the inferences drawn from the results.

4.1 Experimental Setup

Table 4.1 provides the details of the simulation setup on which trials were performed.The tool used for trials is Synopsys Primetime.

Table 4.1: Experimental Setup

Operating System	Red Hat Enterprise Linux v5.9
# of processing cores	10
Vendor ID	Intel(R) Xeon(R) CPU E5-2680
Processor Speed(GHz)	2.80
Tool	Synopsys PrimeTime vJ-2014.06-sp2
Scripting Languages	Perl v5.8.8, TCL v8.5

The flow of the trials and that of the tool is shown in Figure 4.1. All the inputs to the tool is provided in the beginning, however, different inputs are effectively utilised at different stages of the flow.

As explained in chapter 3, the first task is to identify the dominant scenarios. For this, the results of the signoff kit is provided as one of the inputs to the tool. The methodology for this phase has been implemented using Perl scripting language. After the successful extraction of the dominant scenarios, the next step is to filter out the

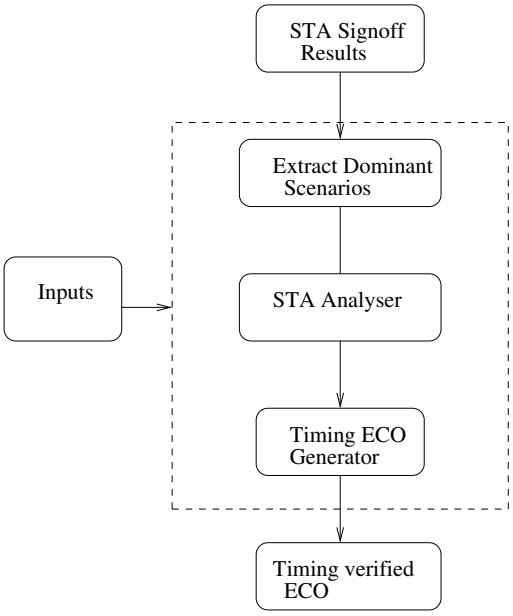


Figure 4.1: Tool Flow

violating endpoints. Fixing these endpoints, it is observed that many other violating endpoints are also fixed to a great degree. The violating endpoints are majorly filtered at two stages: *Skew analysis*, *Crosstalk analysis*. This step of the flow is the *STA Analyser*. The inputs to the flow are the Saved sessions, user set skew threshold, crosstalk threshold. This step is performed on Synopsys Primetime DMSA using TCL language, which is used for launching dominant multi-scenarios simultaneously on different cores called the *slaves* using Primetime licenses equal to the number of scenarios. The last stage is performing ECOs on the filtered endpoints. The filtered endpoints have two categories, one for crosstalk fixing and other for setup and hold fixing as explained in chapter 3. This step of the flow is called the *ECO Generator*. It is again implemented using TCL language on Synopsys Primetime. The inputs to this step of the flow includes the list of buffers to be inserted for hold fixing and crosstalk fixing.

The above mentioned trials were performed on many SoCs, containing numerous blocks having multipleIPs, out of which the results of two designs(SoCs) been captured for this work. The specifications of the two for *28nm Technology node* have been shown in Table 4.2.

4.2 Dominant Scenario Extraction

The results obtained on the basis of algorithm developed in section 3.2.1, is shown in Figure 4.2. the trials were performed on *Design A*, having 64 scenarios after signoff. After running the first step of the flow , refer to Figure 4.1, A small count of scenarios

Table 4.2: Specifications of designs under study for 28nm Technology node

Design A (SoC1)	Application	Set-top-box chip	
	Specifications	Total cell count	9 million
		Memories count	707
		Total chip area	70mm2
	Blocks (partitions)	4	
Design B (SoC2)	Application	Networking chip	
	Specifications	Total gate count	52million
		Total register count	20million
		Total chip area	500mm2
	Blocks(partitions)	6	

equal to 10 out of 64 is taken for the purpose of showing the effectiveness of this step, and on the basis of real data for each scenario in terms of NVP and WNS count, a list of dominant scenario is extracted (16/64 were obtained for *Design A*). The mean NVP value was found as: 1900 and mean WNS value was found as:-1.67ns From the available 10 scenarios (S1-S10), the chosen dominant scenarios and their respective reasons for choosing are as follows:

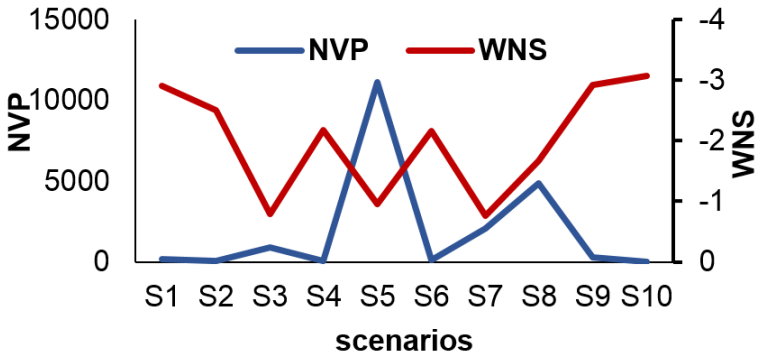


Figure 4.2: Dominant scenarios extraction for *Design A*

i. S1, S10: These two scenarios although have lowest NVP count, but highest WNS count. This ensures that the devised algorithm is not pessimistic, and ensures that scenarios extracted are not just the extreme cases in terms of choosing the slowest ad the fastest corners for analysis.

ii. S5: Although this scenario has high low WNS value but highest NVP count ensuring the robustness of the algorithm.

iii. Similarly other scenarios are S4, S6, S8.

4.3 Design Summaries

4.3.1 Design 'A'

Here, Table 4.3 presents a full step by step filtering of violations followed by fixing. This provides an insight into the intermediary results, for better understanding and analysis of the flow. For an SoC, Design A, refer to Table 4.2 , consists of 8 clock groups, the number of dominant scenarios extracted were 16 out of 64. To achieve 75% fix rate, it shows number the number of iterations performed with the count of violating paths left after each stage along with the number of ECOs performed for crosstalk fixing and setup and hold. Run0 corresponds to iteration 1 and so on. The skew threshold is kept as 100ps, crosstalk lower threshold is kept as 400ps.

Table 4.3: Detailed ECO flow summary of *Design A*

Count	**Run0**	**Run1**	**Run2**	**Run3**
Total violating paths	30912	9923	8624	7635
Total non violating paths	1491401	1512392	1513690	-
Violating paths filtered after skew analysis	25722	8764	7563	-
Violating paths filtered after crosstalk analysis	19925	4520	5663	-
Violating paths left after crosstalk fix	2797	3643	900	-
Buffers inserted for fixing crosstalk	9800	613	111	-
Buffers inserted for fixing hold	1179	200	69	-
Cells resized for fixing crosstalk	10125	552	42	-
Cells resized for fixing setup and hold	7314	214	143	-

4.3.2 Design 'B'

A. Same analysis was performed on another SoC, *Design B*, that is to achieve 75% fix rate, on 18 out of 64 scenarios, Table 4.4, provides a short summary of the results achieved. It consists of 12 clock groups. The skew threshold is kept as 100ps and crosstalk threshold is kept as 400ps.

B. On the basis of the results obtained from *Design B*, a trend of violations fix rate has been shown in Figure 4.3. This figure provides a comparative analysis of the distribution of type of violation fix rate per iteration. It is evident from the figure that conventional method is a random process, wherein the process of

Table 4.4: Brief ECO flow summary of *Design B*

ECOs	**Run0**	**Run1**	**Run2**	**Run3**	**Run4**	**Run5**
No. of violating paths	146625	45601	38855	37815	37096	35776
No. of non violating paths	6712785	6667184	6673932	6674971	6675689	6677012
Buffers inserted	24263	1326	434	221	78	-
Cells resized	28614	5795	2435	1687	823	-
Violations Fixed(%)	-	68.90	73.50	74.21	74.7	75.3

fixing the violations is a random process. Depending on the area of expertise of the designer, a particular violation is targeted first, rather than fixing a higher priority violation type first, creating a danger for ping-pong effect. In comparison, the automated ow focuses more on prioritizing the type of violations and filtering the the endpoints in every stage thus preventing the danger of ping-pong effect. By observing the trend, it is clear the violations fix rate is significantly better as compared to the traditional flow as the fix rate (shown by black line in the figure) is certain to improve at each stage unlike in the conventional flow where the fix rate cannot be predictive.

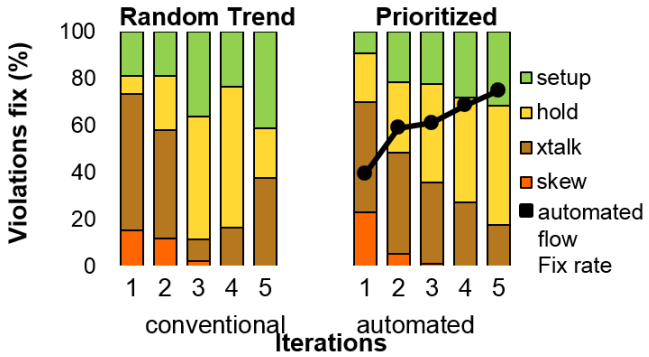


Figure 4.3: For*Design B*, comparative analysis of Trend of fixing violations between conventional and automated flow

C. A slack distribution graph ,shown in Figure 4.4, represents the percentage of violations ,to achieve 75% fix rate, fixed in 5 iterations saving approx 93% of manual effort as compared to the conventional flow, along side the slack distribution of all the violations before fixing. This shows the effectiveness in reduction of violations and significant improvement in the path slacks.

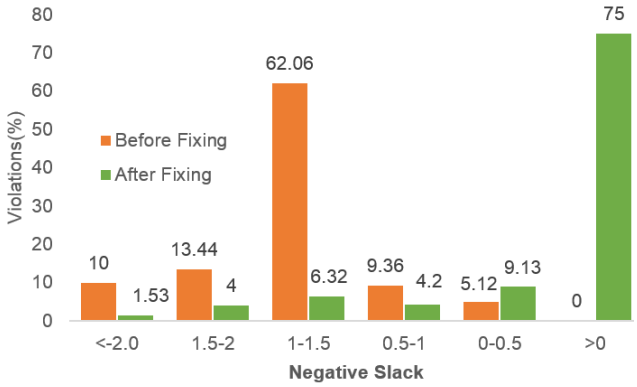


Figure 4.4: For *Design B*, Slack distribution of violating paths(%) before and after fixing

4.4 Inferences

All in all, Table 4.5, provides the summary of the benchmarks achieved, after performing trials on *Design A*, *Design B*.

Table 4.5: Performance metric

	Criteria	**Traditional method**	**Proposed method**	**% Reduction**
Design A	Effort	3.5 man months	4 man days	96.20
	Iterations	8	2	75
	Count of ECOs	25	2	92
Design B	Effort	5 man months	5 man days	93.33
	Iterations	12	5	58.33
	Count of ECOs	60	5	91.66

Chapter 5

Conclusion And Future Work

5.1 Conclusion

The results achieved from the proposed methodology proves the efficacy over other existing methodologies. This approach accelerates the timing closure by eliminating the costly ECO iterations involved. It is able to successfully address the process challenges by extracting the worst case scenarios for the multi scenario analysis and ECO guidance flow followed by accurate filtering of violating endpoints at each prioritised stage for fixing the timing violations without creating new ones, thus, preventing the ping-pong effect. It allows the system on chip (SoC) design engineer to accurately correct the timing violations without intervening manually and letting the tool to perform comprehensive analysis of timing violations followed by fixing of the violations in a hierarchical manner. The results of this methodology show the achievement of 90% to 95% cycletime goals, gain in capacity and memory savings. It thereby reduces the verification time and final time to market.

Hence, this methodology seamlessly integrates the various steps of the signoff flow, significantly reduces the manual effort involved and thereby contributes in improving the runtime and performance.

5.2 Future Work

Currently, the focus has been timing analysis, Integrating analysis and fixing of crosstalk noise would make this tool even more robust. Along with this, integration of analysing the constraints of the design for unclocked flops and endpoints because of which the violating paths ,even though existing, maybe missed, can be a part of it.

References

[1] **Filtering process**. http://www.presentation-process.com/filter-diagram.html.

[2] R. Goering. **Whats Needed to Fix Timing Signoff?** http://community.cadence.com/cadence_blogs_8/b/ii/archive/2013/06/05/dac-2013-panel-what-s-needed-to-fix-timing-signoff, 2013. [DAC Panel].

[3] Myeoungwoo Jin Deokkeun Oh, Eunsuk Park and Juho Kim. **K-Critical Path Search Based Multi corner Multi mode Static Timing Analysis**. *ISOCC*, pages 212–213, 2014.

[4] S. Narendra J. Tschanz A. Keshavarzi S. Borkar, T. Karnik and V. De. **Parameter variations and impact on circuits and microarchitecture**. *Design Automation Conference*, pages 338–342, 2003.

[5] Sudhakar Jilla. **Using multicorner multi mode techniques to meet the P&R challenges at 65 nm and below**. http://www.techdesignforums.com/practice/technique/using-multi-corner-multi-mode-techniques-to-meet-the-p-r-challenges-at-65-nm-and-below/, 2007.

[6] J.BHasker and Rakesh Chadha. *Static Timing Analysis For Nanometer Designs: A Practical Approach*. Springer, 2009.

[7] *Primetime and Primetime SI User Guide*.

[8] Yao-Kai Yeh Jui-Hung Hung, Yung-Sheng Tseng, and Tsai-Ming Hsieh. **A New ECO Technology For Functional Changes And Removing Timing Violations**. *International Symposium on Quality Electronic Design*, pages 1–5, 2011.

[9] Olivier Coudert. **Timing and Design Closure in Physical Design Flows**. *International Symposium on Quality Electronic Design*, pages 1–6, 2002.

[10] Kan Mei War. **CAD Automation Module Based on Cell Moving Algorithm for ECO Timing Optimization**. *IEEE Symposium on Industrial Electronics and Applications*, pages 284–288, September, 2011.

[11] Bernie Mortell. **Signoff-Driven ECO Guidance for Faster Timing Closure**. https://www.synopsys.com/Company/Publications/SynopsysInsight/Pages/Art2-signoffeco-IssQ3-11.aspx?cmp=Insight-I3-2011-Art2, 2011. [Synopsys].

[12] Joel R. Phillips Luis Guerra e Silva, L. Miguel Silveira. **Efficient Computation of the Worst-Delay Corner**. *DATE*, pages 1–6, 2007.

[13] Arvind NV Sreeram C Vish Visvanathan-Shailendra Dhuri Roopesh Chander Patrick Fortner Subra Sripada Qiuyang Wu Rajagopal KA, Sivakumar R. **A comprehensive solution for true hierarchical timing and crosstalk delay signoff**. *International Conference on VLSI Design*, pages 1–6, 2006.

[14] Siddhartha Nath Seung-Soo Han, Andrew B. Kahng and Ashok S. Vydyanathan. **A Deep Learning Methodology to Proliferate Golden Signoff Timing**. *DATE*, pages 1–6, 2014.

[15] Bakhtiar Affendi Rosdi Tee Kok Tiong Ang, Lay Sean. **Cone Extraction Technique for Incremental Static Timing Analysis**. *IEEE Symposium on Industrial Electronics and Applications*, pages 1–6, September, 2011.

[16] Joel R. Phillips Luis Guerra e Silva, L. Miguel Silveira. **Efficient Computation of the Worst-Delay Corner**. *Design, Automation & Test in Europe Conference & Exhibition*, pages 1–6, April 2007.

[17] Yegna Parasuram Tung Cao Amit Chowdhary-Bill Halpin Karthik Rajagopal, Tal Shaked. **Timing driven force directed placement with physical net constraints**. *ISPD*, pages 60–66, 2003.

[18] Hai Zhou Rong Ye, Feng Yuan and Qiang Xu. **Clock Skew Scheduling for Timing Speculation**. *Design, Automation & Test in Europe Conference & Exhibition*, pages 929–934, March 2012.

[19] Min Zhao R. Mahapatra Di Wu, Jiang Hu. **Timing driven track routing considering coupling capacitance**. *ASP-DAC*, pages 1156–1159, 2005.

[20] Wei Chen Yongqiang Lu Qiang Zhou Weixiang Shena, Yici Cai and Jiang Hue. **Useful clock skew optimization under a multi-corner multi-mode design framework**. *Quality Electronic Design (ISQED)*, pages 62–68, March 2010.

[21] Shih-Hsu Huang Yu-Hui Lin Shih-Hsu Huang, Yu-Hui Lin. **Utilizing clock skew for timing reliability improvement**. *TENCON 2007 - 2007 IEEE Region 10 Conference*, pages 1–4, November 2007.

[22] Denise Powell Jesse Craig. **Effects of Specialized Clock Routing on Clock Tree Timing, Signal Integrity, and Routing Congestion**. *SNUG*, 2005.

[23] EDWARD L. TITLEBAUM BORIS ANDREEV and EBY G. FRIEDMAN. **SIZING CMOS INVERTERS WITH MILLER EFFECT AND THRESHOLD VOLTAGE VARIATIONS**. *Journal of Circuits, Systems, and Computers*, pages 437–454, March 2006.

[24] Kiat-Seng Yeo Wang-Ling Goh, Samir S. Rofail. *Low-Power Design: An Overview*. Pearson Education, Inc, June 2002.

[25] David Money Harris Neil Weste. *CMOS VLSI Design: A Circuits and Systems Perspective*. Pearson Education, Inc, 4th edition, March 2010.

[26] Vishwani D. Agrawal Tezaswi Raja and Michael L. Bushnell. **Transistor Sizing of Logic Gates to Maximize Input Delay Variability**.

[27] M. Jasmin. **Optimization Techniques for Low Power VLSI Circuits**. *Middle-East Journal of Scientific Research*, pages 1082–1087, 2014.

[28] Marek-Sadowska M. Xiao, T. **Crosstalk reduction by transistor sizing**. *Design Automation Conference, 1999. Proceedings of the ASP-DAC '99. Asia and South Pacific*, **vol. 1**:137–140, Jan, 1999.

[29] Stephen T. Quay Charles J. Alpert, Anirudh Devgan. **Buffer Insertion for Noise and Delay Optimization**. *Design Automation Conference*, pages 1–6, 1998.