Timing Closure of Partitions for Lower Technology Nodes

Submitted By

MEENAKSHI RAO

16MECV13

Electronics & Communication Engineering Department
Institute of Technology
Nirma University
Ahmedabad - 382 481

May , 2019

Timing Closure of Partitions for Lower Technology Nodes

Major Project Report

Submitted in partial fulfillment of the requirements
for the degree of

Master of Technology
In
Electronics & Communication Engineering
(VLSI Design)

By
Meenakshi Rao
(16MECV13)

Under the Guidance of

Internal Guide
Dr Piyush Bhatasana
Professor (VLSI Design)
Nirma University

External Guide
Sachin Gupta
SoC Design Engineer
Intel Technology India Pvt Ltd.

Electronics & Communication Engineering Department
Institute of Technology
Nirma University
Ahmedabad - 382 481
May , 2019

Declaration

This is to certify that

1. The thesis comprises my original work towards the degree of Master of Technology in VLSI Design at Nirma University and has not been submitted elsewhere for a degree.

2. Due acknowledgment has been made in the text to all other material used.

MEENAKSHI RAO
(16MECV13)

Certificate

This is to certify that the Major Project entitled **"Timing Closure of Partitions for Lower Technology Nodes"** submitted by **MEENAKSHI RAO (16MECV13)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in VLSI Design, Nirma University, Ahmedabad is the record of work carried out by her under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.The results embodied in this minor project, to the best of our knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Dr Piyush Bhatasana
Internal Guide

Dr N M Devashree
Professor (VLSI Design)

Dr D. K. Kothari
Head, EC Dept.

Dr Alka Mahajan
Director, IT - NU

Date :

Place : Ahmedabad

CERTIFICATE

This is to certify that the Project entitled **Timing Closure of Partitions for Lower Technology Nodes** submitted by **Ms. Meenakshi Rao (16MECV13)**, towards the submission of the Project for requirements for the degree of Master of Technology in VLSI Design, Nirma University,Ahmedabad is the record of work carried out by her under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.
Place: Banglore
Date :

Mr. Sachin Gupta
External Guide,
Soc Design Engineer ,
Intel Techology India Pvt. Ltd.,
Banagalore

Acknowledgment

I would like to extend my thanks to my guide Prof. Piyush Bhatasana, Department of Electronics and Communication, Institute of Technology, NIRMA UNIVERSITY, Ahmedabad for his constant motivation and guiding me through this project. He has been extremely helpful with his knowledge and enthusiasm. This project would not have been possible without the kind support he offered. Also I would like to thank Mr. Sachin Gupta, SOC Design Lead, HPG Group, at Intel Corporation for his guidance and help. At Intel Corporation for his constant help and support in executing this project. I am grateful to all my team members for giving me an opportunity to be a part of Intel as an intern and helping me in completing this project. Last but not the least I would also like to thank my parents for their kind cooperation and encouragement which helped me in completion of this project.

Besides my guide, I would like to thank PG Cordinator Dr N M Devashree ,Institute of Technology, NIRMA UNIVERSITY, Ahmedabad , for the support and his excellent guidance directly and indirectly.

- Meenakshi Rao
(16MECV13)

Abstract

Metal interconnects basically used for make the interconnections between different part of the circuitry in order to realize any System on Chip (SoC) design. These metal interconnects affect the performance of the design as the process technology drastically shrinks. For nanometer process technologies, the coupling effect in the interconnect causes noise and crosstalk. These noise and crosstalk can affect the operating speed of the design, which is responsible for the timing aspect of the design. These problems are negligible in the older technologies. Thus, the physical design and verification of latest process technologies should include the effects of crosstalk and noise. If the timing of a design is not verified, then the design may not perform at the desired operating speed it was designed for. Apart from timing there are two other factors that needs to be considered while designing. Those are Power and Area. There will always be a trade-off between these three factors. Static Timing Analysis (STA) is one of the many techniques used by the designers to verify the timing of the design and also for closing the design with respect to timing, which is called as timing closure. There is one more method called Dynamic Timing Analysis (DTA) or Timing Simulation, which was used in older days for timing verification. STA is static because the analysis is done when the design is stable and it does not depend upon the input vectors being given to the design. STA uses setup check and hold check for verifying the timing of the design. Synopsys PrimeTime is the tool used for STA. During the physical implementation of the design (Pre-Signoff/ICC stage) also the timing can be optimized by using efficient floorplanning, placement and routing techniques. Most of the timing optimization are done in physical implementation stage only. Synopsys IC Compiler II is the tool used for the physical implementation. The main aim of this master thesis is to investigate various timing optimization techniques and methods to fix the timing violations in both pre-signoff and signoff stage. In signoff stage, the design will be in Engineering Change Order (ECO) mode and there will be no further design optimization performed in this stage. The timing fixes are given in the form of ECOs and those are sourced and routed. The techniques that are used in this stage are cell sizing and buffer insertion. Only in the pre-signoff stage, optimization of the design is done by changing the floorplan, placement and route of the design. In both the stages of timing optimization, power and area of the design will also change and hence they should also be checked and maintained within limits. These techniques and methods are analysed and implemented to achieve better timing results and ultimately to validate the operating speed of the design.

Keywords: STA, Crosstalk and noise, Setup and Hold check, Signoff, ECO, DTA

Table of Contents

List of Figures

Abbreviations

SDC	Synopsys Design Constraints
SDF	Standard Delay Format
DRC	Design Rule Check
ECO	Engineering Change Order
CTS	Clock Tree Synthesis
HDL	Hardware Description Language
SOC	System on Chip
VLSI	Very Large Scale Integration
GMD	Graphics Media Display
CPU	Central Processing Unit
MPU	Micro Processor Unit
IP	Intellectual Property
IOT	Internet of Things
DCN	Distributed Control Node
PLC	Programmable Logic Controller
UPF	Unified Power Format
CTS	Clock Tree Synthesis
APR	Automatic Place and Route
DTA	Dynamic Timing Analysis
OCV	On Chip Variation
ECO	Engineering Change Order
LVS	Layout vs Schematic
GDSII	Graphic Data System for Information Interchange
WNS	Worst Negative Slack
FEP	Failing End Points
SPEF	Standard Parasitic Extract Format
QOR	Quality of Results

Chapter 1

INTRODUCTION

1.1 Core Technical Area

1.1.1 Overview

Very-large-scale integration (VLSI) is the process in which the creating an integrated circuit (IC) by combining hundreds of thousands of transistors or devices into a single chip. With the advancement in VLSI technology, there is a constant reduction in the feature size of VLSI devices (i.e. the minimum transistor size). The feature size decreased from about 0.25 m in 1997 to about 10 nm today. Such a continual miniaturization of devices has had a strong impact on VLSI technology in several ways. One of those impacts is the timing analysis. As increase in the number of transistors per chip increases the rated speed or frequency at which a design is to be operated has to be met and the number of timing paths rises exponentially high due to multiple connections in the design. The continuous decrease in feature size and corresponding increase in chip density and operating frequency have made exhaustive timing analysis a major concern in VLSI design. Hence, Static Timing Analysis has branched out as an entirely separate domain of expertise in itself in modern day System on Chip Design.— This System on Chip is positioned to be a successor to the previous one and is an improvisation to the already existing one. The platform vision is to take forward the success of the existing SoC by providing new value added features and technologies and also enable significant improvements with the latest CPU, GMD (Graphics Media Display) and other technologies. This System on Chip is designed to cater to various segments within the Internet of Things. It is begin targeted for entry level Micro Processor Unit (MPU) segments in various Internet of Things (IoT) markets with a Thermal Design Power primarily targeted for 6-10W SKU. All these markets have the same principal requirements with regards to power, performance and price point but differ in terms of IP (Intellectual Property) block/feature requirements. Hence the SoC architecture needs to take care of the various IP integration challenges for all these target segments yet meet the common goals for power/performance and price. However the current SoC is primarily driven by industrial, office automation and retail segments. SoC is primarily targeted at industrial automation and Distributed Control Node (DCN) applications. With the birth of Industry 4.0, there is a fundamental change happening with regard to systems architecture across all the domains in the industrial automation segment. This change is primarily driven by the need to connect every

machine to a common cloud infrastructure, gather data from these machines constantly to understand and improve their efficiency, control them in a time sensitive manner to avoid unnecessary cycles and localize the intelligence within the machines to drive better resource allocation. Customers are looking at enabling these systems like PLC/PAC (programmable controllers), motor/motion controllers etc. with sufficient compute horse-power and IO integration to meet their goals of efficiency. Industrial applications are among the major volume drivers for this SoC. Primary segments within the industrial market are broadly classified into Industrial PC (IPC), Programmable Logic Controller (PLC) / Programmable Automation Controller (PAC), Motion Controller (MC), Human Machine Interface (HMI), Industrial Gateway, Test and Measurement. Also Motion Controller applications include Machine Tools, Production Machines, Cranes, General motion control machines etc.

1.2 MOTIVATION

Timing Analysis is carried out using multiple proven techniques which have been in effect for the past few decades. Some of the techniques which are commonly used are Static Timing Analysis, Timing Simulation etc. The method of timing simulation is also referred to as Dynamic Timing Analysis (DTA). The concept of timing simulation is to apply input vectors as stimulus on the input signals, observe the resulting behavior, advance the timing with new set of input vectors and observe the new resulting behavior and so on. One of the key features of timing simulation is that along with the analysis of the timing aspect of a design, the functionality of the circuitry is also verified extensively, which may not be required as the functionality and logic checks are effectively carried out at the Register Transfer Level (RTL) level. Such a verification of functionality at the late design stage may consume a substantial amount of time to carry out timing analysis which may unwantedly increase the time to market for an SoC. Moreover, timing simulation verifies only specific portions of the design which gets exercised by the stimulus. Ideally, verification through timing simulation is only as exhaustive as the test vectors used. Also, the process of simulation is too slow since a single design contains millions of gates. In other words, verification of timing checks by timing simulation offers a much slower runtime which in turn in modern day VLSI design affects the time to market critical for an SoC. Hence timing simulation as a technique is not widely practiced today for timing analysis. But since the input vectors are given during the timing analysis and the functionality is also verified, timing simulation is found to be more accurate than STA. STA is the most frequently used and modern day adapted technique for exhaustive timing analysis. This technique has multiple advantages which has been the key reason why it has become the most popularly used method for timing verification in modern day VLSI design. The name static defines that the technique is input independent and works based on the constraints being applied for the particular design rather than the input vectors. This method is a faster and simpler way of timing verification and is capable of verifying millions of gates in a shorter amount of time. STA is static because the analysis is statically (by keeping the design stable) and does not depend upon the inputs being applied at the input pins of the design. It is complete and exhaustive verification of all timing checks of a design. Hence with its advantages over the other methods, STA is the widely used method for timing verification.

The main aim of this master thesis is to implement the STA run for multiple partitions in a

design. For a design at block level, a PrimeTime session is generated by the STA run for each blocks. The aim is also to investigate and implement various timing checks available in SoC backend design flow for a VLSI system on a chip design at the block level through exhaustive timing analysis. Synchronous checks such as setup and hold checks are carried out at the data paths for multiple endpoints and slack are met. At a later stage, the transition and capacitance violations are also addressed and also the reason behind such violations and implementing fixes for the same are also done.

1.3 RESEARCH OBJECTIVE

To gain knowledge on the basic concepts related to timing closure and better understanding of the concepts. A brief literature survey must be done on the timing concepts and also on the methodologies and techniques used for meeting the timing requirements of the design. There are several tools used for verifying and optimizing the timing of a design. Synopsys tools (PrimeTime, IC Compiler) are used in this project so a good knowledge of those tools much be achieved. Better understanding of the stages at which timing optimization and timing violation fixes are performed. There are basically two stages: pre-signoff/ICC stage and signoff/post-ICC stage. Able to implement an efficient floorplan such that the timing is optimized as much as possible. Also, should be able to do better placement and routing. The basic ideas, techniques and methodologies used in this stage must be known. IO port optimization, bound creation and Clock Tree Synthesis (CTS) are some of the techniques used in the pre-signoff stage. To perform STA on the design in order to obtain the timing violations (setup and hold) occurring in the design. To fix the setup and hold violations by writing appropriate Engineering Change Order (ECO). Must be able to analyze a timing path and write ECOs for fixing the timing violations. Better understanding of the ECO should be achieved. Should be able to tell whether a path is fixable or unfixable just by analyzing the timing path. Cell sizing, VT swap, buffer insertion and removal of buffer are some of the techniques used in the ECO. Full understanding of these techniques helps to write an ECO that can be sourced and routed. Analysis of the buffers and understanding the behavior of buffers at different corners helps to fix hold violations because hold violation fix involves insertion of the buffers. To find the discrepancies in the slack value of a path in block level and full chip level. This analysis helps to make the slack equal at block level and full chip level. Finally the ultimate objective is to validate the design whether it can operate in the rated speed or frequency.

1.3.1 Physical Design

Physical design is used to describe the geometries (typically through CAD) used to represent materials used in VLSI design. For example, polygons and rectangles are used to represent silicon and metal layers on chips.

Physical design is tends for the transforming or action of to convert Netlist into layout which is manufacture-able. Physical design process is often referred as PnR (Place and Route) or APR (Automatic Place & Route).

Floorplan is the first and the most reproving step in Physical design. Timing, Noise, IR and Routing issues. A bad floorplan will blow up the area, power and affects reliability and life of

the IC and also increases the overall IC cost.

Most of the Place and Route tools generally provide an automatic floorplan option. Automatic floorplan option is developed its own macro placement based on the effort & other options. But these options are not good enough to give optimum floorplan for all kind of designs. This option will be handy, when design has 100s of Macros, but generated floorplan needs a lot of modification for further optimizations.

Next is the Placement stage where all the standard cells are placed in the design which typically includes the size, shape & macro-placement. Placement will be driven by different types criteria like timing driven, congestion driven, power optimization in upf etc. Strategy of Timing & Routing convergence depends a lot on quality of placement of the design. Clock Tree Synthesis abbreviated as CTS is one of the most important stages in Place and Route. The Quality of Result of CTS decides timing convergence power. In most of the ICs, clock consumes 31-41 %of total power. So the efficient clock architecture , clock gating & clock tree implementation all those helps to reduce power. During Clock tree synthesis, buffers or inverters are added in the clock nets to achieve minimum insertion delay and skew, while meeting the clock DRVs. Various executions are performed during CTS such as CCDO (Concurrent Clock and Data Optimization) and CTO (Clock Tree Optimization). Once the Clock tree synthesis optimizations are done, the clock tree is fixed and routed. Further optimizations cannot be done on the clock tree except either buffer sizing (upsizing and downsizing) or gate sizing. Hence, post CTS, only data path can be optimized. The various post CTS optimizations include meeting DRVs, Setup check & Hold check , Area & Power optimization, Congestion reduction. Routing create physical connections to all the data signal if clock nets are already routed after clock tree optimization. There are three stages in routing namely Global routing, Track assignment and Detailed routing. Once the routing is completed and the parasitics are extracted, the timing analysis is done which we will be the area of implementation for this project which we will be looking in detail into in the next sections.

1.4 Project Area

1.4.1 Timing Closure

1.4.2 Choosing a Strategy for Timing Closure

There is no single strategy that ensures quick and easy timing closure; however, a strategy based on the size and number of timing violations can be useful.

1. To meet the timing requirements of a Logic design by applying certain modifications to the components in the design.

2. To perform Static Timing Analysis (STA) on the design in order to obtain the timing violations(setup and hold) occurring in the design.

3. To fix the setup and hold violations by writing appropriate Engineering Change Order (ECO).

4. To validate the design whether it can operate in the rated speed (frequency).

   Timing closureis the process by which a logic design consisting of primitive elements such as combinational logic gates and sequential logic gates is modified to meet its timingrequirements. The modifications include, Logic optimization of the combinational elements (sizing of the cells). Pulling and pushing of the clock (insertion of buffers in clock path).

1.5 Static Timing Analysis

To meet the timing requirements of a logic design by applying certain modifications to the components in the design.

To perform Static Timing Analysis (STA) on the design in order to obtain the timing violations(setup and hold) occurring in the design.

To fix the setup and hold violations by writing appropriate Engineering Change Order (ECO).

To validate the design whether it can operate in the rated speed (frequency).

Timing, area, and power constraints drive the operation of synthesis with Design Compiler Topograpical and physical implementation with ICC2 Compiler.

1.5.1 Static Timing Analysis Using PrimeTime

Creating a PrimeTime session for a block

The PrimeTime tool reads a gate-level netlist (SDC)File from the synthesis or physical implementation tool together with the UPF descriptions generated by those tools.

1.5.2 Timing Paths

Every design is designed to operate at a particular frequency.

This frequency attribute determines the operating speed of the design and becomes the timing aspect.

But designing a circuit for a frequency is not as easy as it is said.

A lot of timing issues and constraints arise while designing.

Timing closure is the process of fixing all these timing issues and closing the design with respect to the timing.

To identify the timing issues and fixing them, Static Timing Analysis(STA) is used.

To analyze a particular block, first we need to create a PrimeTime session for the block. The PrimeTime tool reads the gate-level netlist and parasitic data, and verifies the design timing using information provided in the logic (.db) library. The PrimeTime takes gate level netlist, design constraints in the form of SDC file, parasitic data from the parasitic extraction post routing in the form of SPEF, logic library containing physical data and information for physically aware ECO generation to generate a PT session.

1.5.3 Analysing the REG to REG paths

Every timing path will have a start point, end point and combinational cloud between them. The initial steps include launching the PrimeTime shell. You can start a PrimeTime shell in either the command line interface or the graphical user interface. For the command line interface, enter the following command at the Linux shell prompt: % pt_shell

1.5.4 Steps

For the analysis of paths of a particular block, first we have to restore a particular session using the respective paths where the session is generated.

A session path is a .analyzedesign link which is specific to an operating condition including the process, voltage and temperature. Hence restoring a particular session path lets us check and resolve the violations in that particular PVT corner. This is done by the restore_session ¡session path¿ command. Once the session is restored we can start analyzing the reports. Cell delay is the time taken by the logic to propogate from the clock pin to the output pin. Net delay is the time taken by the logic to propogate from the output pin of a cell to the input pin of the next cell in the logic path.A report_timing command gives the report of a path having the worst negative slack in the entire design and a path report which gives the clock and output pins along with delays.

Some of the switches we can use along report timing are as mentioned below

1.5.5 -delay_type

Specifies the type of path delay constraint to consider for finding and sorting paths with the worst slack:

max (default) - max delay (setup constraint)

min - min delay (hold constraint)

1.5.6 -nworst_paths per endpoint

Reports up to the specified number of worst paths per endpoint. The default is 1. A larger value results in a larger report and more runtime. Allowed values are 1 to 2000000.If you set -nworst to any number greater than 1, the command automatically does the following: Implicitly sets -max paths equal to the -nworst setting if the -max_paths option is not used in the command, so that at least one set of multiple paths to an endpoint can be reported.

Implicitly sets -slack_lesser_than 0.0 if the -slack_ lesser_than and -slack_greater_than options are not used in the command, so that only violating paths are reported.

1.5.7 -max_paths , max_path_count

Reports up to the specified maximum total number of paths among all path groups. The default is equal to the -nworst setting, or 1 if the -nworst option is not used. Allowed values are 1 to 2000000.

1.5.8 -slack_greater_than minimum_slack

Reports only paths with slack greater than the specified minimum slack value; these paths have a negative slack better than the specified minimum slack value (or a positive slack that is farther from causing a violation). This option is intended to be used with the slack lesser than option to report paths within a specific range of slack.

1.5.9 -slack_lesser_than maximum_slack

Reports only paths with slack less than the specified maximum slack value; these paths have a negative slack worse than the specified maximum slack value (or a positive slack that is closer to causing a violation).

1.5.10 -input pins

Shows cell input pins as well as cell output pins in the timing path. As a result, the report shows the incremental net and cell delays separately at each point, instead of combined. By default, the report shows only the cell output pins.

1.5.11 -nets

Shows nets in the timing path. By default, the report does not show nets.

1.5.12 -nosplit

Prevents line splitting. This can useful for scripts that extract information from the report. By default, the report generates a new line when the text cannot fit in the allotted space in a column.

1.5.13 -transition time

Shows the transition time (slew) in the path report for each driver pin and load pin, appearing as an additional column labeled ”Trans”. By default, the report does not show transition time.

1.5.14 -capacitance

Shows the total capacitance in the path report for each net, appearing as an additional column labeled ”Cap”. By default, the report does not show capacitance. The reported value is either the CCS receiver capacitance (the default) or the lumped capacitance, depending on the report capacitance use CCS receiver model variable setting.

The timing issues are called as Timing violations.

When performing timing analysis, PrimeTime first breaks down the design into timing paths. Each timing path consists of the following elements:

(a) Startpoint The start of a timing path where data is launched by a clock edge or where the data must be available at a specific time. Every startpoint must be either an input port or a register clock pin.

(b) Combinational logic network Elements that have no memory or internal state. Combinational logic can contain AND, OR, XOR, and inverter elements, but cannot contain flip-flops, latches, registers, or RAM.

(c) Endpoint The end of a timing path where data is captured by a clock edge or where the data must be available at a specific time. Every endpoint must be either a register data input pin or an output port.

1.5.15 Different Types of Timing Paths

Flop to Flop Paths
Input to Flops Paths
Flop to Output Paths
Input to Output Paths

Chapter 2

BACKGROUND AND RELATED WORKS

2.1 OVERALL DESCRIPTION

Any VLSI system design flow starts with the formal specification followed by a series of steps carried out one by one in a sequential flow to finally produce a packaged IC chip. A typical design cycle is as represented in the flow chart as shown in Figure 2.1.1 The main emphasis of this master thesis is the timing closure at block level

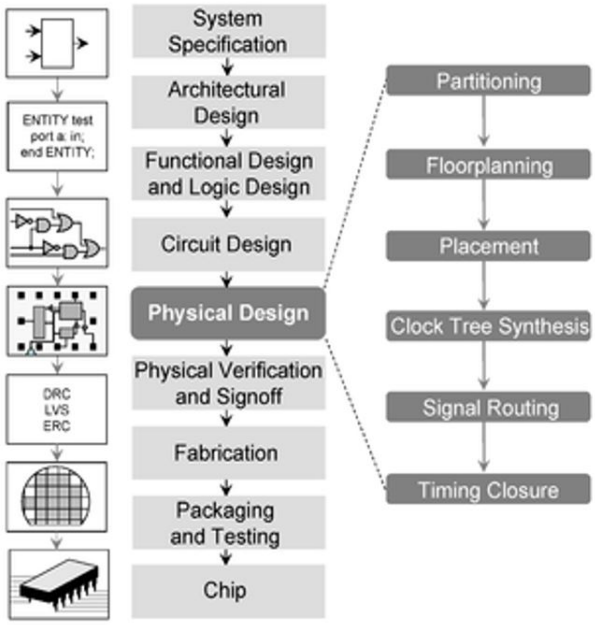

Figure 2.1.1: VLSI System Design Cycle

which is a part of the Physical Design stage of the design cycle. Also the entire

physical design of a block will also be done to understand the optimization techniques used in the design flow. The timing optimization by performing floorplan and placement is done here. The first and foremost step in any system design is to lay down the specifications of the system. System specification, in general is a high level representation of the system. Architectural design describes the basic architecture of the system. Some of the architectures may be RISC (Reduced Instruction Set Computer) or CISC (Complex Instruction Set Computer). The system architecture also defines the number of Arithmetic Logic Units (ALUs), structure of pipelines, size of caches etc. Next up, functional design identifies the main functional units and the interconnects between them in a system. In Logic Design, the control flow, register allocation, arithmetic operations and logical operations of the design that represent the functional design are derived. Logic Design involves formulation of the design in the form of Boolean expressions at the RTL using Hardware Descriptive Languages (HDLs) such as Verilog, VHDL etc. which can be used in simulation and verification. Once after implementing the design using HDL, the functionality of the design has to be verified. The objective of circuit design is to generate and develop a circuit based representation of the logic design. Most often the circuit representation is referred to as the netlist. Once the netlist of the design is there, steps like Synthesis, Partitioning, Floor-planning, Placement, Clock Tree, Routing and Timing Closure can be carried out. All the SoCs used today are synchronous in nature i.e. it is timed or it has a reference clock. Typically there is a single clock in many systems or there can be multiple clock phases which are generated from a single external source. Hence all the activities of the chip are synchronized with respect to the clock that is applied from outside typically. So the clock signal is used to synchronize the storage elements such as the flip flops. The clock provides a common reference signal distributed throughout the chip. Since it is distributed throughout the chip, the total length of the clock net can be pretty large which also the reason the clock is often referred to as a high fanout net. Logic evaluation begins at the rising edge of the driving clock. This is due to the fact that when we receive the rising edge of driving clock, the input coming from previous stage gets stored in the register. To understand the timing of a digital design, the need for clock in a design has to be understood. In a digital circuit, nothing is standalone. Every component is interdependent, that is the outputs of one logic may be input to many other logics. Wherever there is interdependency, sequencing of the events is necessary and we should have a control over the events. So a control signal is required to control the events in the digital design. The clock has many components in order to be defined. Time period, waveform, source and name are some of the components. There are three important needs for clock.

- Acts as a control input signal to the circuit.
- Sequences the events in the design.
- Determines the operating speed of the design.

2.2 LITERATURE SURVEY

STA cannot be done manually because there will be lakhs of timing paths in the design. The Electronic Design Automation (EDA) tools come to the rescue for this problem. The EDAs will be fed with libraries which will contain the delay information of each and every cell in the design. These delays act as constraints to the timing paths. In reality, the exact time of the arrival of clock signal is not known and also the chip behaviour. This is mainly because of the environment conditions. The delay information may vary at each and every point of the environment condition (Temperature). These variations have to be fed in the EDA tool while simulating so that after fabricating the chip may perform as it was meant to be designed. These are the reasons for constraining the design and STA checks whether the chip meets the constraints given. Constraints form the core part of timing analysis.

2.2.1 TIMING CLOSURE

The SoC must not only follow the geometric requirements but should also meet the designs timing constraints. The optimization process tasked to meet the timing requirements are collectively referred to as timing closure. Every design is designed to operate at a particular frequency. This frequency attribute determines the operating speed of the design and becomes the timing aspect. But designing a circuit for a frequency is not as easy as it is said. A lot of timing issues and constraints arise while designing. Timing closure is the process of fixing all these timing issues and closing the design with respect to the timing. To identify the timing issues and fixing them, STA is used. The timing issues are generally called as Timing violations. There are three steps in which the STA works. First the design is broken in to sets of timing paths. Then the signal propagation delay is calculated along each path. Finally, violation of timing constraints is checked. Dynamic Timing Analysis (DTA) was the technique used for timing analysis initially. STA is called static because the analysis is done when the design is stable. The design will be stable when there is no input given. DTA is dynamic because the analysis is done when the design is active. The design will be active when it is given inputs. DTA requires a comprehensive set of input vectors to check the timing paths in the design. It can verify the functionality as well as timing of the design. It is only as exhaustive as the input vectors are given because a particular test vector cannot exercise all the portions of the chip.

2.2.2 TIMING PATHS

The end user of a design will have only one concern regarding the timing of the design that is how long will it take for the output to change for a particular input given. This timing relationship between two pins of the device is given by timing arc. Basically, it represents the timing characteristics of the element or block. When performing timing analysis, PrimeTime first breaks down the design into timing paths. Each timing path consists of the following elements: Startpoint: The start

of a timing path where data is launched by a clock edge or where the data must be available at a specific time. Every startpoint must be either an input port or a register clock pin. Combinational Logic Network: Elements that have no memory or internal state. Combinational logic can contain AND, OR, XOR and inverter elements, but cannot contain flip flops, latches, registers or RAM. Endpoint: The end of a timing path where data is captured by a clock edge or where the data must be available at a specific time. Every endpoint must be a register data input pin or an output port. There are several timing paths which can be divided as per the type of signals such as data path, clock path, clock gating path and asynchronous path. Unconstrained paths do not belong to any of the path groups. To report unconstrained paths, set the timing report unconstrained paths variable to true. This variable enables us to view the unconstrained paths in the design. Every timing path consists of a startpoint, a combinational logic network and an endpoint. The time taken by the data to propagate through the logic network is referred to as the delay. This comprises of the sum of delays of the cells as well as nets along the path. For a single startpoint endpoint pair, there are multiple paths through which the logic can propagate to the required destination point. The actual path taken depends fully on the state of the other inputs along the logic path. Since there are multiple paths for a single startpoint endpoint pair, the maximum and minimum time for the logic to propagate throughout he specific startpoint endpoint pair can be obtained. The path that takes the maximum time for the logic to propagate is referred to as the max path. Similarly, the path that takes the minimum time for the logic to propagate is referred to as the min path. In other words, a max path is often referred to as the longest path whereas a min path is referred to as the shortest path. There are four types of timing paths.

- Flop to Flop Paths
- Input to Flops Paths
- Flop to Output Paths
- Input to Output Paths

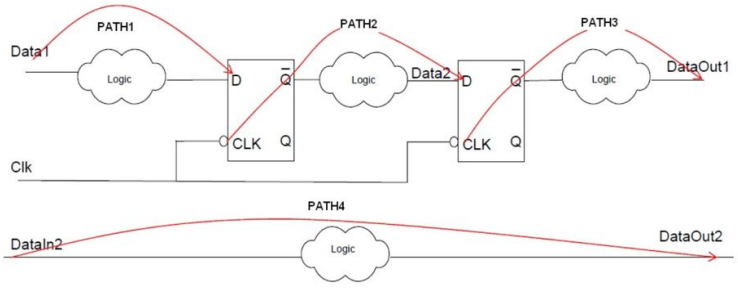

Figure 2.2.1: Four types of Timing Paths

Path1 represents the Input to Register path as it starts at an input port and ends at the data input of a sequential element, Path2 represents the Register to Register path as it starts at the clock pin of a sequential element and ends at the data input pin of

a sequential element, Path3 represents the Register to Output path as it starts at the clock pin of a sequential element and ends at an output port and Path4 represents the Input to Output path as it starts at an input port and ends at an output port. Except Input to Output path, all other paths are synchronous paths.

2.2.3 STA CONCEPTS

Timing arc represents the timing relationship between two pins of any device, block or any boundaries. To understand timing arc the design has to be seen from end user point of view. An end user may be mobile phone user, laptop user, IP consumer or standard cells consumer. The end user will have only one concern regarding the timing and that will be how long it will take to get the output after applying certain input. So, the end user is concerned about the relationship between different pins/ports of a product. After a lot of simulation and characterization the relationship between different pins in terms of timing is obtained. Basically, timing arc represents the timing characteristic of an element or block. There are two types of timing arcs based on delay and characteristics: Delay arc and Constraint arc. A timing arc has, Startpoint Input pin, Output pin or Inout pin and Endpoint Always an Output pin or Inout pin.

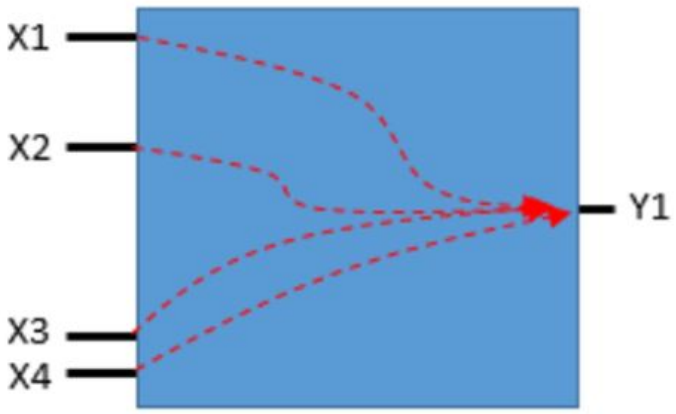

Figure 2.2.2: Input to Output Timing Arc

The black box in figure 2.3 is having four input pins and one output pin. Hence, four timing relationships can be found in this black box. X1-Y1, X2-Y1, X3-Y1 and X4-Y1. The end user will never ask what is inside the black box if the timing information is provided. But inside the black box there will be nets and cells because of which there will be two arcs: Cell arc and Net arc.

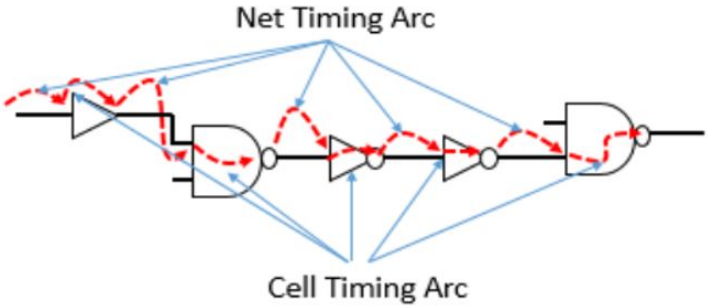

Figure 2.2.3: Net and Cell Timing Arc

Timing arc answers the following questions,

- For a particular type of input (rising or falling), what type of output is obtained?
- How much time (delay) the cell will take to respond for a particular input?
- Is there any constraint on any pin?

The first question can be understood by knowing how the pin is logically connected with the output pin. Logical connection means what will happen for rising input whether the output falls or rises or remain unchanged. Timing arc answers this with a property known as Unate. The unateness specifies how the edges (transitions) can propagate through a cell and how they appear at the output of a cell.

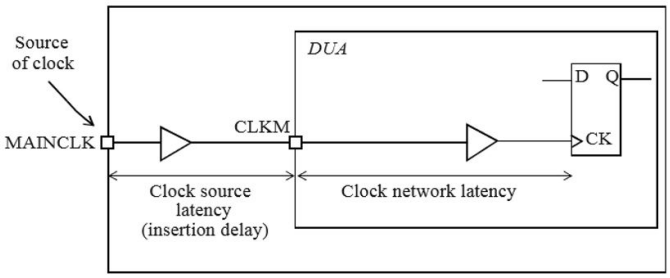

Figure 2.2.4: Source and network latency

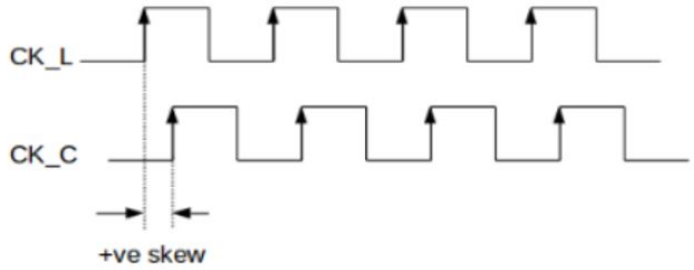

Figure 2.2.5: Positive Skew

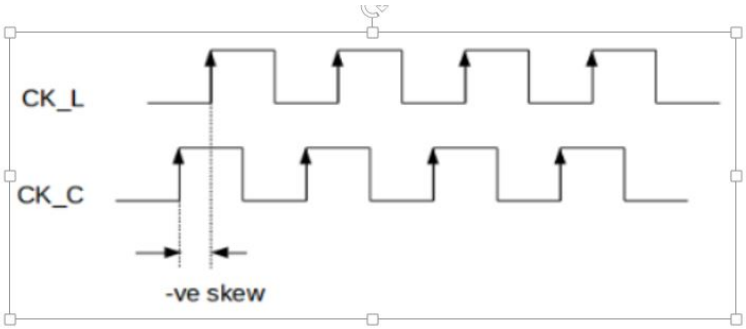

Figure 2.2.6: Negative Skew

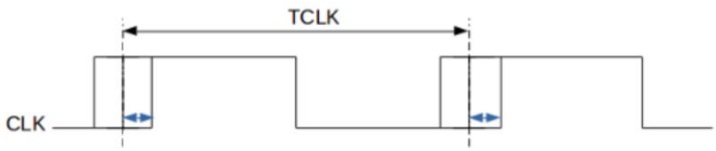

Figure 2.2.7: Uncertainty in Clock Pulse

It basically constraints the Input Register and Output Register timing path. The desire latency can also be specified.

PVT (Process, Voltage and Temperature) combination defines the operating condition of the design. The chip should work at even -40C and 100C. Delay is the most important factor in the timing of a circuit. Different combination of PVTs give rise to different delays. A timing path will have different delays for different corners. The design is simulated at different corners of PVT which IC may face after fabrication.

This affects the oxide thickens and causes fluctuation in dopant and mobility. The transistor width and length is also affected. The relationship between process variation and delay. current flowing through the parasitic inductance causes voltage bounce because of the supply noise caused by it. Usually, a chip will be powered by a battery or voltage regulator. A battery cannot provide constant supply for long

period of time and a voltage regulator can also not be consistent for over a period of time. The relationship between delay and voltage. Temperature around the chip varies in a big range, which will affect the delay factor of the design. Hence it is taken in to consideration. When temperature increases, the mobility and threshold voltage decreases. Delay increases with decrease in mobility and decreases with decrease in threshold voltage. In sub-micron technologies, a phenomenon called temperature inversion happens . When temperature increases, the delay will increase. But when the temperature starts to fall, for some time the delay will decrease. But after one point the delay will start to increase again because the threshold voltage will take upper hand and becomes the deciding factor. Previously, the mobility of electrons would have been the deciding factor. This phenomenon is seen only in the deep sub-micron technologies.

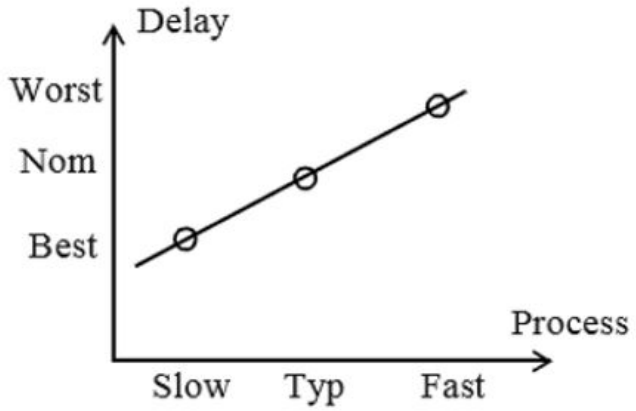

Figure 2.2.8: Delay vs Process

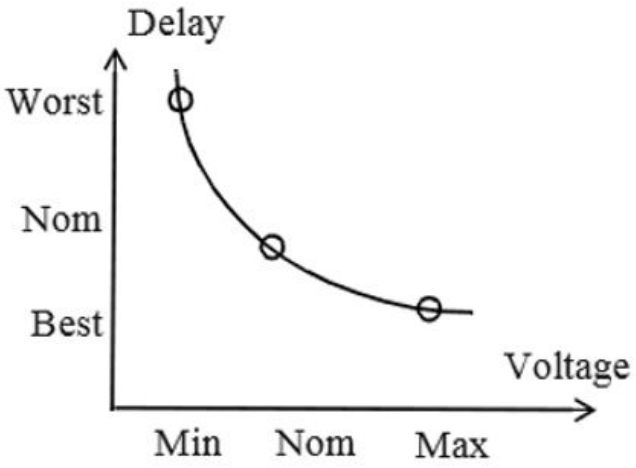

Figure 2.2.9: Delay vs Voltage

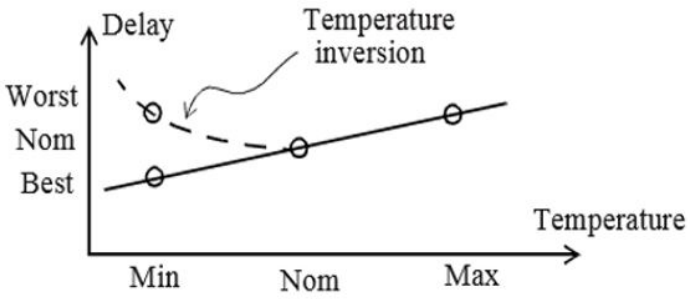

Figure 2.2.10: Delay vs Temperature

The setup timing and hold timing check constitute the primary checks for any design.

Conditions	Setup	Hold
Process	Slow	Fast
Voltage	Low	High
Temperature	High	Low

Figure 2.2.11: Critical Corners for Setup and Hold

On Chip Variations (OCV) are PVT variations that are seen at transistor level.

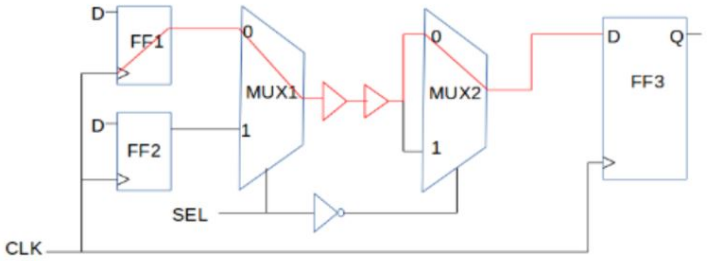

Figure 2.2.12: False Path

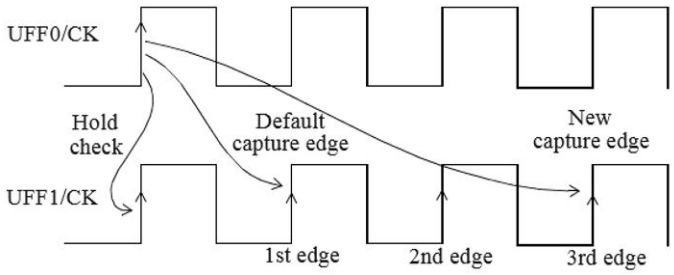

Figure 2.2.13: Multicycle Path

2.2.4 CLOCK TREE SYNTHESIS

Once the clock trees are built, it undergoes several buffer insertions and subsequent skew optimizations. An important point to be noted in the case of CTS is the use of clock buffers in clock path instead of normal buffers. Minimum pulse width is an important Design Rule Check (DRC) check performed on any design. It is required to maintain a minimum pulse width for proper functioning of the entire circuit.

The changes in the pulse width can be attributed to the unequal rise and fall transition delay caused by the circuit. The fall time transition in a clock signal is affected by the width of NMOS. On the other hand, the rise time transition in a clock signal is affected by the width of PMOS. Since NMOS has higher mobility than PMOS due to the presence of free electrons, the fall transition is lesser than rise transition. Hence the rise time is more than the fall time. If a series of unbalanced buffers are connected in a circuit, the fall time decreases and rise time increases each time the clock signal passes through a buffer. In such a scenario, after a certain amount of time, the clock signal gets absorbed due to reduced pulse width. Hence, to make rise time equal to fall time of a clock signal, clock buffers are introduced in which we the width of PMOS is 2-2.5 times the width of NMOS providing a balanced buffer. An input signal of 50

2.2.5 Skew Optimization

The propagation delay is controlled by the size and location of the buffers.

2.2.6 Buffer sizing

A binary search algorithm is implemented to find the appropriate size of clock buffers through an iterative process since it affects downstream optimization's.

2.2.7 Wire sizing

The size of wire is an important factor as it affects the power consumption and is prone to manufacturing issues.

2.3 CROSSTALK AND NOISE

Any undesired or unintentional effects affecting the nominal operation of the chip is called noise. Crosstalk noise can be referred as unintentional coupling of activity between two or more signals. It is mainly caused by the capacitive coupling between two neighbouring signals. In nanometer devices, this noise can impact the functionality of the device or even the timing of the device. There are several causes for this noise in the chip. The number of metal layers in the chip nowadays have been increased drastically, because of the technology shrinking. Also, the wires that are used nowadays are thin and tall rather than wide and tall because of which capacitance will be more between two neighbouring wires. The standard cell count is also increased in latest technologies which increases the congestion and causes lot more interactions. Higher frequency designs have faster edge rates, which will cause more current spikes and greater coupling impact. Noise margin for the designs is little because of low supply voltage.

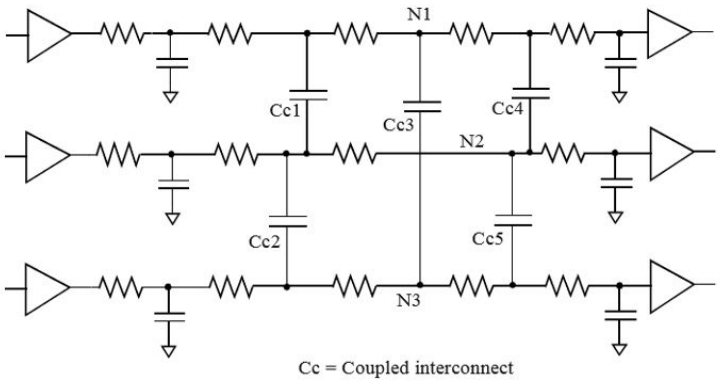

Figure 2.3.1: A Coupled Interconnect Example

The capacitive coupling happens between two signal nets as shown in figure 2.14 and causes switching activity on one of the nets. The signal net that gets affected is called Victim and the signal net that affects the other net is called the Aggressor. Cc1 and Cc4 are coupling capacitances between the nets N1 and N2. Cc2 and Cc5 are coupling capacitances between the nets N2 and N3. There are two types of noise effects caused by crosstalk: Glitch and Delay.

2.3.1 CROSSTALK GLITCH

In a steady victim signal if the coupling of switching activity of neighbouring aggressors causes a noise, then it is called as crosstalk glitch. It basically affects the functionality of the design. There are two types of glitch based on the polarity: Positive glitch and Negative glitch. If the glitch is caused by a rising aggressor then it is called positive glitch and if the glitch is caused by a falling aggressor then it is called negative glitch. The magnitude of the glitch depends upon, coupling capacitance between aggressor net and victim, aggressors slew, grounded capacitance of the victim net and driving strength of the victim net. There are four types of glitches based on the aggressor and victim nets signal position. They are mentioned in figure 2.15. The rise glitch is caused when the victim is low and the aggressor is transitioning from 0 to 1. The fall glitch is caused by an aggressor transitioning from 1 to 0 while the victim is already high. Similarly, the overshoot happens when victim is high and aggressor goes from 0 to 1 and undershoot happens when the victim is low and the aggressor goes from 1 to 0. Not all glitches are hazardous. A glitch is differentiated as hazardous or non-hazardous based on its parameter. It has two parameters: Height and Width. Figure 2.16 shows the safe glitches and potentially hazardous glitches. As long as the glitch height remains under the margin shown in figure 2.16, it will not affect the design. Also if the width of glitch is more, then it is also considered to be safe.

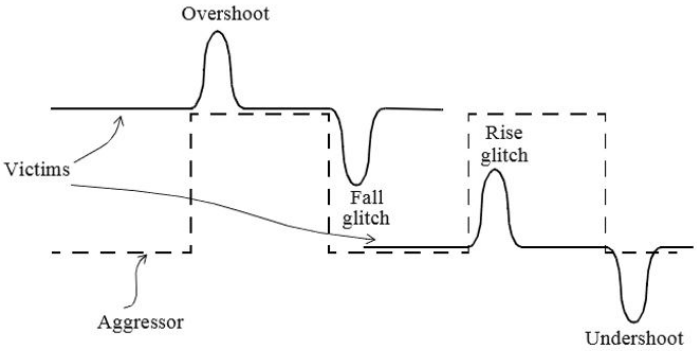

Figure 2.3.2: Types of Glitches

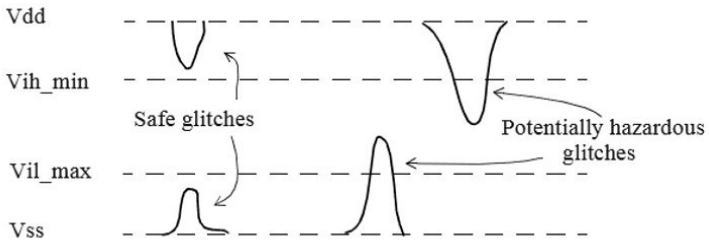

Figure 2.3.3: Glitches based on Noise Margin

2.3.2 CROSSTALK DELAY

Crosstalk delay refers to the change in timing caused by the switching activitys coupling of the aggressor net on the victim net. This basically happens when both the aggressor and victim nets are transitioning. But in the case of glitch victim will be steady. The delay is caused by the charging current through the coupling capacitance when the neighbouring net is switching. The direction of aggressor net switching determines the magnitude of the delay induced. Without any crosstalk if the delay is calculated, then the total coupling capacitance is provided by the cell driving the aggressor net. The charge required for coupling capacitance will be more if the aggressor and victim are switching in opposite direction. Based on the magnitude of delay induced by the aggressor net, there are two types of delay: Positive crosstalk and Negative crosstalk. Positive crosstalk delay is a scenario in which the aggressor will be transitioning from low to high and the victim net will be transitioning for high to low. This opposite direction switching increases the delay for the victim net. This impacts both the driving cell of the net and also interconnect by increasing the delay in both of these. the positive crosstalk delay in which the delay increase has been shown, thereby causing timing error. Negative crosstalk delay is caused when both the aggressor and victim are switching from low to high. This kind of transitioning reduces the delay on the victim net. It impacts both the driving cell and interconnect by reducing the delay in both. A depiction of the negative crosstalk delay. Both the crosstalk delays affect the timing.

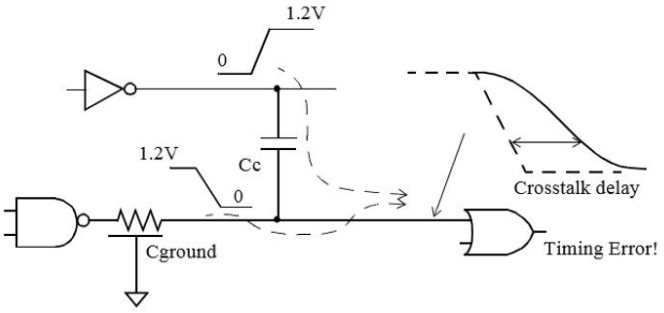

Figure 2.3.4: Positive crosstalk delay

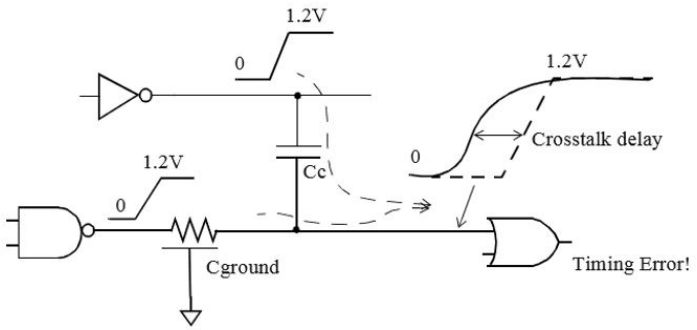

Figure 2.3.5: Negative crosstalk delay

2.3.3 PRIMETIME STA RUN

i. Read in the design data, which includes a gate-level netlist and associated logic librarie.
ii. Specify timing and design rule constraints.
iii. Specify clock characteristics.
iv. Specify timing exceptions.
v. Specify the environment and analysis conditions such as operating conditions and delay models.
vi. Specify case and mode analysis settings.
vii. Back-annotate delay and parasitics .
viii. Apply variation (optional).
ix. Specify power information.
x. Specify options and data for signal integrity analysis.
xi. Apply options for specific design techniques.
xii. Check the design data and analysis setup.
xiii. Perform a full timing analysis and examine the results.
xiv. Generate engineering change orders (ECOs) to fix timing violations or recover power.
xv. Save the PrimeTime session.

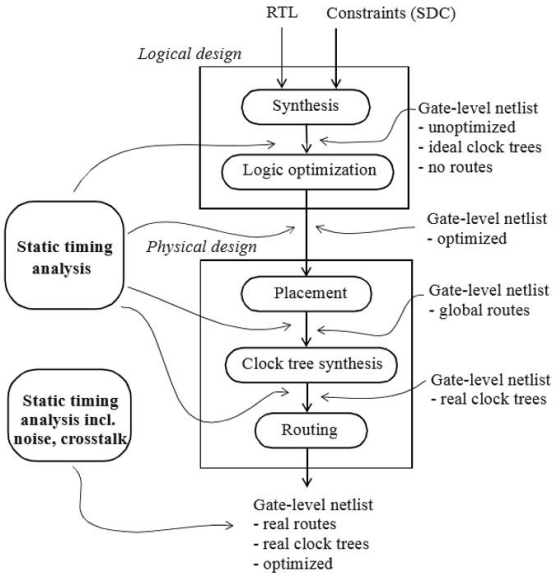

Figure 2.3.6: STA in Design Flow

2.3.4 ENGINEERING CHANGE ORDER

In the PrimeTime Suite, an Engineering Change Order is an incremental change in chip design to reduce the various timing violations. The ECOs are implemented after loading the PrimeTime sessions of the particular blocks post the STA run. The violations include DRCs, timing violations and power violations. The purpose of PrimeTime Suite is to search and find these violations and to provide fixes for these issues in the design. The methods adopted to fix these violations include cell resizing, replacement of cells or inserting buffers. The types of violations to be fixed primarily include setup hold violations and DRC noise violations. The primary commands used to generate ECOs for fixing violations are fix eco drc, fix eco timing and fix eco Power. These PrimeTime commands are used for fixing violations and optimizing the design by improving the timing requirements providing ECO fixes. While implementing fixes for violations, one of the important factor to be taken care of is the order in which the fixes should be carried out. This is due to the fact that timing may degrade in related paths due to the fixes made or ECOs implemented. It is also seen that setup and hold fixing does not degrade DRC violations such as max capacitance and max transition violations, Whereas ECOs implemented to fix DRC violations can degrade setup or hold violations, since DRC fixing has the highest priority. Since hold fixes preserves setup slack and setup fixes cause a slight increase in the hold violations, it is always seen that the DRCs fixes are given the highest priority followed by setup violation fixes and then hold violation fixes.

2.3.5 UNFIXABLE VIOLATIONS

As a matter of fact, all the existing violations cannot be fixed by the tool itself due to some or the other design constraints. The fix eco timing and fix eco drc commands feature a verbose switch which when exercised reports the unfixable violations and the reasons that they are unfixable. It may also be noted that to use this feature alongside the fix eco timing command, the eco report unfixed reason max endpoints variable has to be set to a positive integer. A timing ECO unfixable violation reason report can be generated without performing ECO changes which may provide feedback and guidance on how to address these unfixable violations. To get a full estimate of all the unfixable violation reason report, the fix eco timing command with the estimate unfixable reasons switch can be used. The unfixable violations are often indicated by pre-fed symbols or alphabets in the report generated that gives the specific reason. Some of the reasons can be summarized as given below,

A There are available library cells outside the area limit.
B Delay improvement is too small to fix the violation.
C - The violation is in clock network.
I Buffer insertion with the given library cells cannot fix the violation.
S Cell sizing with alternative library cells cannot fix the violation.
T Timing margin is too tight to fix the violation.
U UPF restricts fixing the violation.
V Net or Cell is invalid or has dont touch attribute.
W Fixing the violation might degrade DRC violation.

Chapter 3

PHYSICAL DESIGN

For every VLSI systems, physical design is a process of instantiating each and every design components with the corresponding geometric representations. The physical design flow starts once the design and verification teams complete the functional and logic design and the design is thoroughly verified for bugs and errors. Essentially, physical design plays a significant role in the performance of the SoC due to factors such as performance, power, area etc. The various parameters that affect the competence of an SoC in the market are speed which is derived from the operational frequency, the area it occupies since smaller SoCs can be integrated into a variety of mobile devices and power since low power consumption leads to longer usage time all of which factors can be optimized to achieve a better yield during the physical design stage. Physical design is split into several stages namely Partitioning, Floorplanning, Placement, Clock Tree Synthesis, Routing, Timing closure etc. Each of the steps are carried out in succession beginning with Partitioning and so on. Physical design is given as a chapter in this thesis because timing optimization is done during this process.

3.0.1 Partitioning

The process of splitting up the entire circuit or system into smaller subsystems or modules is referred to as partitioning. Floorplanning The process of arranging the location of hard IPs or macros, the external ports and sub circuits or modules is referred to as Floorplanning. Placement The process of finding and determining the spatial location of each and every cell within a particular block is referred to as Placement. Clock Tree Synthesis The process determining the buffering, gating and routing of the clock signal to meet the required clock skew and clock delay estimates is referred to as CTS. Routing The process of allocating resources for interconnections between different metal layers and routing tracks in channels is referred to as Routing. Timing Closure The optimization of performance of the SoC is carried out by specialized placement and routing techniques and implemented ECOs is collectively referred to as timing closure.

3.0.2 COMPONENTS IN THE DESIGN

Macro or Hard IP:

The functional logic in a design are broadly classified as Soft IPs and Hard IPs. The Soft IPs are referred to as reconfigurable logics which can be changed or modified as per the user requirements and design constraints as and when required. The Hard IPs are large sets of non-reconfigurable logic which perform a reusable functionality. Usually, they possess read only characteristic in the design. A macro cell can vary in size anywhere from a couple of standard cells to a large embedded processor or a memory block such as RAM. Technically a macro is a cell without pre-defined geometric dimensions. The macros can be placed almost anywhere in the entire area of the block. Placing the macros are usually dependent on factors to ensure minimum routing distance to the ports.

Standard Cells :

The design of each SoC is carried out using different methodologies such as the Standard Cell methodology or Full Custom methodology. The current methodology implemented for this SoC is a standard cell methodology. This methodology involves the use of pre-designed standard cells fed in the library contrary to the Full Custom methodology. A standard cell is a group of logic gates that perform a Boolean function. It has a pre-determined functionality. Its dimensions are a multiple of library specific fixed dimensions.

Via :

Via is a connection between metal layers usually used to connect routing structures on different metal layers. Pin is an electrical terminal used to connect a component to the external environment and it can be an input pin or an output pin based on the functionality which they serve. Port represents a group of pins interacting with the external world/environment. It can be in, out or inout based on the directional functionality. Net is a wired interconnect indicating a physical representation of connection between two cells in a layout. Metal layer is a process level in SoC manufacturing patterning the different components in a design. During physical design, the components of a design are assigned to different metal layers.

Voltage area :

Voltage area is also called as power domain. It is a specific area in the design, which is connected to a particular voltage level. It is used in multi-voltage design, where different voltages are used for low power purpose. Bounds are implemented at specific areas in the design where there is a requirement of a different power supply other than the existing one or when there is a need to

confine certain amount of cells. For example, where ports are to be powered by the corresponding power supply other than the parent supply.

3.0.3 INPUTS REQUIRED FOR PHYSICAL DESIGN

Synthesis and APR are the two important steps in physical design. The entire process requires a proper set of inputs without which the design is not possible. The inputs will come from different teams. The design formulated as the RTL logic code is verified and released by the Front End to Back End team. The clock information comes from the clock team. Full chip information will also be needed for layout data. So a full chip layout team will provide the layout information (boundary and IO port placement). And a full chip netlist team will provide the additional RTL information needed. The tags for all these information are kept in modelling file. Since physical synthesis is done, the placement information of the cells, IO ports and other components of the design are given. The APR requires certain inputs from the output of synthesis. Netlist, UPF, SDC, DDC and standard cell DEF are some of the important inputs required. APR also requires the placement information that were used in synthesis.

3.0.4 SYNTHESIS

The technology independent Register-Transfer Logic is synthesized and converted to technology dependent netlist satisfying constraints during the process of synthesis. The Synopsys Design Compiler and Design Compiler Topographical are used to carry out synthesis. The Design Compiler deals with only logical synthesis whereas the DC Topographical performs physical synthesis. The inputs required for synthesis include logical libraries, physical libraries and timing scan constraints. The logical libraries include GTECH, Symbol and DesignWare libraries. The GTECH is a virtual internal library of Synopsys Inc. After the elaborate step in synthesis, the design gets mapped to this virtual library before it is mapped to a technology library. This library describes the basic functionality of the cell and comes handy to understand how the design gets mapped in the early stages itself. The Symbol library has different versions for different EDA vendors and is also an internal library of Synopsys Inc. This library contains different symbol shapes of the various cells that are mapped to the GTECH cells and which at a later stage would be mapped to the tech library. The physical libraries include DesignWare library and the standard cell library. The DesignWare library gives practical implementation for any and all of the complicated function for example, multiplier, divider etc. Similarly the standard cell library is a technology dependent library provided by the foundry to the designer. It contains the logical, timing and physical information about all the cells that are used during the compile stage. There are certain important terminologies in the synthesis process. Search path: The path to the library or lib files are specified in the search path. The Design Compiler

uses this path to search for the libraries that are specified in the design. Target Library: During the compile stage of synthesis, the Design Compiler uses the libraries specified in this variable. Link Library: The references of all the files in the design are resolved using the libraries specified here during linking. Physical Libraries: These libraries are used for physically aware synthesis since these libraries contain the physical information of all the cells in the design. Macro/Hard IP library: The IP definitions and interface paths used in the RTL requires the respective library files to be available as a content of the link library.

3.0.5 STAGES IN SYNTHESIS

Import Design:

In this stage the RTL logic is read one by one and is converted to technology dependent netlist which in other words is the netlist mapped to the GTECH cells. This step is bifurcated into two separate processes and executes two important commands i.e. Analyse and Elaborate. The analyze command compiles the RTL files and simultaneously checks for errors and mismatches such as missing definitions, syntax or semantic errors. The elaborate command elaborates all the RTL Verilog files and converts them into a netlist. Once this stage is completed there are two important checks to be performed before we carry out the subsequent stages. They are the link command and the check design command. The link command gives information regarding any unmapped cells in the design and appear as black-boxes. The check design command reports the health checks such as errors related to the black-boxes in the design. Some of the errors reported in the design may be shorted outputs, multi-driven nets, unclocked registers etc. These reports are to be thoroughly checked and feedback must be provided to the front end team as and when required.

UPF:

The UPF file contains all the information related to the power supply domains. The UPF is also a methodology to separate out the functional information from the power information. It contains the power domains, power supplies, special cell policies and power state tables. In this stage the ¡design name upf¿ file is read into the Design Compiler. The check mv design command is executed post reading of UPF files. This command generates a report of the health of the design after UPF implementation. It dumps out a report containing all the issues related to missing isolation cells, power supplies, missing level shifters etc. All the errors we face in this report are cleaned working in close collaboration with the front end team before we can proceed further.

Uniquify:

In this stage, the tool copies and renames any multiply referenced design so that each instance references a unique design. Subsequently the original design is replaced with new and unique designs.

Constraints:

: In this stage, all the information related to the constraints in the design are specified. The constraints include timing, area and power constraints and are responsible for the quality of the synthesized design. It also specifies the operation condition constraints. All the design constraints are either provided in a Synopsys design constraint file or are fed as a group of files that can be sourced individually. The constraints related to timing include the definitions of clock, clock uncertainty, clock groups, clock latency, timing derates, timing exceptions etc. If the design comprises of LBIST / MBIST cells, the corresponding constraints must be provided for the control and correct functioning of the blocks.

Compile:

In this stage, the mapping of design to a technology dependent library is implemented. Once this stage is complete, all the cells which were till now mapped to the GTECH library are mapped to technology libraries, Also the design is optimized in this stage to meet the timing, power and area constraints that have been specified earlier. The check design and check mv design reports should be checked and must be clean.

Insert DFT:

The synthesis flow also introduces insert DFT stage to enable scan chain insertion as per the recommended rules.

Ungroup:

In this stage, the sub designs of a given level of hierarchy is merged into the parent cell or design. This stage removes the hierarchical boundaries, improves the timing, reduces the levels of logic (delay based auto ungrouping) and improves the area by sharing logic (area based auto ungrouping).

Re-timing:

The tool carries out optimization in this stage by moving the registers forward and backward across the combinational elements in a circuit. It is categorized as forward re-timing and backward re-timing. Forward re-timing removes a

fixed number of register from each input of a gate and inserts the same number of registers at the output. Similarly, backward re-timing removes a fixed number of registers from the output of a gate and inserts it at the front of each gate. This stage is implemented owing to the advantages it brings to the logic optimization. It minimizes the clock period of the circuit and minimizes the number of registers in the circuit. It also reduces the power consumption of the circuit.

Syn final:

The outputs of each of the stages in synthesis are dumped in the outputs directory folder which will be provided as the inputs for APR flow. The reports generated are dumped in the reports directory. Other stages include functions such as removing unnecessary registers that do not drive any load, eliminating registers in a design that never changes its state owing to the constant value on one or more input pins. Also some of the registers can be replicated to optimize timing QoR reports, congestion and fanout issues. Post synthesis, the design is taken to the floorplanning stage and APR flow is executed in Synopsys IC Compiler 2.

3.0.6 APR FLOW

The outputs of the synthesis stage are fed as inputs for the APR flow beginning with floorplanning. The IC Compiler 2 reads the netlist, UPF from synthesis, floorplan boundary, third party IPs or macro placement and IO port placement information from the respective files as and when the ICC2 is launched. Other inputs include DEF file, timing constraint files in SDC, LEF, libraries etc. The APR flow comprises of stages which are detailed in the upcoming sections.

Floorplanning:

Floorplanning is the most important step in backend VLSI design. A good floorplan can give a good chip performance whereas a bad floorplan can affect the timing and performance of a chip adversely. Power-planning is a method adopted to build the power supply network that includes proper placement of power and ground nets.
The hard blocks have fixed area and dimensions whereas the soft blocks have fixed areas but dynamic aspect ratios including their positions. Ideally, this stage also ensures that each of the modules are assigned a particular shape a physical location to facilitate gate placement. In this stage, also each pin having an external connection is assigned a location for the purpose of routing of internal and external nets. The design of floorplan is to optimize the locations as well as the aspect ratios of individual blocks to derive desirable floorplan attributes. The location of macro or hard IPs, location of ports and location and dimensions of voltage areas are finalized in the floorplan. The ports are

classified based on the respective input power supply. The UPF contains critical information about the power domains in the design. The voltage areas are created based on the positions of I/O ports, cells that are to be in the respective power domain and routing connections to the pins to minimize the use of always on buffers. The I/O ports are aligned as per the pin connections by analysing the fly lines so to minimize the routing distance. The macros in the design are placed in order to avoid congestion and routing shorts. For ports placed in a different power domain, secondary power grids can be created in the form of bounds to facilitate optimized routing. The bounds are created by sourcing the required bounding box coordinates in the PRE-floorplan and POST-floorplan hookup files. The check pin placement is a mandatory command that executes DRC checks for ports to check and analyze for any pin shorts, technology spacing or layer violation issues. After power planning and voltage area creation, the files containing the location information of I/O ports, macros and voltage areas are dumped out and rewritten. This completes the floorplan of the block.

Placement:

Once the design completes the floorplan stage, the standard cells or logic elements within each block is placed with the objective of maximum optimization by minimizing the length of connections between elements. The primary objective of this stage is to assign location and orientation to all the elements in a circuit within the given layout, following the constraints as well as fulfilling the optimization goals. This stage can further be classified as global placement and detailed placement. Global placement involves assignment of general locations to all the objects within a circuit whereas detailed placement refines the placement by moving object locations to legal cell sites enforcing non-overlapping constraints. This detailing of cell placement enables highly accurate values of delay in the circuit and optimizes timing. Gates, standard cells and macros are rectangular in design and are represented by nodes whereas nets are represented by edges. Also, some of the elements might have fixed locations whereas will constitute the movable group. The primary objective of global placement is overall uniform density distribution. Hence some overlaps between objects are allowed to prioritize density uniformity in the design, Cell legalization aligns all objects within the rows and columns removing overlaps by minimizing displacements from globally placed locations. Detailed placement provides an incremental change in the location of all cells by performing operation such as swapping objects or shifting objects in a row to create space for other objects. Typical runtime for global as well detailed placement are similar whereas the memory consumption for global placement is much more than required for detailed placement. The output of placement should be a well laid layout where in all nets in a design can be routed simultaneously. In other words, the performed placement must be efficiently routable. Electrical effects including delay and crosstalk should be considered during placement for a better optimized design and routing quality. Once the placement is complete, full chip routing is per-

formed in three stages i.e. global routing at a higher level, detailed routing at a lower level and timing driven routing.

Global Routing:

The purpose of global routing is to connect the pins with same electrical potential using physical wires. Post placement, the layout is represented as routing regions. . A sequence of horizontal or vertical channels used by the signal net where the adjacent tracks are connected by inter-layer vias is referred to as a route track. Similarly, the physical region constituting the routing tracks or channels are referred to as the complete routing region.

Chapter 4

TIMING CLOSURE METHODOLOGY

Timing closure is done by analysing the design with respect to the timing aspects of that design and it involves fixing all kind of timing violations. For analysing the design STA is used.
There are two stages in the design flow in which the methodologies for timing closure are applied. Post-ICC/Signoff stage and Pre-signoff/ICC stage. In the signoff stage, the design will be in ECO mode. The design will be frozen for further optimization in the floorplanning, placement or in the route stage. Hence, the timing fix will be done by writing ECOs. In the pre-signoff stage, the design will be in optimization mode. The timing fixes can be done by changing the floorplan, placement and route of the design. Usually, most of the timing violations will be fixed in ICC stage of the design flow.

4.1 POST-ICC/SIGNOFF STAGE

In this stage, the timing violations are fixed by sizing cells, replacing cells or inserting buffers and the fixes are done once the design has been out of the APR flow. All the commands to make these changes are given in the ECO and is performed on a small portion of the chip to prevent disturbing the placement and routing of the rest of the chip. Loosely, ECO is a TCL file that contains the necessary commands for fixing the timing violations. Each and every violation has its own methodology to fix it. The most prominent violations are setup hold violation, transition violation, capacitance violation, crosstalk noise and crosstalk glitch. In these, the transition and capacitance violation comes under DRC violations.

4.1.1 SETUP HOLD VIOLATION FIX

The fixing can be done in two areas Data path and Clock path. Data path optimization is the technique used on data path in which the standard cells

will be upsized (increase the drive strength) or downsized (decrease the drive strength). Clock pulling and clock pushing is the technique used on clock path in which clock buffers will be inserted or removed from the clock path. Clock path should be touched only if there is no scope for data path optimization. Clock is a high fan-out net, so a change in clock path might affect the entire design. Methods to fix setup violation are given below.

Methods to fix Setup violation are given as

- If the drive strength is low for increase and decrease drive strength , change the drive strength with Upsizing the standard cells (increase the drive strength) in data path.
- Pull the launch clock uses if there is no chance in data path.
- Push the capture clock uses if there is no chance in data path.
- Removing buffers from data path (hold margin should be checked).
- VT swap Replacing high VT cells with low VT cells.
- Replacing buffers with two inverters placing farther apart so that delay can adjust.

buffers with two inverters placing farther apart so that de

- If the drive strength is high for increase and decrease drive strength , change the drive strength with Downsizing the cells (decrease the drive strength) in data path.
- Pulling the capture clock uses if there is no chance in data path..
- Pushed the launch clock uses if there is no chance in data path..
- Inserting buffers/Inverter pairs/delay cells to the data path.
- By increasing the wire load model, we can also fix the hold violation.

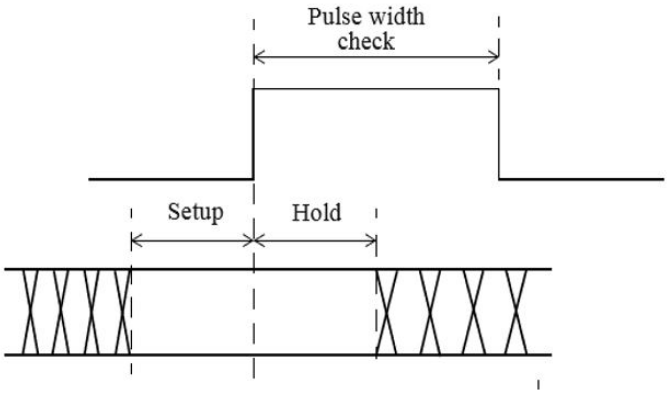

Figure 4.1.1: Setup time and Hold time

4.1.2 TRANSITION AND CAPACITANCE VIOLATION FIX

In Primetime echo G MAX TRANSITION command will display the maximum transition limit of the design and report timing -trans will show the transition values of the path. A transition violation in a timing path makes the data to arrive late by inducing more delay and thereby causing a setup violation. It can be fixed by following steps.

- Upsizing the driver cell thereby reducing the delay induced by the cell.
- Decreasing the net length by moving cells nearer or reducing long routed net.
- Adding Buffers in the net which will break a long net in to smaller nets.
- Increasing the width of the route at the violation instance pin. This will decrease the resistance of the route and fix the transition violation.

4.1.3 CROSSTALK GLITCH AND CROSSTALK DELAY FIX

Crosstalk noise is caused by the building up of capacitance between two neighbour nets. By preventing this capacitance built up and by preventing the formation of aggressor net that is by nullifying the effect of switching, this problem can be avoided. The timing reports of PrimeTime will already contain the crosstalk delay information. For the glitch information separate noise report is generated. There are several methods to fix the timing violation caused by the crosstalk noise. But first the way how STA will do setup check and hold heck with crosstalk enabled has to be understood. In addition to fixes provided for crosstalk noise, there are methods to prevent them before occurring.

4.1.4 SETUP AND HOLD ANALYSIS WITH CROSSTALK

4.1.5 PREVENTION OF CROSSTALK NOISE

Crosstalk noise is very critical in nanometer designs. Instead of fixing them after their occurrence, if there is a way to prevent it from occurring then that should be capitalized properly. In the physical design phase itself there are techniques that can be used to prevent the crosstalk noise. This will help to improve the timing.

Shielding:

Spacing of wires:

Guard ring:

4.1.6 PRE-SIGNOFF/ICC STAGE

In this stage, most of the timing violations are fixed by physically optimizing the design in floorplan, placement or route stage. For all the designs, timing closure cannot be achieved in this stage. But the violations can be fixed as much as possible and after that the design will be frozen for any physical changes or optimization. All the options for timing fix in this stage are performed during the APR flow. In this stage ECOs are not used for the fixing of violations. Not any incremental change is done during this stage.

4.1.7 EFFICIENT FLOORPLAN

Floorplan is one of the reproving and important step in physical design. Timing violations can be prevented by a good floorplan. It's better to understand the basic design of Floorplan.

4.1.8 MACRO PLACEMENT

A design consists of standard cells, macros, IO ports, boundary cells, voltage areas and upf components. Floorplan decides where all these components should be located in the design. The most efficient practice of the placements of these components is placing all the standard cells in the core area all the macros near the boundary of the design. The pins of the macros should be facing the core area and the boundary so that the routing can be done effectively. In this way a good timing can achieved in the floorplan stage of the design itself. The orientation of the macro can also be changed if there is any timing violation is seen. There should also be a minimum spacing between two macros so that the power rails can be routed properly. show below how effectively macros shall be placed in a design.

4.1.9 IO PLACEMENT

The IO ports are considered to be pins for the blocks through which it communicates with the external world or other partitions. All the IO ports are placed on the boundary of the design only. IO port optimization is a technique through which port-pin alignment is achieved. This paves way for better timing. In IO port optimization the ports of the design and the pins inside the design are aligned in such a way that there is only a minimum amount of distance between them. Hence, routing can be made efficiently utilizing less amount of metal. The factor that ports will be talking to the neighbouring partitions should also

be considered during IO port optimization. Also, if the port size is same as that pin size then chances of getting DRC violations during routing is very less.

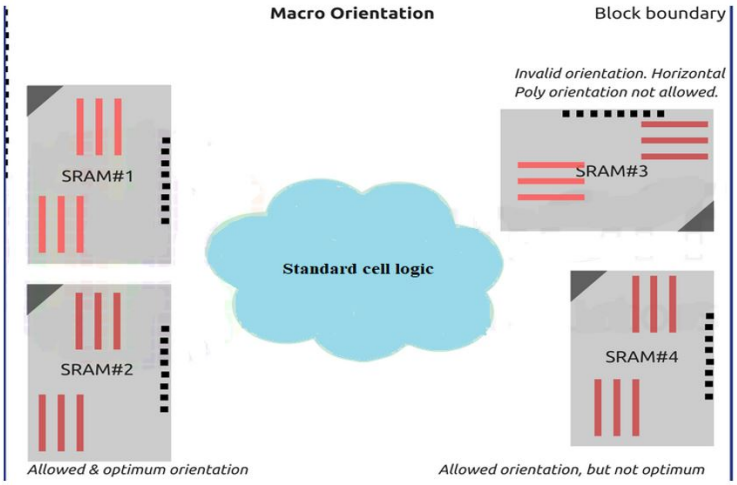

Figure 4.1.2: Efficient macro placement and orientation

4.1.10 STANDARD CELL PLACEMENT

In modern design, the number of standard cell count in a design has increased drastically. Placing them one by one is a tedious task. Hence, some constraints are given for placing them. A core area is allocated for standard cells placement and all the cells are confined within that area. Also there is one technique called bound creation. Bound is an imaginary area in the design in which a group of cells are confined within. In this way the distance between two cells will be reduced and routing can be done with minimal metal. This ultimately improves the timing. If the placement of a cell is bad then it can be moved manually to a better place.

4.1.11 CTS

There are two important steps in the CTS. They are clock tree building and clock tree balancing. Once the CTS is done, then the skew and latency reports should be analysed. If they are found to be worst then the clock tree has to be rebuilt to make them better.

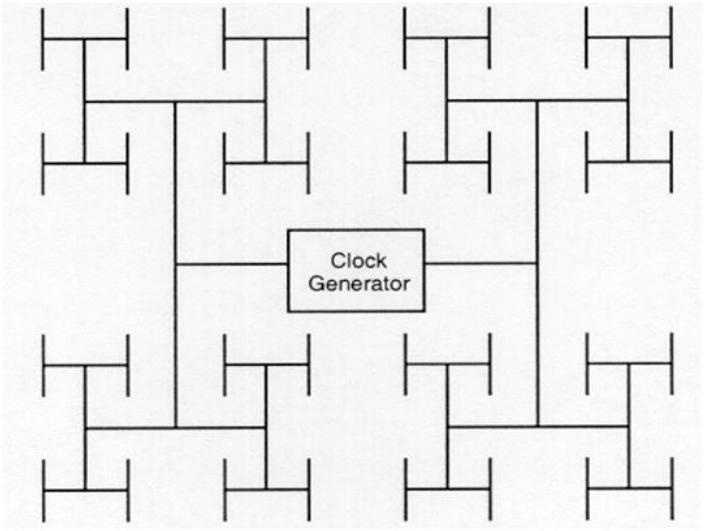

Figure 4.1.3: Typical H-tree

Clock tree optimizations can be done by buffer sizing, gate sizing, High Fan out Net (HFN) synthesis and buffer relocation.

Chapter 5

TIMING CLOSURE IMPLEMENTATION

5.0.1 POST-ICC/SIGNOFF STAGE IMPLEMENTATION

The implementation was carried out in two stages, post ICC and ICC. In post ICC the design will be in ECO mode that is the violations are fixed in an incremental way. Synopsys PrimeTime tool is used in this stage to analyse the timing paths and source the ECOs.

5.0.2 REG TO REG TIMING PATH ANALYSIS

To analyze the reg to reg timing path, first the path needs to be reported in the PrimeTime session. For this the PT session has to be restored first. The command restore session is used. The command report timing is used for viewing the path. The exact syntax is report timing from start point to end point. This will in default show the max path. For min path delay type min should be added at the end of the command. Every reg to reg timing path will have a start point, end point and combinational cloud between them.

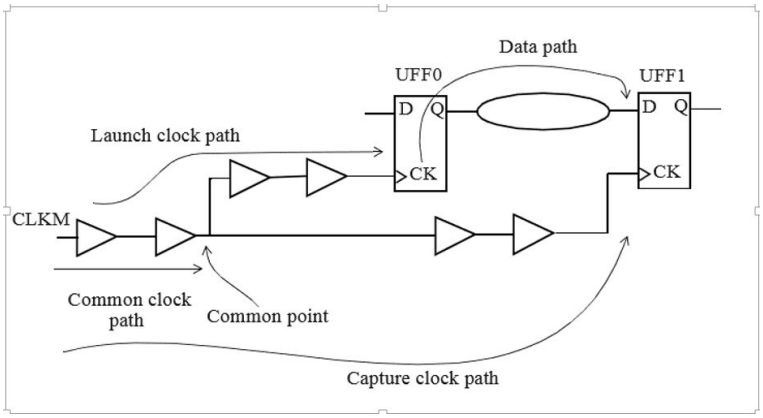

Figure 5.0.1: A typical Reg to Reg timing path

5.0.3 HOLD VIOLATION FIX

Hold violation occurs when the data arrives early at the end point. This is prominent in the min corner where the delay will be less. The arrival time of the data has to be delayed to fix this and by inserting buffers at the end points this can be achieved. But this can be attempted only if the path has enough setup margin in the max corner. Paths that dont have enough setup margin are called min-max path and they are removed from the list, as this process is only for paths with setup margin. Also the block should have enough area to add new buffers. A TCL script can be written to insert buffers at all the endpoints. The inputs given to the script are unique endpoints list with slack information, new buffer name and the delay value of it. These information are required to fix the hold violation. The concatenation command is used on the PrimeTime reports to generate the endpoints list. Syntax is cat path to reports/block name/ — egrep pattern — grep pattern. There may be repetition of endpoints in the list obtained using cat command. To get unique endpoints perl script can be used which will sort the list and eliminate the same endpoints. Using another tcl script the endpoints list is assigned to a variable. The endpoints must be reported in max corner using report timing through endpoint to check whether enough setup margin is there. Buffer information should also be given to the script. Finally the tcl script should be sourced in the PT session and redirect the output to a new file. The command source script.tcl ¿ eco.tcl is ran. An ECO to fix the hold violations is the first output of running the script. The exact syntax would be insert buffer endpoint name buffer name. After sourcing the ECO, an updated design with some of the hold violations fixed is obtained. Fixing hold violation has a negative effect on setup margin and vice versa.

Blocks	Before ECO		After ECO	
	No. of Violating Paths	Slack Value of the Worst Path (ns)	No. of Violating Paths	Slack Value of the Worst Path (ns)
Block 1	1754	-765.12	258	245.32
Block 2	5678	-312.98	442	156.72
Block 3	598	-445.29	34	79.54
Block 4	2791	-555.17	112	98.10
Block 5	3763	-124.67	169	179.47

Figure 5.0.2: Hold violations status before and after ECO

5.0.4 BUFFER ANALYSIS

As a part of fixing hold violations, the analysis of a buffer performance is to be done. The delay induced by the buffer that is to be inserted in the endpoint must be known since buffer insertion technique is used for fixing hold viola-

tions. Only three buffer variants are allowed to be used for fixing hold violation. Buffer with low drive strength (buf07), Buffer with medium drive strength (buf13) and Buffer with high drive strength (buf27). The delay induced by these buffers can be found by inserting these buffers in the design using insert buffer command. After adding these buffers the updated timing path can be reported and the delay value can be seen. This information is used while fixing hold violation.

Buffers	Delay Induced (ns)	
	In max corner	In min corner
Buf 01	85.63	40.10
Buf 06	100.11	50.32
Buf 09	118.92	59.62

Figure 5.0.3: Delay information of the buffers

The above table shows the delay information of the buffers that are to be used in the hold violation. It can be seen that the buffers induce more delay in the max corner than in the min corner. So, there should be enough setup margin in the max corner. Usually 1.5 times of the min slack is required as the setup margin.

5.0.5 SETUP VIOLATION FIX

Setup violation occurs when the data is arriving late at the end point. This is checked at the next edge of the capture clock with respect to the launch clock. This is prominent in the max corner because the delay will be more. To fix this the data should be made to arrive early by reducing the delay incurred by the data path (for data path optimization). The violating path is analysed first to find the scope of fixing it in data path. Cells causing large violations are upsized. High VT cells can be replaced with low VT cells but this will increase the leakage power. ECO is written with size cell commands for the fix and is sourced in the PrimeTime session. The exact syntax would be size cell cell to be replaced new cell. An updated design with fixed setup violations is obtained after sourcing the ECO.

Blocks	Before ECO		After ECO	
	No. of Violating Paths	Slack Value of the Worst Path (ns)	No. of Violating Paths	Slack Value of the Worst Path (ns)
Block 1	56	-322.12	12	56.78
Block 2	112	-289.43	34	78.51
Block 3	345	-178.39	78	44.58
Block 4	643	-102.66	110	32.76
Block 5	436	-71.12	159	86.17

Figure 5.0.4: Setup violation fix status before and after ECO

5.0.6 TRANSITION AND CAPACITANCE VIOLATION FIX

Fixing transition and capacitance violation is similar to fixing setup and hold violations because these will ultimately be reflected in the setup and hold checks as one of the violations. If the transition and capacitance value of a net or cell exceeds the maximum limit specified in the design, then it will be marked as a violation. These violations come under DRCs and are fixed first. A bad transition in data path causes more delay thereby making the data to arrive very late. This is bad for setup and may cause setup violation. But in case of hold check this bad transition will be friendly as it is delaying the data, which is good for hold check. It will be masking the hold violation. If this violation is not addressed at the initial stage then while fixing it at the final phase, it will cause hold violations. Transition violation can be seen at data transition as well as clock transition. The driver cell of the bad transition net is noted and is upsized by using the size cell command. Buffers are inserted in long nets, which will break them in to small nets using insert buffer command. But for inserting buffer, the availability of enough setup margin has to be made sure. The same fixes apply for capacitance violation also. Unlike transition limit, the limit for capacitance in the design will be for both maximum and minimum. But here the fixes have been given only for violations crossing the maximum limit. There were only negligible amount of minimum limit capacitance violation.

Blocks	Maximum Transition Limit (ns)	Worst Transition Value (ns)	Violation Amount (ns)	Transition Value after ECO (ns)
Block 1	300	367.91	67.91	227.83
Block 2	300	450.16	150.16	179.00
Block 3	300	417.28	117.28	284.59
Block 4	200	359.46	159.46	112.62
Block 5	200	299.77	99.77	92.39

Figure 5.0.5: Transition violation fix status before and after ECO

Blocks	Maximum Capacitance Limit (pf)	Worst Capacitance Value (pf)	Violation Amount (pf)	Capacitance Value after ECO (pf)
Block 1	100	247.62	147.62	94.60
Block 2	100	191.11	91.11	71.18
Block 3	100	163.49	63.49	24.39
Block 4	50	150.86	100.86	33.38
Block 5	50	77.30	27.30	17.62

Figure 5.0.6: Capacitance violation fix status before and after ECO

5.0.7 CROSSTALK NOISE FIX

The fix for crosstalk delay and glitch is done in both the pre-signoff and signoff stage. The help of both the stages are needed. The STA tool using the setup and hold analysis methods for crosstalk will provide the delay information. The total crosstalk delay (sum of all the individual paths crosstalk delay) in that design is calculated. The glitch details are also provided by the STA tool by using the timing windows of the aggressor net. The highest glitch magnitude and the aggressors responsible for it in a group of aggressors are obtained. This analysis is done in the signoff stage.

Blocks	Total Crosstalk Delay before Shielding (ns)	Total Crosstalk Delay after Shielding (ns)
Block 1	386.13	137.08
Block 2	271.19	91.23
Block 3	453.22	204.09
Block 4	398.47	177.72
Block 5	186.62	65.29

Figure 5.0.7: Crosstalk delay fix status

5.0.8 CONTEXT BASED SESSION CREATION FOR A BLOCK

Context based timing session is a normal PrimeTime session which will have the context information. The latency values in this session will come from the full chip timing team and the value will be of full chip level. This is basically done to achieve better Turnaround Time (TAT) as the time required to update the design in block level is very much less compared with full chip level. Context is based on a synopsys technology called HYPERSCALE. The context information is provided by the full chip level team. If the violation is seen in block level and full chip level, then a fix in block level will itself fix the violation in full chip level. By this the ECO can be verified quickly and ultimately the design will be closed quickly achieving the TAT. Creating a session requires certain input. Since it is context based session, the context information is the first and foremost input required. The HYPERSCALE variable is enabled for this process. In the working area, netlist, spef, upf and other timing information are saved. Session creation command is run finally. The output of this process is the context based session for the particular block which can be found in the working area. If correlation is done between this session and full chip session then there will be no discrepancy. Now this session can be given as input to the PT-ECO.

Blocks	Maximum Height of the Glitch (uV)	Before Shielding		After Shielding	
		No. of Glitches	Height of the Worst Glitch (uV)	No. of Glitches	Height of the Worst Glitch (uV)
Block 1	0.5	413	1.78	71	0.46
Block 2	0.5	412	1.09	87	0.21
Block 3	0.5	361	2.62	36	0.39
Block 4	0.5	276	2.43	108	0.40
Block 5	0.5	225	0.94	173	0.19

Figure 5.0.8: Crosstalk glitch fix status

5.0.9 CORRELATION OF BLOCK AND FULL CHIP LEVEL SESSIONS

The slack value of a path may differ in block level compared to full chip level. This discrepancy is observed by correlation. Both block level session and full chip level session are restored. A violating path is taken and reported in both the sessions and then observed. Three scenarios were observed.

- The path is violating in both the levels with a small difference.
- The path is violating in full chip level but meeting in block level with huge margin.

- The path is violating in full chip level but in block level the same path is unconstrained.

These scenarios are occurring because of the discrepancy between block level and full chip level. In block level the timing information may not be complete. For example, clock to the block may be originating from a different block (top module) so there will be latency. This latency information will be provided in full chip level as it is seen from the top. But in block level this information is not present as it is not seen from the top. Only the latency information from the port of the block to clock pin of flop will be provided.

Paths	Block level	Full chip level
Path1	WNS of -344.58ns	WNS of -356.98ns
Path2	WNS is +681.23ns	WNS is -267.51ns
Path3	Path is Unconstrained	WNS is -491.63ns

Figure 5.0.9: Correlation Observations

5.0.10 PRE-SIGNOFF STAGE IMPLEMENTATION

In this stage the timing fixes are done by making changes in the floorplan and placement of the design because the design is implemented physically. Also, the timing is checked after each and every stage and is optimized there itself. Synopsys ICC 2 compiler is used in this stage for the physical implementation of the design.

5.0.11 TIMING ANALYSIS AFTER EVERY STAGE

Since the design is being physically implemented in this stage, it is better to check timing after each and every stage. If most of the violations are fixed in this initial stage itself then in the post-ICC stage it will be easy to meet the timing. Mostly the timing is checked after synthesis, placement, CTS and routing. First a full run is done right from synthesis to route. If the timing is bad in synthesis then there is no use of going in to floorplan, placement and routing. The QoR reports after those four stages is given in table.

Synthesis		Placement		CTS		Route	
R2R WNS (ns)	FEP	R2R WNS (ns)	FEP	R2R WNS (ns)	FEP	R2R WNS (ns)	FEP
-710.45	271	-795.12	568	-863.41	991	-985.08	1021

Figure 5.0.10: Timing QoR after every stage

5.0.12 TIMING OPTIMIZATION IN SYNTHESIS

Timing with cost function 1		Timing with cost function 2.5	
R2R WNS (ns)	FEP	R2R WNS (ns)	FEP
-710.45	271	-269.72	145

Figure 5.0.11: Timing Before and After Increasing Cost Function

5.0.13 TIMING OPTIMIZATION IN FLOORPLAN

The timing after placement is also looking bad from table aboveSo, optimization must be done after placement stage also. In floorplan, the most important steps are macro placement, voltage area creation and IO port placement. By efficiently doing all these steps the timing can be improved in floorplan.

5.0.14 MACRO PLACEMENT

The placement of macro also affects the timing very effectively. Shows a poor macro placement. It can be see there are two macros in the design. Macro1 is sitting very far from the ports it is talking to whereas, macro2 is sitting close to the ports it is talking to. So, the timing of the paths from macro2 to the ports will be good. The routing will happen to macro1 as shown in the figure. In order to optimize those paths, the macro1 should be moved close to the ports it is talking. This is done by using move-objects command or by manually moving the macro in the layout window. After moving the macro, the new macro placement information is dumped out and is used for future runs. The optimized macro placement is shown in the figure below-

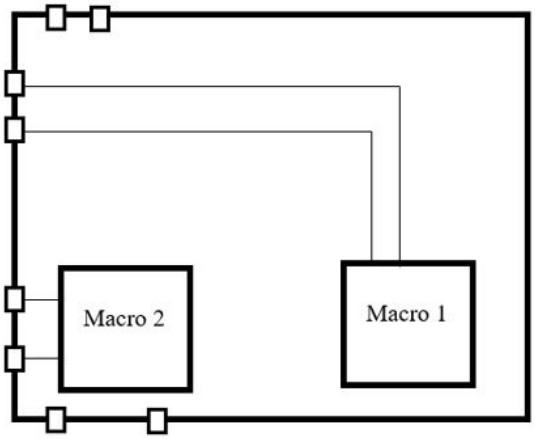

Figure 5.0.12: Macro Placement before Optimization

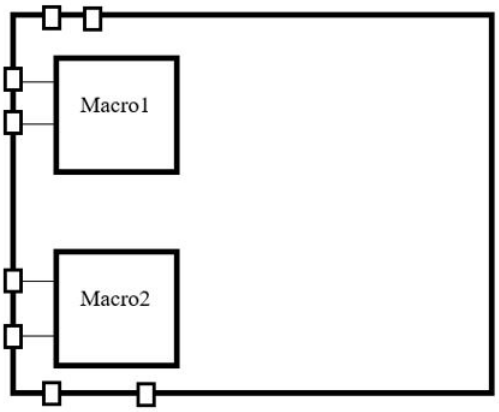

Figure 5.0.13: Macro Placement after Optimization

5.0.15 IO PORT OPTIMIZATION

As a part of the layout information, in addition to boundary of the design the placement information of the IO ports will also be given. They also should be placed on the boundary of the design. These ports act as the gateway between external world and the components inside the chip. Aligning the ports along with the pins of macros or cells will reduce the routing the distance, improve the timing and helps the neighbouring partitions. In figure 5.4, it can be seen that the IO ports are randomly placed with respect to the pins of macro. This will affect the timing of the design very much. For the optimization, firstly the pins that are connected to the ports must be identified. This is done by using all-connected command. This is done in the floorplan stage so that the pin information is obtained easily. get-ports command is used to get all the ports. Then, the location of the pins is identified by getting the bounding-box information of the pins using bbox attribute of the pins in get-attribute command. After this, the ports are moved to the exact location of the pins. From there the ports are moved to the boundary by giving the specific delta value. The delta value is the difference between designs X or Y and macros X or Y.

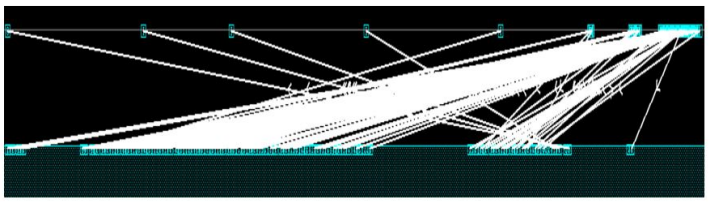

Figure 5.0.14: IO Port Placement before Optimization

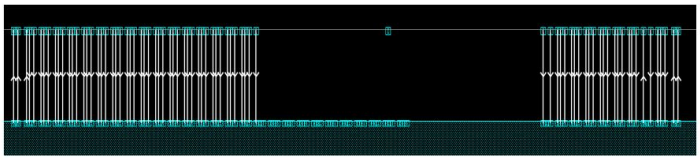

Figure 5.0.15: IO Port Placement after Optimization

5.0.16 TIMING OPTIMIZATION IN PLACEMENT

Irregular placement of the cells and sequential elements can also affect the timing of the design. Floorplanning gives the location information of the components in the design. During placement process only the components are placed physically in the location allotted for them.

5.0.17 BOUND CREATION

Above figure shows the design in which some of the cells are scattered at the bottom of the block. The dotted line represents a bus, which is a group of connections that travel from one point to another point. Since this bus is travelling for long distance, it is considered to have bad timing QoR. This can be fixed by placing the cells closer together. This is achieved by creation of bounds. create-bound command is used for creating the bounds at a particular region. After that, the cells are confined within that bound area. The small rectangle specified with dotted lines in figure. is the bound that has been created. All the cells that were at the bottom are now confined in that area thereby improving the timing QoR by reducing the distance travelled by the bus.

Timing QoR before Optimization		Timing QoR after Optimization	
R2R WNS (ns)	FEP	R2R WNS (ns)	FEP
-775.83	517	-287.41	290

Figure 5.0.16: Timing after Placement Optimization

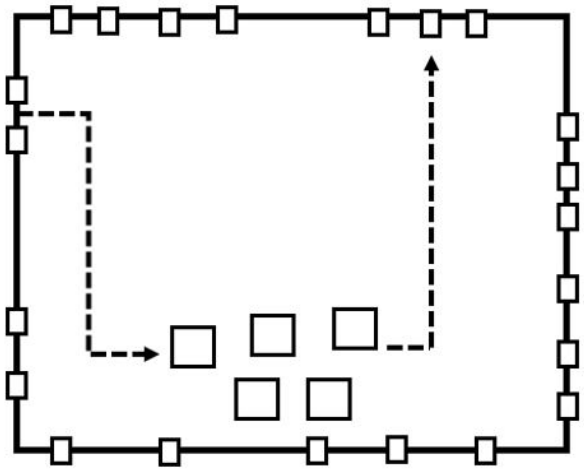

Figure 5.0.17: Cells Placement before Bound Creation

5.0.18 TIMING OPTIMIZATION IN CTS

There are several ways to build an efficient clock tree in the design. Global drivers are the cells that act as clock source. They are placed all over the design as per the requirement. Clock is obtained from them and are evenly distributed to all the sequential elements. The main aim of CTS is to minimize the latency and skew. In figure , the blue coloured block is the global driver and the two

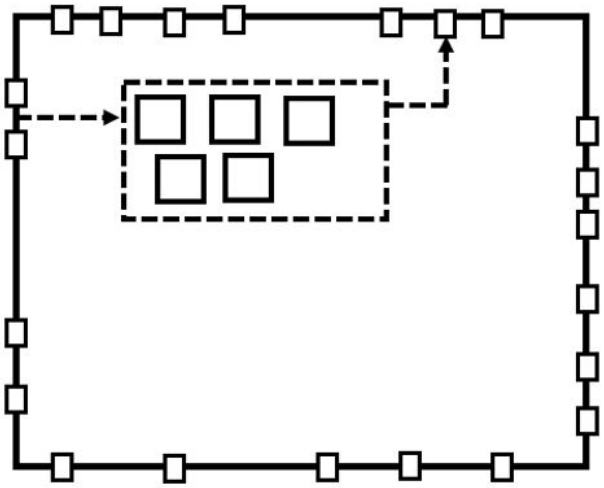

Figure 5.0.18: Cells Placement after Bound Creation

small rectangles are sequential elements. It can be seen that the flip flop at the top will receive the clock signal faster than the flip flop at the bottom. This creates skew issue in the design. To fix it, the tool will add buffers in the shortest path to match the skew value between two flip flops. This is done automatically by the tool. But the path has to be mentioned to the tool.

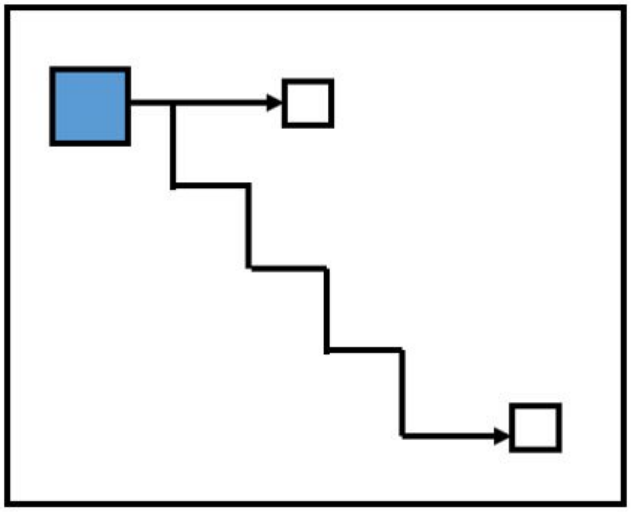

Figure 5.0.19: CTS with Bad Skew

Timing QoR before Optimization		Timing QoR after Optimization	
R2R WNS (ns)	**FEP**	**R2R WNS (ns)**	**FEP**
-775.83	517	-287.41	290

Figure 5.0.20: Timing QoR after CTS Optimization

Chapter 6

CONCLUSION

Various methods for achieving the timing closure of the design were discussed. Timing optimization in both the pre-signoff and signoff stage is important. Usually the design will be in the post ICC/signoff stage for long time because this stage is responsible for the signoff and closure. Once the signoff team signals, then only the design will be moved in for tape out (fabrication). That is the reason why timing team is more important in the entire design flow. Timing closure basically involves analyzing loads of data and deriving a solution form that analysis. From the analysis, it can be seen that hold violation is more critical than the setup violation as it doesnt depend on the timing period. The hold violation occurs only after the clock tree is built. Before the clock tree is built all the clocks will be ideal, that is the latency and skew will be zero. Hence before CTS the setup violations are fixed. The clock pulling and pushing techniques were not implemented in this project, which are to be explored in the future. Also various techniques for optimizing the timing in the pre-signoff stage were discussed. This pre-signoff stage optimization plays a vital role in timing closure because most of the timing issues are solved in that stage itself. There is no pre-defined set of techniques in pre-signoff stage as in signoff stage to fix the timing issues. The techniques that are used are partition specific. There can be designs where IO port optimization is not possible at all. Only the shielding technique of crosstalk fixing was discussed. There are other techniques for fixing the crosstalk noise, which can be studied and explored in the future. The other design constraints, power and area should be considered and checked throughout the optimization process. Generally those factors will also be analyzed and optimized in parallel. With increasing technology more violations will occur in the future designs and complexity to fix the timing issue will also increase. But STA will hold good for analyzing the design and fixing the violations.

References

[1] by J. Bhasker and Rakesh Chadha. Static Timing Analysis for Nanometer Designs A Practical Approach Springer

[2] Synopsys Timing Constraints and Optimization User Guide.

[3] Synopsys PrimeTime User Guide.

[4] Synopsys Primetime Suite Manual for Tool Commands and Error Messages.

[5] Synopsys IC Compiler II Timing Analysis User Guide.

[6] Solvnet by Synopsys.

[7] Intel documents.