Physical Design And Clock Tree Synthesis Methods For A 8-Bit Processor

Debaprasad Daxiniray

Department of Electronics and Communication Engineering

National Institute of Technology Rourkel

Physical Design And Clock Tree Synthesis Methods For A 8-Bit Processor

A thesis submitted in partial fulfillment

of the requirements of the degree of

Master of Technology

in

Electronics and Communication Engineering

(Specialization: VLSI Design and Embedded Systems)

by

Debaprasad Daxiniray

Roll No: 214EC2166

Under supervision of

Prof. Debiprasad Priyabrata Acharya

May 2016

Department of Electronics and Communication Engineering

National Institute of Technology Rourkela

Department of Electronics and Communication Engineering

National Institute of Technology Rourkela

Debiprasad Priyabrata Acharya

Associate Professor

May 31, 2016

Supervisors' Certificate

This is to certify that the work presented in the thesis entitled "*Physical Design And Clock Tree Synthesis Methods For A 8-Bit Processor" s*ubmitted by *Debaprasad Daxiniray,* Roll Number 214EC2166, is a record of original research carried out by him under my supervision and guidance in partial fulfillment of the requirements of the degree of *Master of Technology* in *VLSI Design and Embedded Systems.* Neither this thesis nor any part of it has been submitted earlier for any degree or diploma to any institute or university in India or abroad.

Prof. Debiprasad Priyabrata Acharya

Supervisor

Dedication

This work is dedicated to my family…..

Signature

Declaration of Originality

I, Debaprasad Daxiniray, Roll Number 214EC2165 hereby declare that this dissertation entitled Physical Design and a Clock Tree Synthesis Methods For a 8-Bit processor presents my original work carried out as a postgraduate student of NIT Rourkela and, to the best of my knowledge, contains no material previously published or written by another person, nor any material presented by me for the award of any degree or diploma of NIT Rourkela or any other institution. Any contribution made to this research by others, with whom I have worked at NIT Rourkela or elsewhere, is explicitly acknowledged in the dissertation. Works of other authors cited in this dissertation have been duly acknowledged under the sections “Reference” or “Bibliography”. I have also submitted my original research records to the scrutiny committee for evaluation of my dissertation.

I am fully aware that in case of any non-compliance detected in future, the Senate of NIT Rourkela may withdraw the degree awarded to me on the basis of the present dissertation.

31st May, 2016
NIT Rourkela

Debaprasad Daxiniray

Acknowledgment

First and foremost, I would like to express my deep and sincere gratitude to my supervisor Prof. Debiprasad Priyabrata Acharya for providing excellent guidance, platform and encouragement throughout the journey of this work. Without his guidance, this work would never have been a successful one. I am also deeply grateful to Mr. Debasis Sahoo and Mr. pradeep Patra Consultants at Sankalp Semicoductor Pvt. Ltd. for their valuable guidance during inception and period of the project. Finally, I am deeply thankful to my supervisor, Prof D P Acharya for provided me the opportunity to work as an intern in Sankalp Semiconductor Pvt. Ltd. with lots of good advice and support with clear guidance throughout journey. My special thanks to Back-End Physical Design team for the important technical discussions especially Mr. Rama Rao Kotapally, Arvind Kumar Mishra and Anshul Agrawal who helped me a lot whenever I get stuck while working at Sankalp Semiconductor. I am thankful to Sankalp Semiconductor Pvt. Ltd. for supporting this work through an internship. My regards to all my friends here at NIT Rourkela, who made this journey a wonderful one for me. Last but not the least; I would like to thank my parents & family who have always supported me emotionally.

31st May, 2016 *Debaprasad Daxiniray*
NIT Rourkela Roll Number: 214Ec2166

ABSTARCT

Now days a number of processors are available with a lot kind of feature from different industries. A processor with similar kind of architecture of the current processors only missing the memory stuffs like the RAM and ROM has been designed here with the help of Verilog style of coding. This processor contains architecturally the program counter, instruction register, ALU, ALU latch, General Purpose Registers, control state module, flag registers and the core module containing all the modules. And a test module is designed for testing the processor.

After the design of the processor with successful functionality, the processor is synthesized with 180nm technology. The synthesis is performed with the data path optimization like the selection of proper adders and multipliers for timing optimization in the data path while the ALU operations are performed. During synthesis how to take care of the worst negative slack (WNS), how to include the clock gating cells, how to define the cost and path groups etc. have been covered.

After the proper synthesis we get the proper netlist and the synthesized constraint file for carrying out the physical design. In physical design the steps like floor-planning, partitioning, placement, legalization of the placement, clock tree synthesis, and routing etc. have been performed. At all the stages the static timing analysis is performed for the timing meet of the design for better performance in terms of timing or frequency. Each steps of physical design are discussed with special effort towards the concepts behind the step. Out of all the steps of physical design the clock tree synthesis is performed with some improvement in the performance of the clock tree by creating a symmetrical clock tree and maintaining more common clock paths. A special algorithm has been framed for creating a symmetrical clock tree and thereby making the power consumption of the clock tree low. Due to the technology scaling the PVT variations increases due the scaling of interconnects. Due to present demand of multi-mode performance and different corner analysis, the iterations for the all this analysis takes a lot of machine cycle and run time. So some technique called as Distributed Multi Mode Multi Corner (DMMMC) has been discussed for reducing the machine cycle and the run time.

Keywords: *8-bit µP, Synthesis, Physical Design, CTS, K-Mean Clustering*

Contents

List of Figures

List of tables

Chapter 1

INTRODUCTION

1.1 Background

The VLSI industry is going deeper and deeper day by day obeying the Moore's law which says that "in every 18 months. The number of transistors doubles itself on a chip" and this trend is till now continuing as it was framed in the year of 1965. That time the law was getting used in some semiconductor industries for a long-term planning and hence targets for research and developments. The technology window started with some μm and now it has reached to the nm technology that to TSMC is going to launch 5nm technology while INTEL started using the 7nm technology. Hence due to this technology scaling different kinds of problems and design challenges are coming up day by day.

Due to the increased rate of scaling and the growing demand of increased functionality over System-on-Chip, the complexities in designing and doing the timing analysis is becoming very tough day by day. The rate at which the technology scaling is happening, the interconnects are not getting scaled in the same rate but still the width of the interconnects are reducing in every scaling thereby increasing the resistance of the interconnects. Due to this increase in the resistances the analysis for the parasitic corners is increasing. So due the increased number of the modes of functionality and the parasitic corners the analysis views are increasing thereby increasing the machine run iterations. Due to this increase in the interconnect RC values and the effects due this has been discussed in the chapters followed.

1.2 Problems

As the total power consumption in the chip are shared by the clock network power consumption mostly. This clock network power takes around 70-75% of the total power consumption of the chip. So to reduce the total power consumption of the chip, the clock network power consumption should be reduced. As the clock network involves mostly a huge number of buffers and invertors hence there should be some techniques to reduce this huge buffered network and for this a technique has been framed in the following chapters.

For completing the total flow of the physical design starting from the RTL to the GDSII generation there are so many challenges for completing every individual steps with completeness and perfectness. The different aspects of the timing analysis and the concepts were required to be understood for carrying out the timing analysis with the corner analysis.

Due to the decrement in the parasitic dimensions due to the scaling as discussed the run time for timing analysis while in the total span of the physical design becomes huge. So there is a way to reduce the number of iterations and also skipping some of the corner which are irrelevant for analysis. These techniques and the ways have been discussed in the following chapters.

1.3 Motivation and Objectives

The primary motivation was to learn different aspects of Physical Design and concepts behind every step which are running in the industry standard. The physical design steps were covered with total flow from the RTL deign in the Frontend to the GDSII generation in the Backend [15]. After designing the processor with the total steps of physical design it was observed that the dynamic power consumption in the clock tree is very significant in the total power consumption.

As the technology node are going down with the Moore' law thereby increasing the OCV (On-Chip Variations) variations, the clock tree symmetric structure with maximum common path sharing [16] was thought for tackling this OCV to some extent.

While the main Objective here was to reduce the power consumption of the chip by employing different techniques such as the clock gating, power gating and the use of the Multi Bit Flip-Flop structure [1]-[6] and [7]-[14] . And with the use of the proposed algorithm with the clustering concept [1]-[6] the power consumption was reduced and the algorithm was able to achieve a symmetric clock tree which is shown later in the following chapters.

1.4 Proposed Idea

The idea of implementing an algorithm for clustering of the flip-flops into some k number of clusters in a way that no clusters will exceed the load limit that is set as the constraint for each clusters. By doing this we can know that at the leaf level the loads are equal for all the clusters. After this clustering a buffer allocation technique has been proposed, this will allocate the same buffer to all clusters hence making the clock tree symmetric at the leaf level with same kind of buffers. As the clock tree control is now on the hand of the designer with the manual clock tree, the clock tree can be built with less no of buffers. So that the switching will be less in the buffers of the clock tree thereby making the power consumption of the clock tree less as before.

1.5 Thesis Organization

Chapter 2 describes about the architecture of the processor, simulation and synthesis of the processor. Chapter 3 describes about the flow of physical design and the concepts behind each steps taken in this flow. Chapter 4 describes about the Static Timing Analysis and the related concepts and the different measures taken for the improvement. Chapter 5 talks of the

algorithm adopted for making the clock tree symmetric and for reducing the power consumption of the clock network and hence of the total power consumption. Chapter 6 concludes the total discussion of the thesis and talks the future scopes for the project.

Chapter 2

ARCHITECTURE DESCRIPTION AND SYNTHESIS OF A 8-BIT PROCESSOR

2.1 Introduction

Microprocessor, which is an electronic device consisting of arithmetic and logic unit (ALU) and control circuitry for functioning of computer's CPU. Microprocessor is an integrated circuit that interprets and executes the program instructions and behaves intelligently. The processor operates at a speed of the internal clock and the speed of the clock depends upon the no. of pulses per second. With each clock pulse, the processor performs the function that corresponds to the instruction.

This model of the design follows the Von-Neumann computer model. As like in Von-Neumann architecture here is no memory design like RAM, ROM and Cache Memory. This contains some register files like General purpose registers and some registers like Instruction register, program counter. It has an ALU block for performing different arithmetic or logical operations. The design contains the control block for generating the respective control signals for different operations. 8 bit microprocessor contains a number of basic modules which together completes the processor. The processor uses 8 bit data bus to communicate through different sections like General purpose registers, Arithmetic logic unit, CU (control unit), comparator, program counter, address register and instruction register. With the advancement.in integrated circuit technology the power of the processor has increased tremendously. All the modules in the design are coded in VERILOG.

The design has been synthesized with 180nm technology with some data path changes for the optimization. After completing the synthesis by setting some major attributes like connection of TIE cells, clock gating, and setting the design constraints in the SDC constraint file, design was taken forward to physical design of the same. Physical design comprises of Floor planning, power planning, IO arrangements, placement, optimization, Clock Tree Synthesis (CTS), timing checks, Routing, DRC checks, GDSII.

2.2 Architecture Of The µp

The architecture of the designed processor is shown in the figure 2.1 as follows

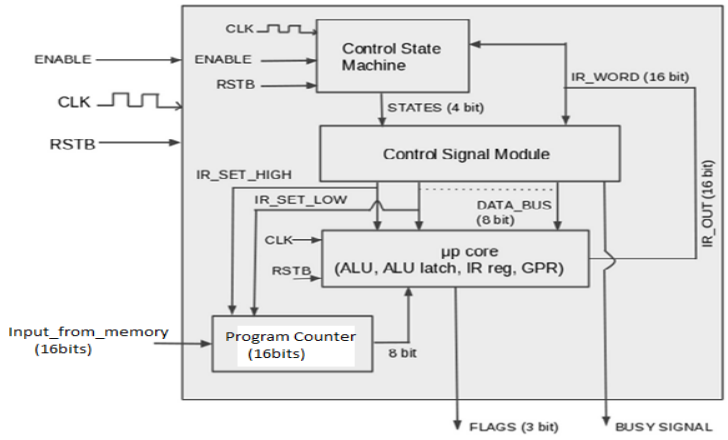


Figure 2.1 µP architecture

According to the architecture, this design contains:

8 bit Microprocessor, µp Core, Control states module, State diagram, Control signals module, Instruction register, ALU and ALU latch, General purpose register, Addressing mode format, Test bench block diagram [24].

2.3 Structure of the Processor Core

The structure of the processor designed contains the GPR, IR, ALU, ALU_LATCH and the different control signals which is written in the µPCore module and is shown in figure 2.2.

2.4 Control States Module and State Diagram

There is module called as the control module which controls all the operation to be performed by generating the appropriate control signals. Hence the control state and the state diagram for the processor are shown in figure 2.3 and figure 2.4 respectively.

2.5 Supported instructions and their opcodes

The processor supports a variety of instructions for their operation and for executing all these instructions the control state machine generates some opcodes and those are described in the Table 2.2 as follows [24].

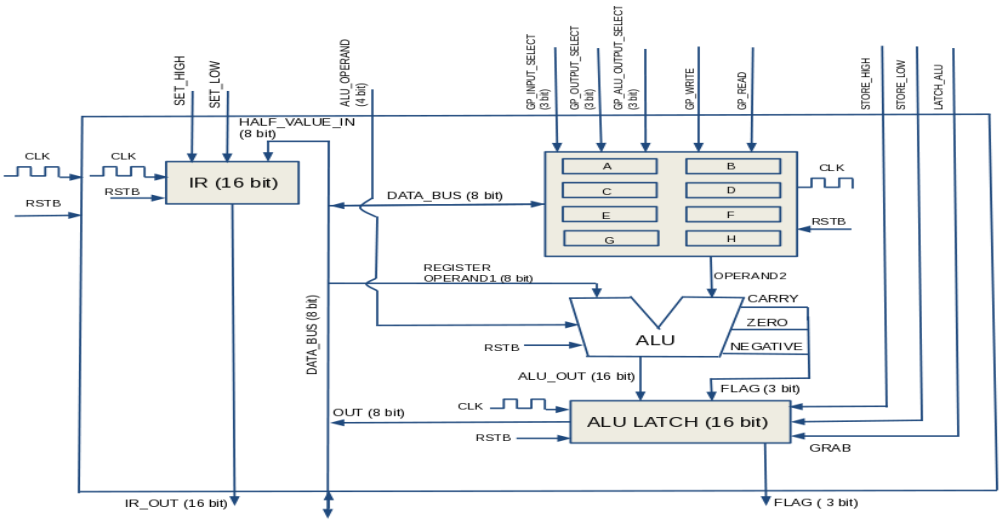


Figure 2.2 Core of the processor

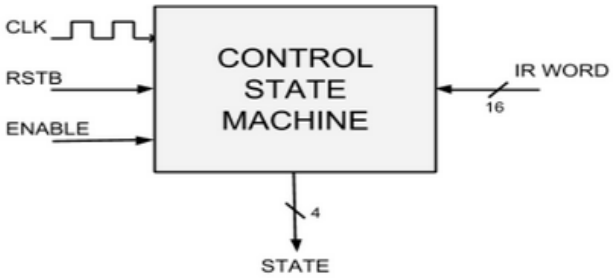


Figure 2.3 Control State Module of Processor

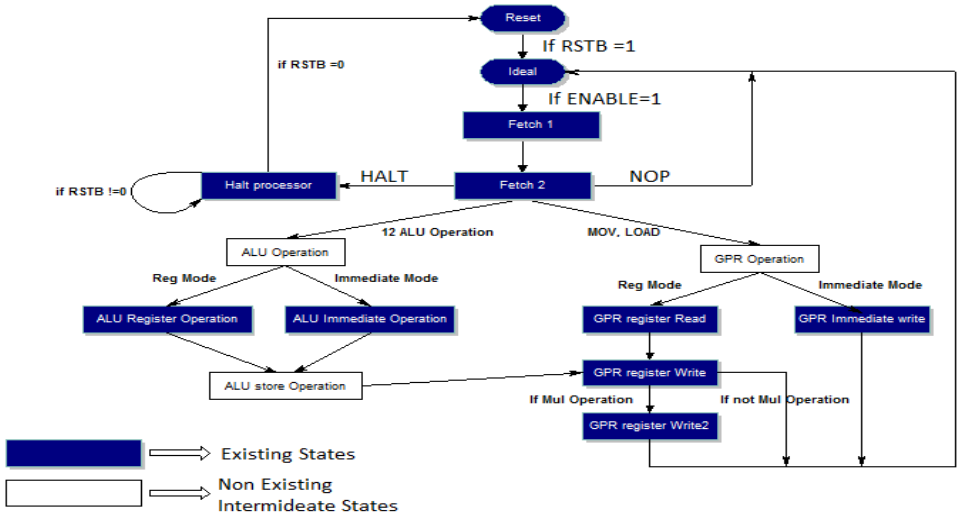


Figure 2.4 Control State machine for processor

The control signals generated in each state is tabulated in Table 2.1.

Table 2.1 Control signals from control module

STATE	**CONTROL SIGNALS**
Reset	BUSY_SIGNAL = 1'b1
Ideal	BUSY_SIGNAL = 1'b0
Fetch 1	IR_SET_HIGH = 1'b0
Fetch 2	IR_SET_LOW =1'b0
ALU Resister Operation	LATCH_ALU = 1'b1 GP_READ = 1'b1(if ADD,SUB,MUL,AND,OR)
ALU Immediate Operation	LATCH_ALU = 1'b1
GPR register Read	GP_READ = 1'b1
GPR Register Write	GP_WRITE = 1'b1 ALU_STORE_LOW = 1'b1 (if not MOV,LOAD) GP_READ = 1'b1 (if MOV,LOAD
GPR Register Write 2	GP_WRITE = 1'b1 ALU_STORE_HIGH = 1'b1
GPR Immediate Write	GP_WRITE =1'b1

Table 2.2 Instructions and their opcodes

INSTRUCTION	OPCODE
ADD	0000
SUB	0001
MUL	0010
AND	0011
OR	0100
LSR	0101
LSL	0110
ASR	0111
ASL	1000
COMP	1001
NEG	1010
END	1011
LOAD	1100
MOVE	1101
HALT	1110
NOP	1111

This processor supports two modes of addressing known as the Immediate Addressing Mode and Register Addressing Mode. The code for the core to understand about the mode of addressing is given as

0: Register Addressing Mode

1: the Immediate Addressing Mode

This also generates three type of flag status known as Carry Flag, Zero Flag and Sign Flag [25]. The Flag format of generation is tabulated in the Table 2.3 as shown below

Table 2.3 Flag Format

CARRY FLAG	ZERO FLAG	NEGATIVE FLAG

- When there is a carry, flag format is 100.
- When result is zero, flag format is 010.
- When result is signed, flag format is 001

Finally the processor was simulated with the test vectors and the result is shown in Figure 2.5

Figure 2.5 simulation result of processor

2.6 Synthesis of the Processor

Synthesis is the process of generating gate level netlist out of the HDL language. This process of synthesis takes HDL (Verilog or VHDL) files, the timing library, DEF file, LEF file and Cap-Table files as its input. This process has two modes of synthesis such as the WireLoad mode and the PLE (Physical Layout Estimation) mode. In WireLoad mode the synthesis tool predicts the RC values from the length of the net and which is not accurate as the wire length and locations or connections are not the real while in synthesis. This WireLoad model is used in the synthesis for estimating the RC wire delays and providing the timing information which can be very close to the real timing while in the placement stage in the physical design environment. So the WireLoad model stays defined in the timing library and it appears as follows:

Wire_load (“WLM1”) {

Resistance: 0.0002; ---------- > R per Unit Length

Capacitance: 0.0003---------- >C per unit length

Area:0.1 ---------- >Area Per Unit Length

Slope: 1.5 ---------- >Used for extrapolating linearly

Fanout_length (1, 0.002); ---------- > at fanout 1 length of the wire is 0.002

Fanout_length (2, 0.007);

}

As the fanout length is defined up to 2 fanout, and the fanout comes 10 then it is out of bound of definition hence the slope will be used for calculating the delays with the extrapolation as follows:

$$\text{Net length} = \langle \text{length of net at fanout 2} \rangle + (10-2) * \text{Slope} \qquad (2.1)$$

For PLE mode of synthesis the inputs are the DEF for floor-planning, LEF and Capable files so that the RC synthesizer can predict the length of the net with little detailed information from the backend environment which is called as the Back Annotation.

2.6.1 Data Path Optimization

This is a part of synthesis for optimizing the data-path for achieving the frequency requirements with modifying the components of the data-path. The changes in the data-path involve the following possible changes:

I Adder Architecture

II Multiplier at the Gate Level

III Booth Encoding

IV Carry Save Arithmetic

V Carry-Propagate Adder

VI Operator Merging

Hence while doing the synthesis, in the data-path of this processor some number of adders and multipliers are there, for which the change of components were tried. The tool is available with some kind of adders and multipliers, so for the timing optimization we keep trying with different sub-architectures available with the tool and also externally designed adders and the multipliers. Finally the data-path takes the lowest time with the Carry-Save-Adder and the Booth Multiplier.

Through the synthesis some other concepts like Boundary optimization, clock gating insertion can be done.

The synthesizer gives the output as the netlist file and a synthesized SDC file which are must for the physical design. The Processor looks like as follows after the Synthesis step in the GUI.

2.7 Conclusion

This chapter describes about the architecture of the 8-bir processor that has been designed with the Verilog code and again the synthesis of the processor with 180nm technology has been shown above. While doing the synthesis, optimization steps such as the data path optimization, boundary optimization are also discussed. In the data path optimization, the sub- Architecture of the components such as different adders and multipliers are used in a testing purpose for the timing optimization.

Chapter 3

PHYSICAL DESIGN OF THE 8-BIT PROCESSOR

3.1 Introduction

This is an act of designing a chip by placing the components of the chips such as different gates and arranging different Macros which are already designed by other designers inside a chip. By the name Physical Design, it should be understood that some physical activities are involved in the design. These physical activities are here for making the design to meet the timing requirements. For making timing to meet some physical activities are to place some buffers at some particular places for meeting the transition requirements, to upsize or downsize the cells for reducing the cell delays and to touch some wire-length reduction etc. now we will discuss the different steps involved in the physical design such as the Floor-planning, Power Planning, Partitioning, Placement, legalization of placement, Clock Tree Synthesis (CTS), Routing.

3.2 Floor-Planning:

The physical design of the chip starts with the floor-planning and this is the major one for getting the layout done. And this floor-planning defines the size of the chip/Block, allocates Power routing resources, places the Hard Macros and finally reserves space for the standard cells [16]. After this all the subsequent steps like Placement, Routing and Timing Closure will depend on how good is the floor-planning is. In real the Floor-planning need to go through many iterations to reach to an optimum floorplan. Different related definitions are described below.

3.2.1 Core Boundary

Floor-planning defines the size and shape of the chip/block. A top level Digital design can have a rectangular/square shape, whereas a sub block may have rectangular or rectilinear shapes. The area where the Standard cells and other IP block will seat is known as the Core Boundary. Power routing spaces are provided outside of the Core Boundary. Floor-planning is controlled by various parameters:

3.2.1.1 Aspect Ratio

This is defined as the ratio of the height and the width and this determines whether you get a square or rectangular floor-plan [16]. A Square shape of a chip indicates that the aspect ratio is 1.

3.2.1.2 Core Utilization

As described above core area is the area where the standard cells seat. And out of this core area the areas which are utilized through the cells or macros defines the Core Utilization and is given as

$$\text{Core Utilization} = (S_a + M_a)/T_a \quad (3.1)$$

$S_a = Area\ of\ the\ chip\ which\ are\ occupied\ by\ the\ standard\ cells$

$M_a = Area\ of\ the\ chip\ which\ are\ occupied\ by\ the\ Macro\ Placement$

$T_a = Total\ area\ of\ the\ chip\ i.e\ the\ Core\ Area$

A Core utilization of 0.7 means that 70% of the total area is available for the placement of the standard cells and remaining 30% is for the routing of the chip.

3.2.1.3 IO Placement/Pin Placement

While doing a digital-top design, we need to place IO pads and IO buffers in the chip [16]. The RTL designer actually provides the side and the relative positions of the PADs. The package is also selected by taking the minimum or maximum die size. To place IOs some modifications are required in the RTL file for connecting the PAD cells to the IO ports by instantiating them.

If a block is getting designed then the pins of the block needs to be placed around the boundary to connect to the Higher Routing Levels. DEF files also can be used by the tool for a custom floor-plan.

3.2.1.4 Macro Placement

Once the floorplan is ready up to the size and the shape and thereby creating the rows for the standard cells, the Macros can be now hand-placed. The location of the Macros where they will sit, will depend on the flylines connection between the macros and the IO ports thereby making the connection wire length smaller.

3.2.2 Power Planning

Power planning starts with the power ring that creates a ring for the chip with VDD and VSS. This power lines for the chip is created with the highest metal available with the technology because these line will always handle a heavy power supply. As the highest metals are having more width hence the resistivity of the metals is less so the IR drop will be less. Then we do special routing which creates the tracks for the proper placement of the standard cells. Those tracks are created with the metal1 with alternative VDD and VSS so that the cells will get power from those. So to maintain the integrity of the metal1 lines from IR drop we add the power

stripes with higher metals so that the stripe will strengthen the metal1 lines. As the track goes horizontally the stripe goes vertically [16].

Here comes now the connection of the Tie and Tap cells an d their definitions and importance are described as follows:

There many types of special cells available in the library which are non-functional in nature but are having great many requirements and importance and those are known as Well tap , End cap, Decap, Spare, Filler and Tie cells

- Well Tap cells: used to connect the substrate and N-well to VDD and VSS respectively to avoid Latch-up effect.
- End Cap cells: It's just a poly extension and is used to ensure that gaps do not occur between well and implant layers to prevent DRC violations
- Decap cells: it's a capacitor for decoupling as no voltage sources are ideal and hence due to glitches in power lines the gate draws heavy power. So to avoid the ground and power bounces decap cells are used
- Spare Cells: used to support the ECO changes after the Tape out of design
- Filler cells: Filler cells are used to establish the continuity of the N- well and the implant layers on the standard cell rows
- Tie cells: used to connect the ports of some gates to VDD and VSS those are 1 or 0 respectively

The Floor-planning and power Planning of the chip looks like as shown in the Figure 3.1

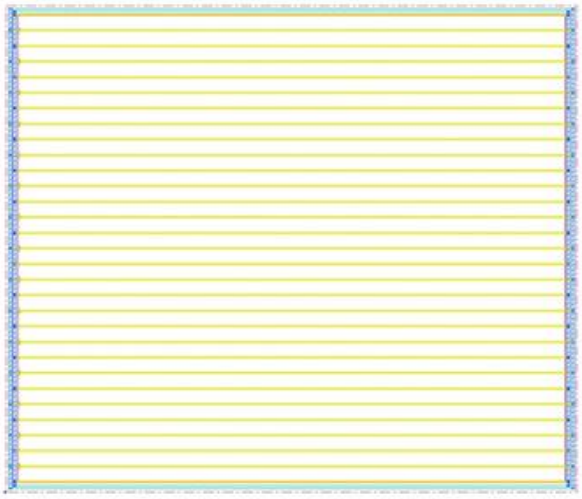


Figure 3.1 Floor-planning Representation

Then comes the power planning with the connection of the Tap and Cap cells. For power planning the metals connections are the upper one available in the technology library as the higher metals are wider so that the resistance will be less and hence the IR drop across the power lines will be less. So by maintaining higher metal lines, all the cells will get the power with less distortion. Hence the arrangements with all the above mentioned concepts are shown in Figure 3.2

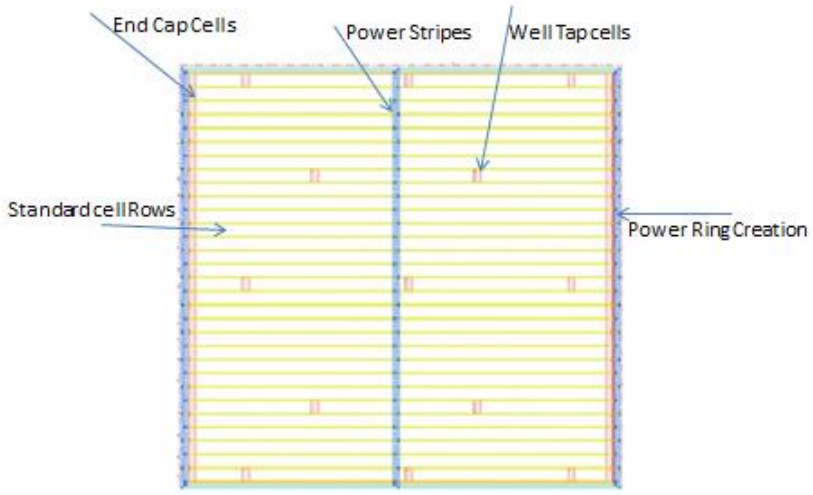


Figure 3.2 Power Planning with Tie and Cap cell connection

3.3 Placement

After the design is ready with the floor-plan we go for placing the cells in the core area of the chip. During the placement of the cells so many things needs to be taken care such as the density limits, overlapping of the cells, padding of the cells etc. so the placement can be of two types such as the Timing Driven Placement and the Congestion Driven Placement. So the difference between these two are that the placer will give importance for reducing the wire lengths in case of Timing Driving Placement whether for this congestion increase while in case of Congestion Driven Placement the placer will give importance to reduce the congestion by spreading out the cells throughout the chip whether that lead to worst timing performance.

While doing the placement step we have to choose for optimizations known as the In-Place Optimization and the Pre-Place optimization. While doing the Pre-place optimization the tool will optimize the netlist by collapsing the High Fan-out Nets and removing any dead paths. While doing the In-place Optimization the tool will optimize the placement by maintaining both from the timing point of view and also from the congestion point of view.

3.3.1 Cell Padding and Module Padding

When the design is imported in the floor-plan we check for any hold violations. If any hold violations will be there before placement or after placement the design has bad connections from the netlist because as the design is Pre-CTS meaning that the clock is ideal thereby reaching all flops at same point of time. For avoiding these hold violations the tool will add the buffers or delay cells thereby needing space for putting these cells. So for doing this buffer addition the flip flops should be padded with some space around it which is called as the Cell Padding. While the module Padding indicates that the module while placement needs more space inside and around it. The module which is more densely packed with very high Effective Utilization needs some space for avoiding the congestion thereby needing the module padding. After the placement with the appropriate paddings and the optimization the chip looks like as shown in Figure 3.3.

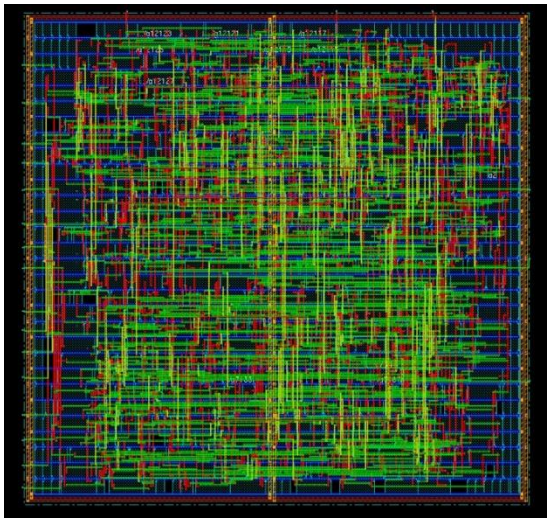


Figure 3.3 Representation of the design after Placement

3.4 Clock Tree Synthesis

As the clock in the design is the ideal one hence we have to take care of the proper distribution of the clock to all the sequential registers. Hence we go for the clock tree synthesis. It makes the clock reach at all the flops with less amount of skew by adding the buffers/invertors in the clock path. In library for clock some special and efficient buffers and inverters are there called as CLKBUF/CLKINV. These are different from the usual buffers/invertors in regard to the transition time, delay and size. The detailed Clock tree Synthesis has been discussed in the following chapters. There are many types of clock trees are available but we used the H-tree type clock tree and after the CTS the chip looks like as shown in Figure 3.4.

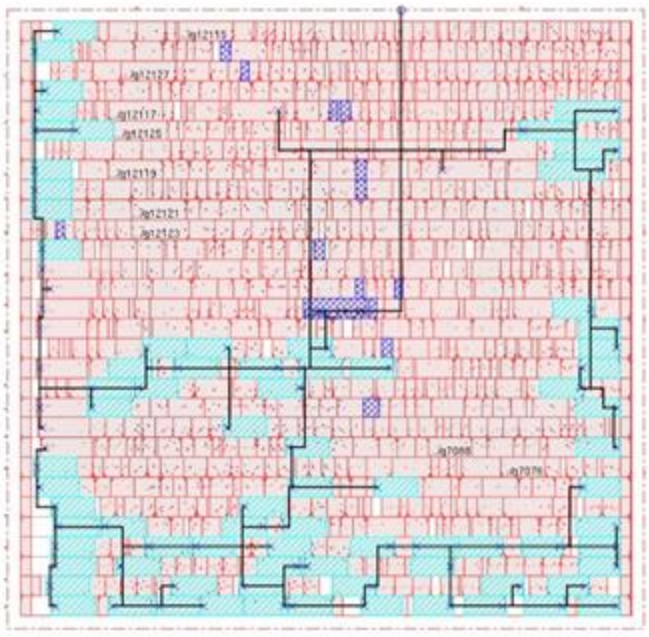


Figure 3.4 Display of Clock Tree Network

Then after the CTS we generate the timing report for the design with the path information's like passing paths and failing paths. For our design all the paths are passing the timing checks with a time period of 4ns. Hence with the timing debug window we can see the timing path reports with green representing the passing paths and red representing the failing paths. And we don't have any red means all paths are passing with a frequency of 250 MHz and the report looks like as shown in the Figure 3.5

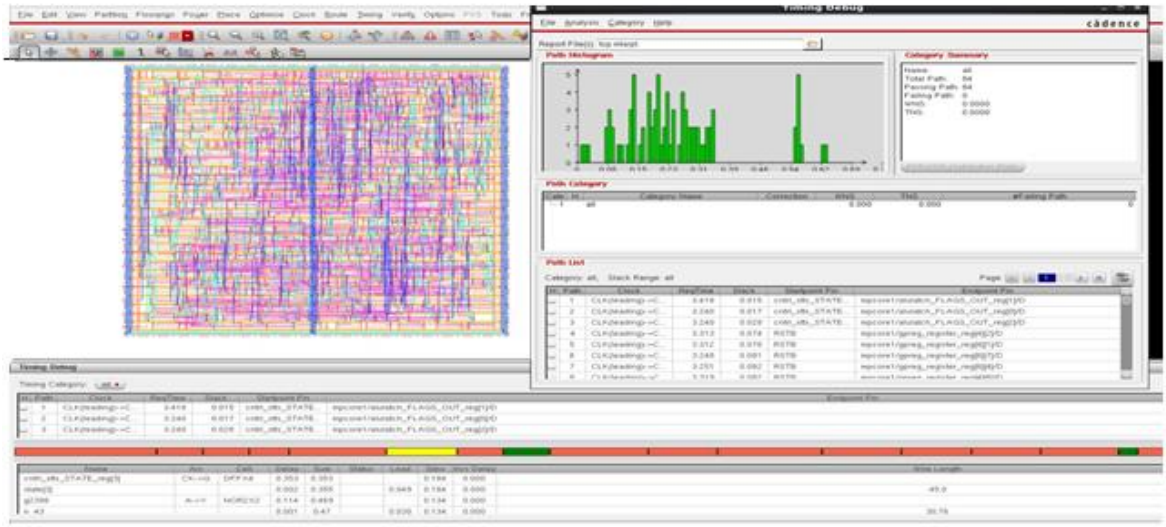


Figure 3.5 Representation of timing path in timing debug window

3.5 Routing

The routing is to locate a set of wires in the routing space that connect all the nets in the netlist. Routing is a very important step which usually follows the clock tree network building as because the clock tree routing more preferred than the signal routing. So Routing is of two types such as the Global Routing and the Detailed Routing (Nano Routing). The Routing takes Netlist, Timing Budget for Critical Net, Locations of Blocks and Location Pins as Input and gives Geometric Layout of all the nets as Output. The Objective of the Routing step is to minimize the wire length, the number of Vias, and just completing all the connections without increasing the chip area and each net meets its timing budget. The techniques involved in the tool for routing is shown in the Figure 3.6.

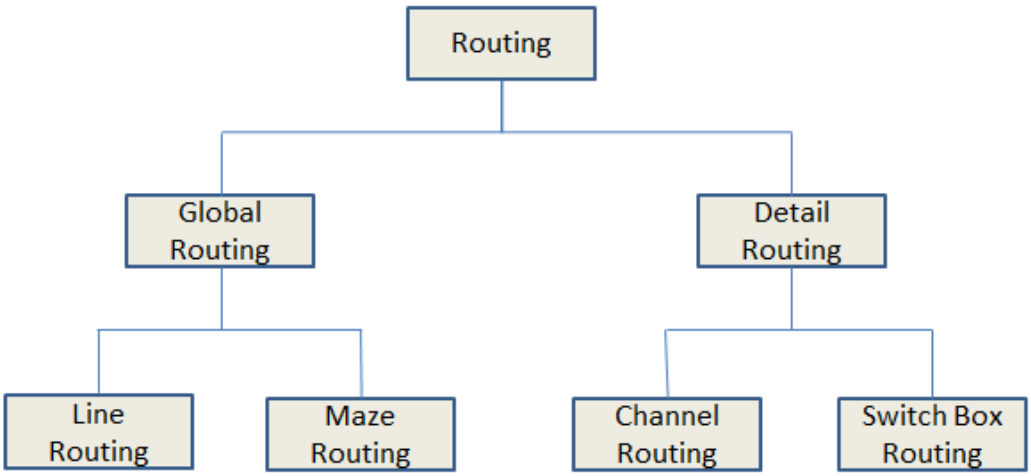


Figure 3.6 Routing Techniques in the tool

3.5.1 Global Routing

In this routing stage the router route the connections from the netlist i.e. the logical connections. This is not the proper routing as it is not optimized for anything. This routing just involves the connections which are driven from the Netlist. This routing is otherwise called as the Trial Route. Through this Trial Route we check the routing congestion by allowing less number of metals layers for routing and observing the congestion. If by allowing half of the available metal into routing and the congestion report is less then the design can be considered as routable. In this way the Global Routing comes into work.

3.5.2 Detailed Routing

As the name says, the router routes with much detail thereby reducing the wire lengths and optimizing the design in respect of the timing expectations of the designer. For this the router first tries to find the channels to route thereby reducing the wire lengths. As the wire

lengths are minimum hence the RC interconnects will have less RC delays hence reducing the time and enhancing the frequency of operation. If there exists any routing congestions then those congestions look like as shown in the Figure 3.7.

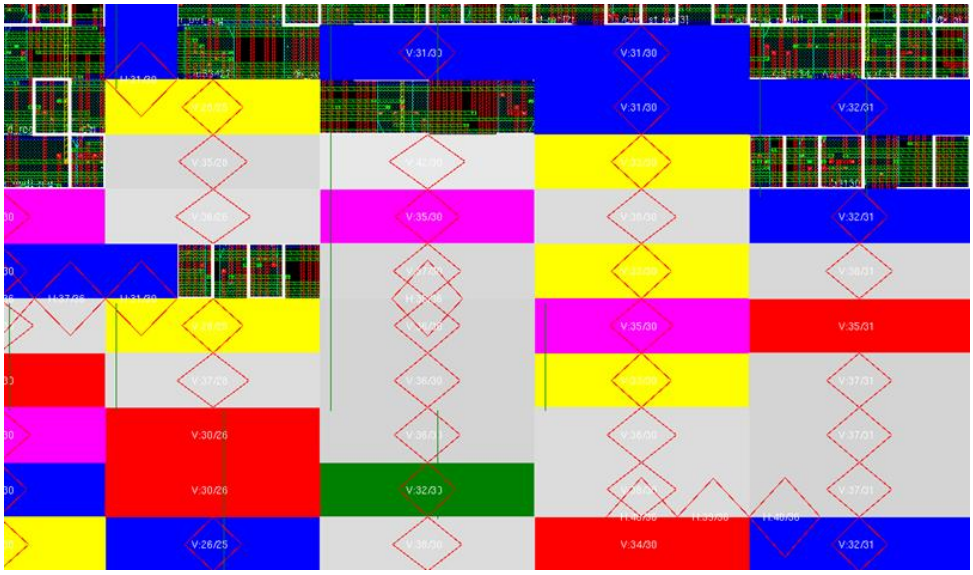


Figure 3.7 congestion representation

3.6 ECO changes

ECO stands for Engineering Change Order. Here the designers change some parts of the design which are implemented by the tool for some kind of optimization. The optimization can be related to the timing, routing issues or any after tape-out functional changes. Hence the ECO can be pre-mask and Post-mask. The need for the ECO changes is discussed as follows.

As a chip goes for the design it passes through all the steps like floor-planning, placement, CTS, Routing and all steps are completed with at least 9-10 iterations thereby taking 9 months to 1 year. Sometimes a new chip is to be produced with just incremental changes over an existing chip. In that cases the chip need not to go all the steps again taking 2 year of period. So everything may be carried out in as incremental manner thereby reducing engineering cost, time to market and the manufacturing cost.

There are two kinds of ECO changes and those are

a. Pre-Mask/Unconstrained ECO/All Layer ECO
b. Post-Mask/Freeze Silicon ECO/Metal Mask ECO

3.6.1 Pre-Mask/Unconstrained ECO/All Layer ECO

This is the phase of ECO which is done before the mask generation in the fab laboratory. As after the ECO change the chip will go for the tape-out hence a designer is free to change whatever changes are needed. ECOs can be done at any of the changes such as the post-place, post-CTS, post-Route etc. ECOs are used to

- Fix timing violations – There may be constraints that were missed on specific nets. An ECO can add buffers/delays to control the timing behaviour of the design.
- Fixing bugs – Last minute bugs are the norm. As the tapeout frenzy catches, the simulations might just throw up that bug everyone missed till then. If it can be fixed by an ECO, engineers prefer it in cases where the runtimes and complexity of the design is large enough to warrant a preservation of the existing results/database.
- Adding functionality – You may not see a lot of post-signoff ECOs in this category. But smaller feature additions can be done using ECOs rather than redoing the design and it can happen.

An unconstrained ECO typically has the following stages.

- Adding/Deleting Cells – At this stage, there is no restriction on adding the cells other than design/layout constraints.
- Updating the connections – The net connections needs to be updated for the existing and newly added cells.
- Placement – Tools can automatically place ECO instances, however I find it better to manually place them for best performance.
- Routing – Today's tools can automatically identify the changed nets and route them.

3.6.2 Post-Mask/Freeze Silicon ECO/Metal Mask ECO

These are done after tape-out and saves significant costs in mask generation by targeting only a few layers for new mask generation. The base layers are all frozen and cannot be changed. All or some of the metal layers are changed to achieve the required functionality of the ECO. The reasons for the ECO and the flow stages stay essentially same; but with some significant differences.

Adding/Deleting Cells – No cells can be added or deleted. Technically you can replace cells if the base layers stay the same (say different TIE cells). However, this may be difficult to control.

- Updating the connections – The net connections needs to be updated.
- Placement – No ECO placement is done, as there cannot be any addition of cells.
- Routing – Today's tools can automatically identify the changed nets and route them.

Freeze silicon ECO typically used spare cells available in the design. These are cells sprinkled in the design in anticipation of use in an ECO. See my article on Spare Cells to know about inserting and placing spare cells.

3.7 Addition of Spare Cells, Filler Cells and Metal Filling

3.7.1 Spare Cell Addition

Spare cells are just the extra cells placed in the layout just with the anticipation that in future they may be needed. Here the future means that after the tape out with the silicon in hand. After the completion of the silicon tests, it might become necessary that the design needs some change. If the design needs some logic changes, just considering that the design logic change needs an OR gate for the logic change, so we can just use the existing OR gate with any disturbance to the Base layer. Just the metal masks are changed thereby needing only metal mask regeneration for the next fabrication.

Spare cells are added in the design for the Future ECO changes. The spare cells are added in the design by either adding a verity of cells present in the library by spreading the cells all around the design. Otherwise a module of different cells can be included into the design at a particular place. After the insertion of the spare cells, whenever the ECO process will need those cells which are sitting at different parts of the chip they will be connected through a single layer of metal thereby not disturbing any other metal layers.

- Sprinkle the individual spare cells in your layout, so from any point you may have a reasonably close library cell.
- Group the spare cells in multiple groups and sprinkle/place each group in the layout.

3.7.1.1 Spare Cell Placement

You need to give some thought as to where to place your spare cells in layout.They are not timing critical, and if we do not give any constraints, PnR tool will place them all together.

However, we do not know which area of the layout will eventually require a connection to the spares.so we do spread the cells across the total region of the chip.

3.7.2 Filler additions

In standard cells APR flow, the cells in the design are placed on the row. To make sure that each cells gets power and ground connection, the cells are abutted together so that the VDD and VSS terminal of neighboring cells short together. This makes it possible to tap power only at one point anywhere in the row. But it is virtually impossible to fill 100% of the die area with regular cells. So, we use filler cells to fill these spaces between regular library cells to route power rails. Filler cells are used for connecting the gaps between the cells after placement.

Basically the Filler cells are the Non-Functional cells which are used to continue the VDD and VSS rails.

They serve 2 main purposes:

- They reduce the DRC Violations created by the base(NWell, PPlus & NPlus) layers.
- They help maintain the Power Rail connection continuity.

Filler cells are used to establish the continuity of the N- well and the implant layers on the standard cell rows. This is one of the Fab constraints, for ease in the generation of the masks.

Generally the Filler Cells are added after the Routing and timing closure and before the LVS and DRC.

3.8 Metal Filling

Metal fill is typically added to design data during chip finishing just before tapeout using Physical Verification tools, although it may also be added after tapeout. At this late stage, the focus is on the lithography requirements and ignores the interconnect capacitance implications. Moreover, the fill added at this stage of the design process must follow conservative rules to not disrupt timing. This is no longer acceptable at geometries of 90nm and below. Dummy metal

fills are required to be placed close to signal nets to meet minimum density requirements and may affect circuit performance to a larger degree. Moreover, designs have multi-voltage areas and it gets harder to code this in Physical Verification tools for doing area-based tied fill insertion. After all these above steps the design of the chip looks like as shown in Figure 3.8.

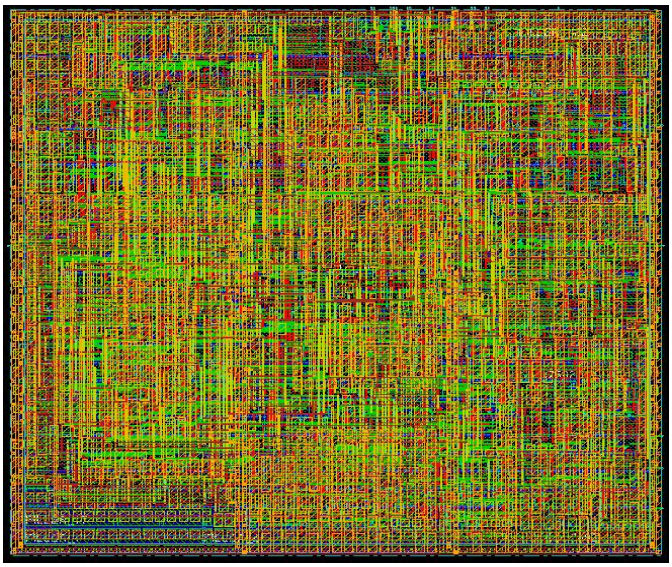


Figure 3.8 Design with filler cells and metal filling

3.9 Conclusion

This chapter described the steps involved in the flow of the physical design. The steps covered are the floor-planning, placement, Clock Tree Synthesis, Routing, addition of spare cells, filler cells and metal filling etc. considerations o be taken while doing the steps are also covered. Concepts behind the ECO changes and "how to do this ECO changes" have also been described above.

Chapter 4

CONCEPT UNDERSTANDINGS OF THE STATIC TIMING ANALYSIS

4.1 Introduction

This chapter talks about the basic understandings and concepts of static timing analysis. This chapter also discusses the previous works done for the better analysis and the ways of doing the analysis. This chapter will cover the timing analysis with setup, hold, recovery, removal, and different corner analysis, the problems associated with the corner analysis, how to reduce the number of corners to be analyzed, how to remove the extra pessimism from the clock network (also called the CPPR-Common Path Pessimism Removal) etc. This chapter will also talk about the effects like the latch-up effect, the antenna process effect etc. The understanding of the libraries such as the timing library (.lib), physical library (.lef), captable, QRC file and the technology file is also included into this chapter.

4.2 Library Understandings

From the name of the library we understand that it contains lots of information in terms of a lots of book hence in this case the library contains a huge and detailed information about the standard cells. The details of the standard cells are also classified in terms of detailed libraries like timing library, physical library and IO library etc. The timing library is represented with the .lib (liberty) extension and the physical library is represented with the .lef (layout exchange format) extension while the IO library is represented with .io extension. Now the question comes what these libraries contain and how these are defined. These libraries are provided by the foundries with the technology node so that after designing the chip can be fabricated with the technology specified conditions. We will start with the .lib file followed by the .lef file.

4.2.1 Timing Library (.lib)

This file contains the timing information for the cells. The timings are the transition times of the pins, propagation delay of the cell depending on the transition time and the output loads with the fan-outs, setup and the hold time of the sequential cells with the comparison of the constrained and the related pin (for a D-flip flop the constrained pin is the clock pin while the data pin is the related pin), the rise and the fall time of the cells with respect to the input pins, the unateness, and finally the power information of the cells with the rising and falling timing arcs. In short this .lib actually is an ASCII representation of the timing and power parameters associated with any cell in particular technology. These timing and power information's are found by simulating the

cells under a variety of conditions like with process, temperature and voltage variations and the simulated data are represented in the .lib file format. The .lib file contains timing models and the data to calculate the following

- I/O delay paths
- Timing check values
- Interconnect delays

I/O path delays and the timing check values are computed on a per-instance basis.

The Cell based delay calculation is modeled by characterizing cell delay and output transition time (output slew) as a function of input transition time(input slew) and the capacitive load on the output of a cell. Each cell has a specific number of input and output paths like as shown in Figure 4.1.

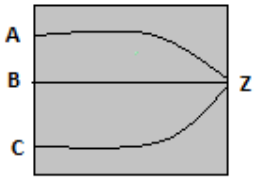


Figure 4.1 Representation of a cell with input and output paths

Paths delays are described as, for each input signal transition that affects an output signal by also depending on signals at other inputs (state dependencies). The delay calculations and the timing checks are as shown in Figure 4.2.

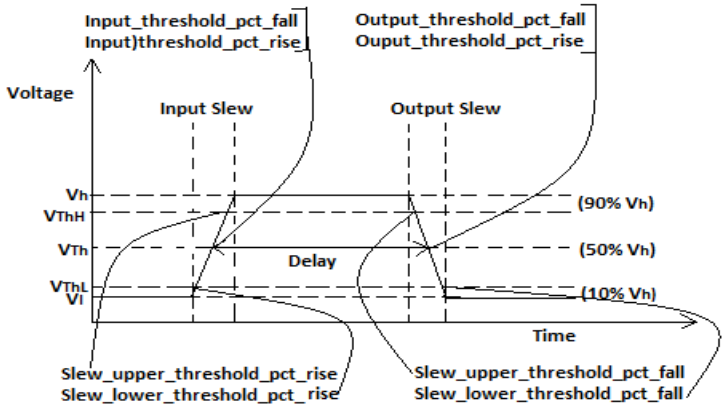


Figure 4.2 Representation of delay calculation and the timing checks

These are the timing calculation related parameters which are defined in the library. These slew related values and the load related values are given in a range of values. Hence the delay values are tabulated with these ranges of the input transition values and the output loads in

terms of NLDM (Non Linear Delay Model) table. Hence when a cell is placed in the design and whatever slew and load this cell will face, according to the ranges of values the respective delay value will be taken from the table for the delay calculation of the path in which the cell is sitting.

What exactly our library file contains is listed as follows:

- Header Information
- Operating conditions, derating factors
- Limits and units
- Cell name
- Area of the cells
- Pin names and the pin capacitances
- Look up tables
- Cell fall and rise
- Pin name and the direction and the function
- Max_capcitance
- Internal power
- Timing
- Pulse width definition, recovery, removal
- Setup and hold timings for flip-flop and latches

This is the end to the discussion about the timing library called as .lib. Now we start discussing about the physical library called .lef file.

4.2.2 Physical Library (.lef)

.LEF file is meant for the physical definition of the cell hence this is called the physical library. This is also represented in ASCII format to describe the standard cell library. This includes the design rules for routing and it contains the abstract of the cell without any internal net list information of the cell. This file has the following sections:

Technology: layer, design rules, via definitions, metal capacitance

Site: site extension

Macros: cell descriptions cell dimensions, Layout of pins and blockages, capacitances

The technology is described by the Layer and via statements. The following attributes are described for each of the layers

- Type : Layer type can be routing, cut,masterslice,overlap
- Width/pitch, spacing rules
- Direction
- Resistance and capacitnce per square unit

- Process antenna factor
- Manufacturing grid definitions
- Macro pin statements

When a cell seats in the design, the information's of the cell size, cell orientation, the shape and coordination of the pins etc. are taken from this physical library. The cells defined in the timing library are mandatorily described with its physical information in the .lef file as a Macro.

4.3 VLSI background and the complexities

Due to the continuous scaling of the technology nodes with respect to the Moore's law which says the transistors numbers on a chip doubles itself in every 18 months, the trend continues till now. Now a days the exponentials design evolution demands high-end performance, speed and low power and high frequency clocks. This increased clock speed in highly dense system-on-chip leads to timing complexity. The variation in the Process, Temperature, Voltage (PVT) and the parasitic effects further enhances the complexity.

Out of all timing is the major part to be considered while designing a semiconductor chip. Besides functional verification, the timing closure is the major element that decides when the chip is to be sent to the foundry for fabrication. At higher technology nodes like 130nm and above the gate delays were dominating over the interconnect delays but now a days with the technology scaling down to the 28 and 22nm the gate delays are very less in comparison to the interconnect delays. And another case was in higher technologies the more separation between the routing of the metal layers causing less cross talk but now in the lower technologies the separarion of the layers are small causing more crosstalk and the delay variation. The phases of domination of delay with the technology scaling trade are shown as shown in Figure 4.3.

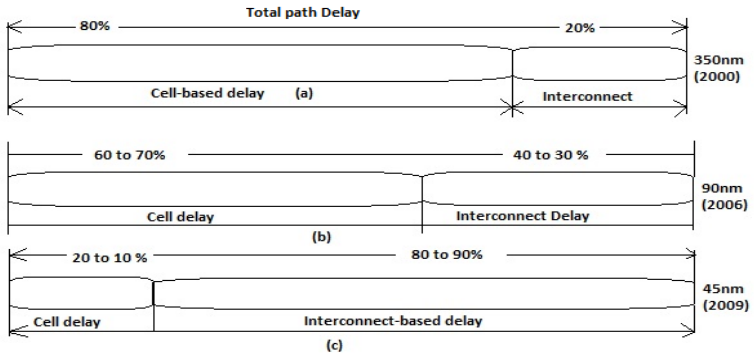


Figure 4.3 Cell and Interconnect delay at Different Technology Nodes

Due to the scaling of technology the length of interconnects increased and the cross section area reduced. Thereby the interconnect delay started contributing to the total data path delay as shown in the above figures (b) and (c).

4.3.1 Problems with the scaling

At lower technology nodes, process voltage and the temperature variations affect the circuit parameters and hence the delay. PVT variations lead to increase in the number of timing verification corners. Due to scaling of technology the voltage level also scales down which creates the situation of temperature inversion and again contributes to the increase in the number of corners. Increased length and the lowered cross section of interconnects increases the resistance of the interconnects. Hence the Increased resistance along with the increased RC delay further raises the number of corners. Hence these increase in the number of corners increase the machine run time and license requirements and hence the cost.

With the continuous rise of the functions, the number of modes also increases which results in a drastic increase of the analysis views. Analysis views are the combination of corners and modes which is carried out using STA tool for TA-signoff closure. The timing closure flow with the consideration of the different corners and the different modes is described in the flow shown in Figure 4.4.

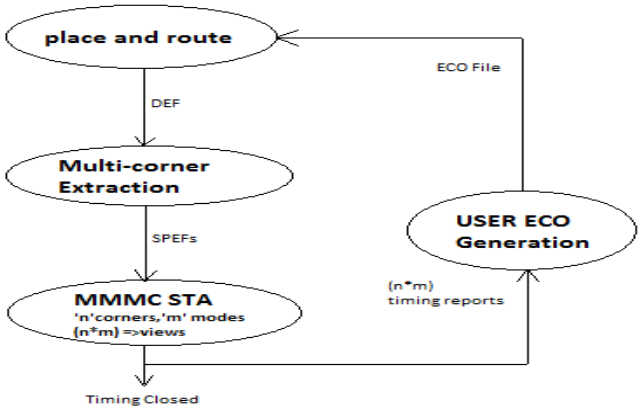


Figure 4.4 TA-sign-off closures Flow

Table 4.1 Analysis of Corners and Modes

Corners(n)	Each corner representing a combination of the parameters like process, voltage, temperature (PVT) and Parasitic (RC) which gives an extreme (best/worst) value of performance
Modes(m)	It is the verity of operations associated with SoC such as Functionality, Test, Scan, BIST, Audio, Ethernet, Video, IO etc.
Analysis View	m * n

If we need at least 8 iterations for every corner and the foundry if gives 12 corners for 45nm technology then by Table 4.1

If 10 functions are involved in SoC;

Then total number analysis views will be 10 * 12 =120

And machine iterations required are 120 * 8 = 960

So to run all corners in each mode parallely and independently then the number of license requirements increases significantly. So the parallel run decreases the run time significantly so there is a trade of between the run time and the machine licenses. For increasing the productivity of the organization we have to identify the critical corners out of all the corners during static timing analysis for meeting the timing constraints.

4.4 Static Timing Analysis

4.4.1 Introduction

Here the theory and the concepts behind the statistical timing analysis will be discussed. This discussion will be including the setup, hold, required arrival time, actual arrival time and the slack variation. Further we will discuss the PVT variations and their effects on the cell and interconnect delays and hence on the timing delay. Also the temperature inversion and its effect on the delay have been covered.

4.4.2 Static Timing Analysis

Static timing analysis is called as static as the analysis of the designs is carried out statically and does not depend on the input data vectors[7]. Another method of verification is there which can verify the functionality as well as the timing of the design, which is called as the dynamic timing analysis. The real purpose of the timing analysis is that can the design operate at the rated speed. That is the design can operate safely at the specified frequency of the clocks without any timing violation.

The design is iterated with the timing checks like setup timing check which ensures that the data should arrive at a flip-flop within the given clock period while the hold check confirms that the data is held for at least a minimum time so that the flop captures the correct data. These checks ensure that the proper data is ready and available for capture. The design is usually specified using a hardware description language such as VHDL or Verilog. The design is constrained through the clock definition in a file called the Synopsys Design constraint (SDC) file. This constraint file includes the information about the generated and the propagated clock, input and output delay, clock uncertainty for the setup and hold check in both the clock and data paths, latency etc. other inputs required for the STA to be performed are the .lib file and the SPEF file that contains the information about the parasitic.

4.4.3 Why static timing analysis

Static timing analysis is a complete and exhaustive verification of all timing checks of a design. Other method as before told requires input stimuli for the verification hence as now a day

a single chip is containing a billions of gates, verification with the input stimuli is very difficult part to do. While STA do the timing check completely and exhaustively as it is independent of the input stimuli's. The stages of applying this static timing analysis are shown in the figure 4.5.

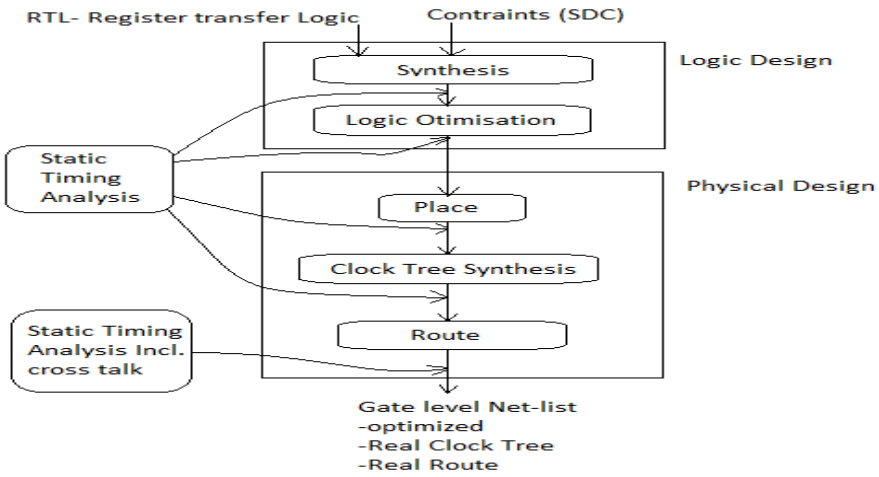


Figure 4.5 CMOS Design Flow at different levels of SoC design

STA is rarely done at the RTL level as at this point functionality is more important compared to the timing. STA can be performed prior to the optimization with the aim of identifying the critical or worst timing paths.

At the logic design level, interconnects are assumed to be ideal as there is no physical information available of the placement and routing. Here the wire-load models are used for estimating the length of interconnects. Here the wire-load models give the estimated RC values from the fan-out of the cells.

4.4.4 Terms and Concepts in STA

4.4.4.1 Propagation Delay:

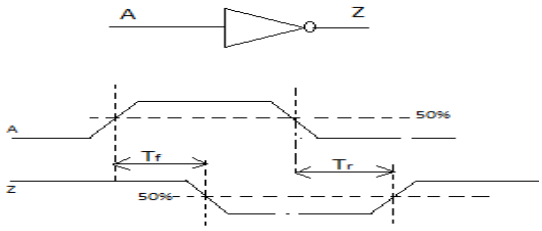


Figure 4.6 Propagation delay measurement waveform

Propagation delay is measured from the 50% threshold when the waveform provides transition from logic 0 to 1 or vice versa when data propagates from input to output.

4.4.4.2 Slew:

Rate of change of the signal is defined as the slew rate. In STA the rising and falling waveforms are measured in terms of whether the transition is slow or fast. Slew is typically measured in terms of the transition time that is inversely proportional to transition time. This is a constraint which is mostly applied on the signal nets or at any pins of the cells. For satisfying the slew requirement the tool adds the buffers and invertors in the net where the constraint needs to be satisfied [7].

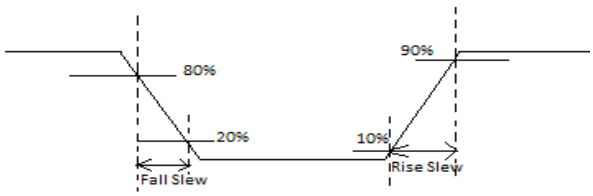


Figure 4.7 Rising and falling edge waveform

4.4.4.3 Skew:

Skew is the difference in time between the arrival of a signal at two different places or between two or more signals, maybe data, clock or both. For example if a clock tree has 100 end points with a skew of 20ps then it means that the latency (source + network) between the shortest and the longest path is 20ps. Skew is a major important factor for maintaining the system frequency which is different from the clock frequency. So the skew is maintained according to the user by different techniques such as the skew grouping or by the clock tree synthesis etc. So the skew is helpful for the setup time meet and is bad for the hold time meet [7]. So this skew is the addition to the clock frequency which in term becomes the system frequency. The More the skew the less is the system frequency. The skew between the arrival of clock at two pins is shown in Figure 4.8.

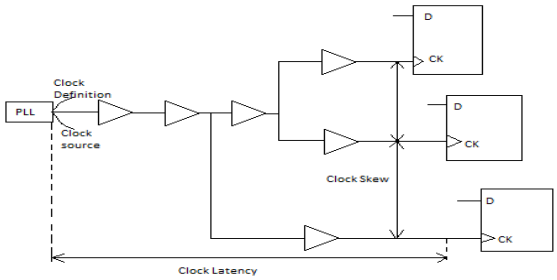


Figure 4.8 Representation of skew in a design

Clock latency is the time taken by the clock to reach to the clock pin of a sequential cell from the clock definition point of the design and the Clock skew is the difference between arrivals at the end points of the clock tree.

STA is often performed with ideal clock trees i.e. no source and network delay so that the focus of the analysis is on the data paths. In ideal clock tree, clock skew is 0ps by default.

4.4.4.4 Uncertainty:

Uncertainty includes the variations in the clock periods called as jitter, skew and the OCV (On-Chip-Variation) at the end of the path. These variations is to be considered more for the setup time as it is fully dependent on the clock period while the hold time is independent of the clock period. Hence while specifying the uncertainty value for the setup and the hold the setup takes more uncertainty value for being more optimistic and the hold takes much less value for these uncertainties while mentioning being optimistic.

4.4.4.5 Min and Max timing path:

Here we have different kind of paths like reg-reg, in-reg, reg-out and in-out. While traversing any of the paths we move through the logic cells like sequential cells, AND gate, OR gate, NOT gate etc. and this traversed path is called as a path delay. So like these paths there can be many paths through which the logic can propagate to a required destination point. Out of these paths we define two extreme paths.

First is called as the max path, which is defined as the paths that takes the maximum delay for the cell and the net delays when signal data or clock propagate through it, this is also called as the worst path. And the other one is called the min path, which is defined as the path that takes the minimum delay for the whole path and also sometimes referred to as the best path.

There is another way of calling max path as late path and the min path as early path. So the max and the min paths is represented with the Figure 4.9.

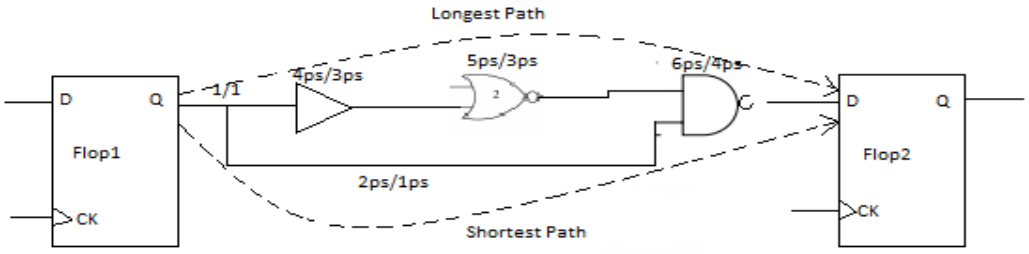


Figure 4.9 Max and Min path representation

4.4.4.6 Operating Conditions

Operating conditions are practical situations to which the designed chip is going to face while in operation. Usually the operating conditions face lot of variations which i are called the PVT variation (Process Voltage Temperature variations). While doing the design a file named MMMC (Multi Mode Multi Corner) in which the designer try to mention the variations in a range for this process voltage and the temperature. The range given is always the extreme conditions for the variations so that the chip doesn't fail for any operating conditions. The delay of the cells and the interconnects are heavily dependent on these kind of variations. For these delay variations we apply some factors called as the pessimistic parameters for accurate analysis of timing. So the delay variations with respect to these process, temperature and voltage are shown in the Figure 4.10.

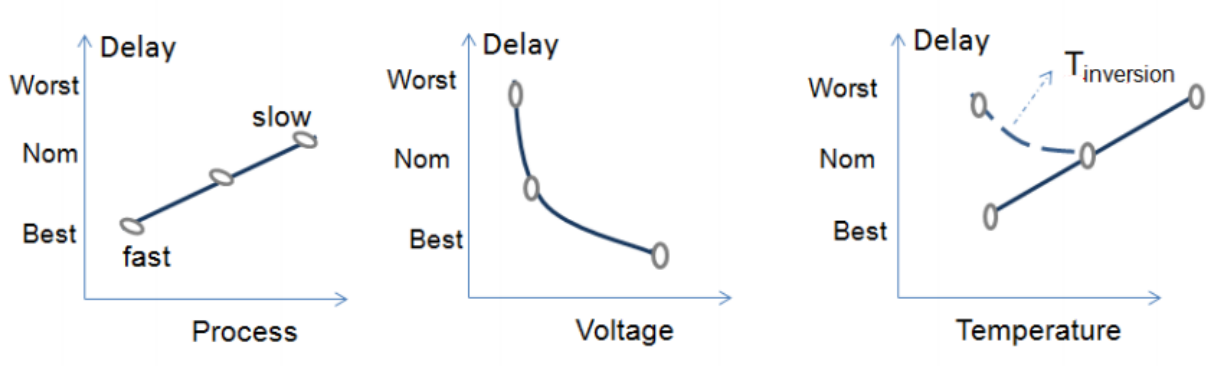


Figure 4.10 delay variations with the PVT

4.4.4.7 Setup and Hold Explanation

These two are the primary timing checks in the digital environment. These two checks are performed rigorously in design as if any of them fails the whole chip will not meet the frequency requirements and the chip will start malfunctioning. So now we will talk about how to perform these two checks. Setup time is checked at the end of the time period while the hold time is performed at the beginning of the period. So we will go little deep into the each of these checks.

(a) Setup checks:

This check is performed between two flip flops while talking to each other; this means that when there is a data-path between two flops there comes the setup checks. There is a timing window in every clock period before which the data should come from launching flop and reach to the capture flop. If the data comes in that window then the check will be called check with setup violations. So the setup window can be as in the Figure 4.11.

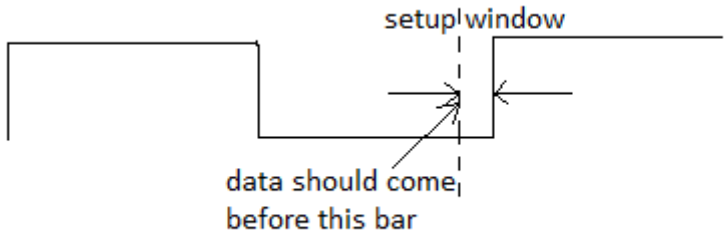


Figure 4.11 Representation of setup window

So the setup time analysis is done with respect to the above mentioned window. This description of setup is basic definitions of setup time. The setup time and the pessimism for the setup time while with the variation will be discussed in the CPPR section as followed.

If setup violations stays in the system then the remedies for this violation is described as follows:

1. Reducing the extra delays in the path due to the buffers which will reduce the cell delays but will increase the wire delays. But decrease in the cell delays will be more effective for the reducing the setup violations in comparison to the wire delays
2. Replacing one buffer with two invertors in a proper interval for maintaining the proper wire transition and hence reducing the cell delays subsequently.
3. Changing the HVT cells in the path with the LVT/RVT cells so that the cell delays will be less and hence the total path delay will also decrease.
4. Increasing the driver strength also called as upsizing the cells so that the delay of the cells will decrease as the cells are now wider than the previous cells. The demerit of this is the power consumption will increase.
5. Addition of repeaters in the path in case of long wires so that the transition increases in that wire and hence the cell connecting to it will behave with less delay so the path delay will reduce.
6. Adding skew in the path called as the Useful Skew that the time for data to reach before setup window will increase and the violations will come down.

Finally with all these steps if the violations do not come down then we will have to increase the clock period. This check is performed in the path by taking the worst case delay in that path so that if the setup check passes the worst case delay then it will pass for rest of the paths.

(b) Hold Time

This check is also performed between the flops having data-path between them. As like setup there is also a timing window for the hold check-up. The data from the launching flop should come after that timing window so that the hold timing check-up passes. That timing window is

associated with the flops from the library characterization so that timing window for Hold time check-up is shown in the Figure 4.12.

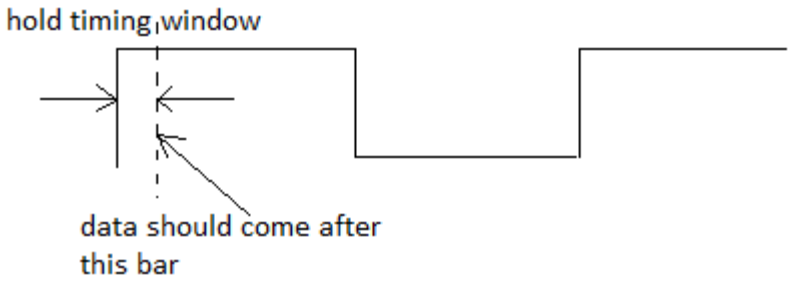


Figure 4.12 Representation of hold timing window

All we discussed are basic concepts of the hold timing analysis. So the real hold timing analysis with the variations and the pessimism factor for these variations in PVT are covered in the CPPR sections followed as follows.

If hold violations stays in the system then the remedies for this violation is described as follows:

1. By adding delay cells in the path by forcing the path delay to come after the hold timing window. Here the noting point will be to consider that adding delay in a path which may have its start and end points in other paths with setup violations.
   If in case the start point of path with hold violation has setup violation with other path so we have to insert to buffer or delay cells nearer to the end point in the hold violating path. All this doubts will be cleared with the following figure in Figure 4.13.

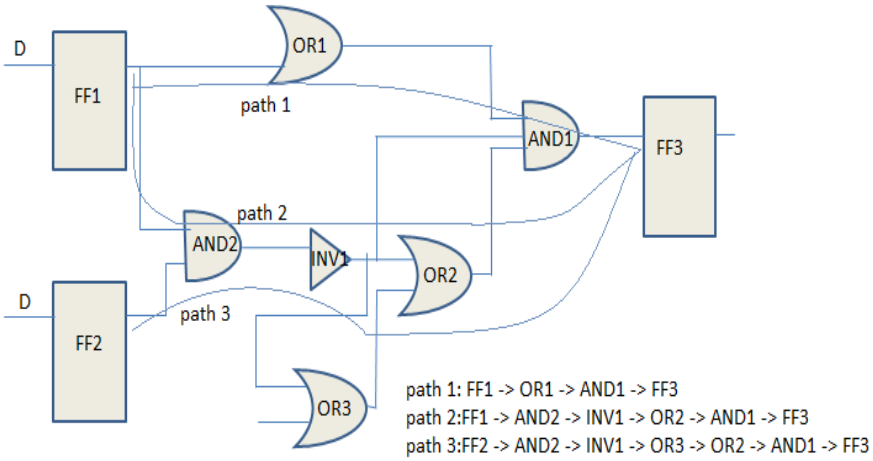


Figure 4.13 Representation of the care that should be taken for the Hold analysis

Table 4.2 Ways to do setup and Hold Violations

Violating Paths		For Fixing the Hold Violations	
Setup	Hold	Do's	Don'ts
Path 1	Path 2	Delay can be added anywhere in the path 2 between AND2 and AND1	Don't add delay between AND1 and FF3 (common section of both paths)
Path 1	path 3	Add delay anywhere between FF2 and AND1	Don't add delay between AND2 and FF3
Path 2	Path 3	Add delay before AND2 in Path3. Can add delay in path3 between INV1 and AND2	Don't ass delay between AND2 and AND1 (common section of both paths)

By decreasing the cell sizes of certain cells in the data path so that he cell delays will be more in the path. It is better to reduce the cells those are close to the capture flops because this way will be less likely to affect other paths and causing new errors.

So in general hold time violations are fixed by the back-end designers during the Place-and-Route while building the clock trees. Setup violations are taken care by the front-end designers.

If a chip is going to the fabrication then it is very important to fix all the hold violations while setup violations may stay in the design. If the chip after fabrication stays with the setup violations then by reducing the frequency or by increasing the voltage in the violated regions the setup time can be taken care.

If a chip after fabrication stays with the hold violation the just dump the chip as nothing can be done for after the chip is fabricated.

4.5 CLOCK TREE SYNTHESIS

Clock tree by its name is indicating that a tree structure is built for the clock network for the proper distribution of the clock in the design. This is the method and structure to make the clock signal propagate in System-On-Chip to a multiplicity of clock sink nets. Here we will discuss the different ways of wiring and buffering so that the tree can be built with minimum number of levels in the clock tree, minimum latency, and minimum clock skew and minimum uncertainty. The clock network can be addressed with two phases such as

(I) Optimal Distribution at the top level
(II) Block or local level based clock distribution

So there is method to integrate these two phases in the system with an automation system. For the growing complexity with the technology scaling the physical geometries will introduce

uncertainty due local random variations and also the impact of parasitic in terms of capacitance and the resistances.

These variations are typically called for SoC design as On-Chip Variations (OCV). There are two types of variations called as Global or Local Otherwise called as inter-die and intra-die variations. Global variations are the variations among the die to die while the local variations cause the performance difference between the transistors in the same die. These variations are modeled as derating factors for calculation of the skew. These factors are calculated as certain percentage total insertion delay. To deal with these variations these derating factors have been modeled so that the timing checks for the setup and hold can be done including these derating factors. Standard engines for the clock tree synthesis are driven by timing closure and these engines are not OCV aware. These are used to fix the setup and hold violations by adjusting the skew values, adding. Removing and changing the buffer sizes and exploiting different clock wire lengths and changing levels also. The different possible variations and mismatches in a clock tree is shown in the Figure 4.14.

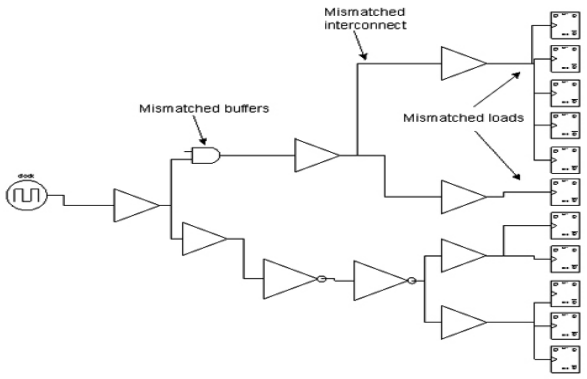


Figure 4.14 Example of a standard clock tree synthesis

The clock tree process is carried out after the placement of the macros and the standard cells. This is because the exact location of the cells comes after the placement. Clock tree network is given more priority for routing first before the real signal routing as this clock signal nets are most frequent than the other signal nets in the design. So for avoiding the interference of the clock net with the signal nets, the clock nets are preferred first for getting routed with the upper metals available after the power routing, so that the clock network can be maintained with less IR drop and the less noisy interference to the signal nets near which the clock net passes. An ASIC design usually has a huge number of I/O ports for the signals entering to the design as input and going out from the design as output and among them clock signal enter with one or two ports from one side of the chip and travels all around the chip. Hence to make the clock reach all the flops with minimum skew or zero skew is very tough task. So to make ease this challenging task of delivering clock signal some type of tree structures algorithms have been adopted by the tool to make the tree for distribution of the clock. So among those tree structures some are given as follows:

1. H-tree
2. X-tree
3. Fish Bone Structure etc.

4.5.1 H-Tree

For a perfect clock tree with perfect synchronization between the clock signals this H-Tree is very efficient in comparison with the others. So in H-Tree the branches move in a rectilinear path like shown in the figure 4.15.

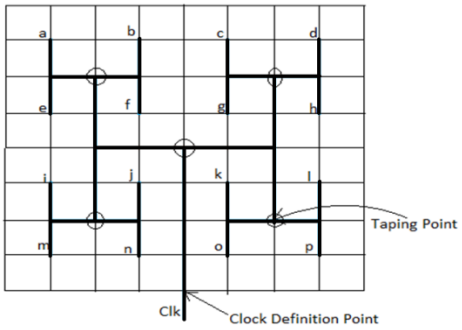


Figure 4.15 H-Tree representation

Here a – p are the clock pins of the sequential cells. And there is a clock definition point because for the clock and external PLL or Oscillator like circuits will be there which would generate the clock then the clock enters to the design through the clock definition point. With the H-Tree Zero skew is achievable easily. All the sequential cells need the clock at the same time. To achieve the zero skew the H-Tree is built in the top level into the different sub-modules as shown above.

4.5.2 X tree

Here the difference between the H-Tree and this X-tree is that the connection isn't rectilinear type. Here the difficulties are cross-talks are more. So the X-Tree is looks as in the Figure 4.16.

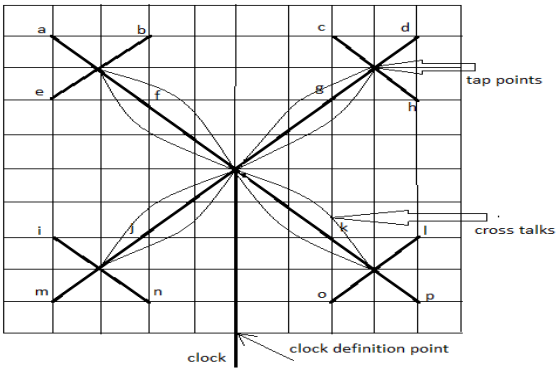


Figure 4.16 X-Tree Representations

4.5.3 Fish Bone Structure

This structure of clock tree looks like the bone of a fish as shown below. Hare the problem with this structure is the clock signal cannot be propagated evenly in the whole design as it goes straight from one end to other. This structure is shown in the Figure 4.17

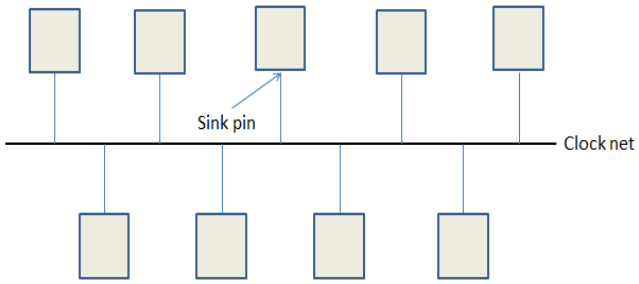


Figure 4.17 Fish Bone Structure

Constraints to the clock tree synthesizer:

Placement database is given as the input to the clock tree synthesizer then the constraints for building the clock tree are Latency, Skew, Maximum transition, Maximum capacitance, Maximum fan-out, list of buffers and inverters etc. clock tree building and tree balancing is done by the tree synthesizer [15].

For maintaining the exact transition (duty cycle) of the clock, Clock tree can be built by clock inverters and clock tree balancing is done by clock tree buffers (CTB) to meet the skew and latency requirements. To make the area and power constraints meet less number of clock tree inverters/buffers should be used. Once the CTS is done, then we have to check the timing again. The outputs of clock tree synthesis are Design Exchange Format (DEF), Standard Parasitic Exchange Format (SPEF), and Netlist etc.

The normal inverters/buffers are not used to build and balance because, the clock buffers provides a better slew and better drive capability while normal buffers and clock inverters provides a better balance with rise and fall times and hence maintaining the 50% duty cycle.

Effects of CTS: Many clock buffers are added, congestion may increase, crosstalk noise, crosstalk delay etc.

Clock tree optimizations: It is achieved by buffer sizing, gate sizing, High Fan-Out Net synthesis, Buffer relocation

4.6 Possible Violations and the Remedies

Violations can be the timing violations and violations related to the constraints those are specified by the designer. If violations happens due to bad conditions of operation the remedy for those violations are given as follows [22].

4.6.1 Setup Fixing

i. Upsizing the cells (increase the drive strength) in data path.
ii. Pull the launch clock
iii. Push the capture clock
iv. We can reduce the buffers from datapath.
v. We can replace buffers with two inverters placing farther apart so that delay can adjust.
vi. We can also reduce some larger than normal capacitance on a cell output pin.
vii. We can upsize the cells to decrease the delay through the cell.
viii. LVT cells

4.6.2 Hold Fixing

It is well understood that hold time will be large if data path has more delay. So we have to add more delays in data path.

i. Downsizing the cells (decrease the drive strength) in data path.
ii. Pulling the capture clock.
iii. Pushed the launch clock.
iv. By adding buffers/Inverter pairs/delay cells to the data path.
v. Decreasing the size of certain cells in the data path, It is better to reduce the cells n capture path closer to the capture flip flop because there is less chance of affecting other paths and causing new errors.
vi. By increasing the wire load model, we can also fix the hold violation.

4.6.3 Transition violation

In some cases, signal takes too long transiting from one logic level to another, than a transition violation is caused. The Trans violation can be because of node resistance and capacitance.

i. By upsizing the driver cell.
ii. Decreasing the net length by moving cells nearer (or) reducing long routed net.
iii. By adding Buffers.
iv. By increase the width of the route at the violation instance pin. This will decrease the resistance of the route and fix the transition violation.

4.6.4 Cap violation

The capacitance on a node is a combination of the fan-out of the output pin and capacitance of the net. This check ensures that the device does not drive more capacitance than the device is characterized for.

i. The violation can be removed by increasing the drive strength of the cell.

Capacitance seen by a output pin can be reduced by buffering some of the Fan-Out paths.

4.7 Conclusion

This chapter included different kind of concepts in the static timing analysis and a detailed clock tree synthesis description. Here different concepts with the solutions have been discussed if these concepts are violated while designing. Different violations such as the setup time, hold time and DRC violations (max transition, max Capacitance, max Fan-out) have been discussed with the cause of violations and the solutions to the violations.

Chapter 5

CLOCK TREE SYNTHESIS WITH REGISTER CLUSTERING AND THE IR DROP AWARE TIMING ANALYSIS

5.1 Introduction

This chapter talks about clustering of the flops in the design with K-mean algorithm after they get placed by the tool. After the placement is done, the clustering of the flops will depend on the total load of the clusters. The load factor for the clusters will depend on the driving strength of the buffers. After the flops are properly placed in the appropriate clusters, proper buffers will be allocated to the clusters to drive the cluster with maintaining the slew constraints given in the clock tree specification file. Here the aim will be construct a well-balanced clock tree with maintaining as much common paths as possible in the built clock tree structure. As much clock paths will be in common the much less optimistic the designer will be towards the clock paths for different scenarios.

5.2 Clustering

It is a concept for the grouping of many data points in a matrix into a group called a cluster. Clustering can be done with a many different aspects like in image-processing, clustering of image pixels with a aim to provide a particular area with a good contrast. Like [2] here we will cluster different flip-flops as data points in the image processing. Like the clustering stated for the image processing applications with the constraint of making some region more contrast region we cluster our flops into different cluster basing on different aspects like the load balancing constraint [4], density constraints, putting some number of flops into a group and forming a multi-bit structure, Density, Rent Parameter, Absorption, Closeness Ratio Cut, Connectivity, etc. So basing on all the stated base of clustering the clustering concepts is working in recent researches. So for clustering there are so many techniques are there for clustering of the data points. Such type of clustering techniques is such as k-mean clustering, Lawler's Labeling Algorithm, Rajaraman-Wong Algorithm, Density Based Search (DBS) algorithm etc. we have adopted the k-Mean clustering for our cluster formation.

5.3 Proposed Methodology for a well-balanced clock tree

Here we will discuss our concepts based on the k-mean clustering. So the k-mean clustering deals with the distance between the flops and makes the groups by assigning the groups with a cluster center. In k-mean algorithm [2], the distance between the cluster center and the flops as data points are measured and it's a pure iterative process that means the cluster forming and the assignment of the cluster centers are changes in every run of the k-mean algorithm as it is a random process. So we will discuss the cluster forming k-mean algorithm in detail which goes as follows:

Inputs to the k-mean algorithm are

$$C_c = \{f_1, f_2, f_3, \dots\dots, f_n\} \text{ With locations } (x_{r_i}, y_{r_i})$$

$$C_g = \{C_1, C_2, C_3, \dots\dots\dots, C_m\}$$

Where the $x_{C_i} and\ y_{C_i}$ are calculated as follows

$$x_{c_i} = (\sum_{j=1}^{k} x_{f_j})/k \tag{5.1}$$

$$y_{c_i} = (\sum_{j=1}^{k} y_{f_j})/k \tag{5.2}$$

Where $f_j \in C_i\ (j = 1, 2, 3 \dots\dots k)$

The distance between the two data points as flops are measured with Manhattan distance measuring method and is described as follows:

$$WL = \sum_{i=1}^{m} \sum_{f_j \in C_i} (|x_f - x_{c_i}| + |y_{f_j} - y_{c_i}|) \tag{5.3}$$

Subject to $C_i \neq \emptyset$

$$C_i \cap C_j = \emptyset\ , i, j \in \{i, j = 1,2,3 \dots\dots\dots, m\}\ \&\ i \neq j$$

$$\cup_{i=1}^{m} C_i] = C_c$$

Where WL= wire length between two flip-flops

Here all distances are measured with the Manhattan distances measures. This was all about the k-mean algorithm of clustering of the data points.

After the clustering of the flops points from their locations with the help of this k-mean algorithm a constraint is applied on all the clusters for modifying the clusters according to the constraints applied. And the constraint here is the total load of the clusters. The buffers are sorted from the library with their load driving capacity. After that according to the manual clock tree

synthesis the load of the buffers intended for driving the leaf level cells (here it is the cluster) is given as the constraint for the clusters to be made. Hence in the k-mean clustering k is decided here with the load constraint as follows

$$k = \frac{T_{capa}}{DB_{capa}} \tag{5.4}$$

Where, $T_{capa} = total\ capacitances\ of\ all\ the\ flops\ in\ the\ design$

$DB_{capa} = capacitance\ of\ the\ buffer\ which\ will\ drive\ the\ cluster$

The clustering is done with this load constraint as deciding factor for the k value. After by the definition of fundamental k-mean clustering the clusters are formed and after that if any cluster's total load is exceeding the max load constraint then the components of that particular cluster will be measured to the nearest clusters with respect of distance between the cluster components and the cluster centers and after that those distances will be sorted in ascending order. The data point sitting in the last of sorted list will start splitting from the violated cluster to the nearest cluster and like this the data points sitting in the last of the sorted list will split till the time the violated cluster's total load comes down to the constrained value of load for the clusters. Like this way all the clusters will be maintained with equal load and now all the clusters will be driven by the same buffers/invertors in the leaf level of the clock tree

The algorithm described above has been maintained in a flow chart which will look like in the following way in the Figure 5.1.

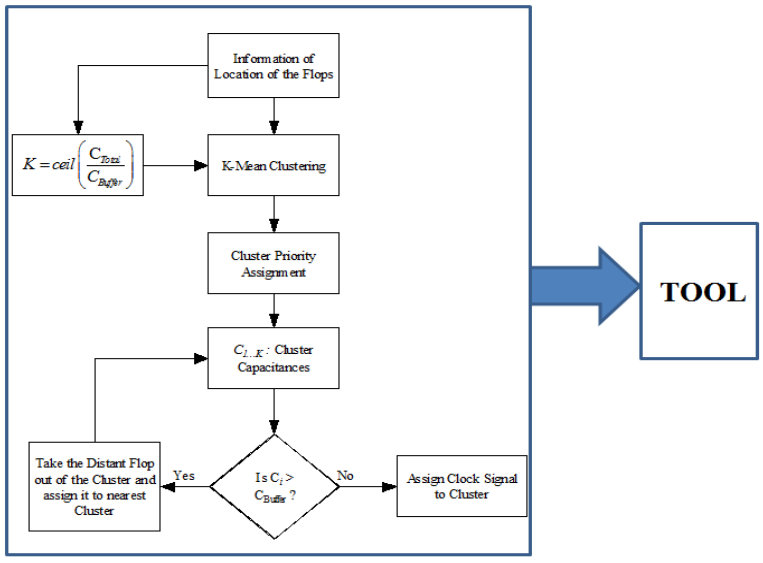


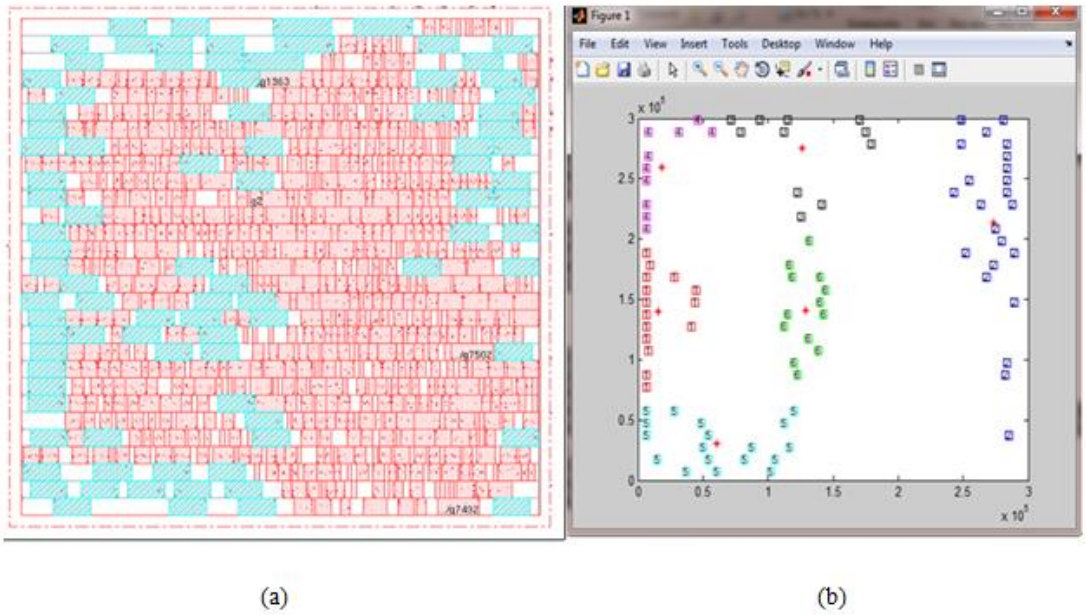
Figure 5.1 Algorithm for clustering and allocating of buffers

The clusters are prioritized in decreasing order with the distance from the center of the chip. This technique is adopted for analyzing of IR drops. As the IR drop increases from one end towards the mid of the chip [9], the tool should be given the information about the flops sitting in the mid region. As IR drop is more in the mid of chip hence the cells sitting in those regions will behave with providing more delay in comparison to other cells. Hence those cells will be slower and hence it is likely that there will be setup violations in the paths containing those cells. Hence to avoid those setup violations the tool will be directed to take some actions in those areas by taking measures as follows:

1. Floor-planning iteration will be done for adding power stripes in those mid regions for maintaining the voltage drop.
2. For the cluster sitting in those regions the CTS engine will provide "Useful Skew" for those flops so that the setup time violations can be taken care but the demerit will be the total overall system frequency will decrease due to this Useful skew addition.
3. The mid regions can be maintained with a voltage region with little more voltage domain but the demerits is that extra level shifter will be used and hence power consumption will be more due to the extra level shifter and the increased voltage level as

$$p_{dynamic} = V_{dd}^2 * f * C_l * \alpha \qquad (5.5)$$

After the implementation of the algorithm through Mat lab coding the clusters look like as in the Figure 5.2



(a) (b)

Figure 5.2 Representations of the Flip-Flops (a) in original design and (b)as cluster points

So now the data points in terms of the flip-flops are grouped into clusters as shown in the figure above. Now as per the load constraint defined in the algorithm the cluster having more total load than the constraint given, according to the algorithm the data points of that cluster will start splitting from the cluster and searching nearest clusters for being a part of that cluster thereby reducing the total load of the previous cluster and making all the clusters of equal load. The splitting and inclusion of the cluster points are shown in the Figure 5.3.

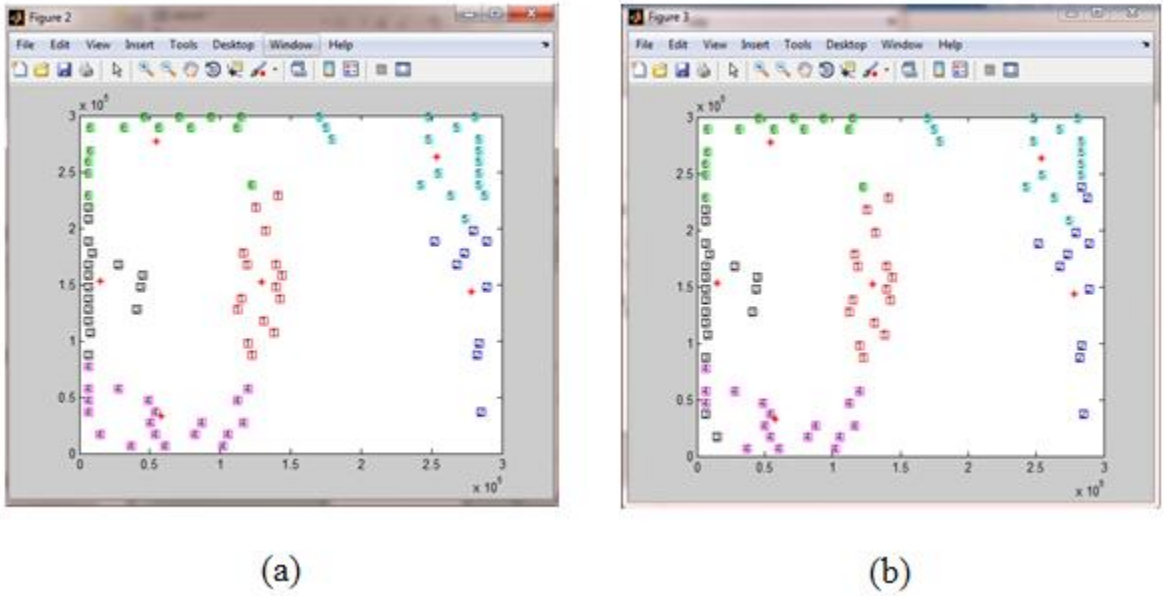


Figure 5.3 Clusters of flip-flops with IR Drop Priority assignment (a) Before load constraint (b) After load constraint

In the above figure as per the algorithm the groups have been prioritized basing on the distance of their cluster centers from the midpoint of the chip so that the IR drops in the mid regions which are the highest, can be dealt with special care. This IR drop makes the cells in that region slower so there is potential that this may cause the timing violation such as the setup and the hold so the following conditions should be satisfied for avoiding the violations.

The set up time condition that needs to be satisfied is given as follows:

$$Cd_l - Cd_c < T_p - \left(CL_{d_{max}} + T_{clk-q} + T_{setup}\right) \qquad (5.6)$$

Where $Cd_l = delay\ of\ clock\ signal\ in\ launching\ path\ from\ source\ to\ sink\ point$

$Cd_c = delay\ of\ clock\ signal\ in\ capturing\ path\ from\ source\ to\ sink\ point$

$T_p = time\ period\ of\ the\ clock\ signal$

$CL_{d_{max}} = maximum\ combinational\ logic\ path\ delay$

$T_{clk-q} = delay\ from\ clock\ to\ q\ of\ a\ flop$

$T_{setup} = setup\ time\ of\ a\ flop$

Similarly the condition for the hold that need to be satisfied is given as follows:

$$Cd_c - Cd_l < CL_{d_{min}} + T_{clk-q} - T_{hold} \quad (5.7)$$

$Cd_l = delay\ of\ clock\ signal\ in\ launching\ path\ from\ source\ to\ sink\ point$

$Cd_c = delay\ of\ clock\ signal\ in\ capturing\ path\ from\ source\ to\ sink\ point$

$CL_{d_{min}} = maximum\ combinational\ logic\ path\ delay$

$T_{clk-q} = delay\ from\ clock\ to\ q\ of\ a\ flop$

$T_{hold} = Hold\ time\ of\ a\ flop$

For avoiding these timing violations the launch and the capture paths in the clock network should be fastened or slowed down depending on the condition.

5.4 Buffer Assignment with Symmetric clock tree structure

The flip flops of the design are clustered with k Mean algorithm with the constraint of [4] that all the clusters will not exceed a load limit that means the sum up of all the clock pin loads of the flip flops should not exceed the limit of load that is set as maximum allowable load for cluster. So that a same type of buffer can drive the clusters so that a symmetric clock network can be achieved. Symmetric clock network is very helpful for making the global skew zero which is very costly [21] without our technique. As a clock tree is built with a constraint to make the global skew as zero, to make the constraint pass the tool will add a huge number of buffers which is very costly in terms of dynamic power consumption by [14].

Clock trees can be automatic or manual with desired number of levels with desired type of buffers. With our algorithm the clock tree leaf level capacitances will reduce to a considerable less value so that the total dynamic power will reduce.

Another efficient clock tree synthesis is to maintain as maximum possible as common path in the clock tree network. So that the OCV variations on this clock path can be maintained minimum, hence the pessimism factor will be less thereby making the system faster by removing the extra pessimism. The clock tree with manual construction is shown in the Figure 5.4 for the clusters created by our algorithm.

ClockNetname	CLK
Level Number	3
LevelSpec	1 1 CLKBUFX1
LevelSpec	2 2 CLKBUFX1
LevelSpec	3 6 CLKBUFX2
PostOpt	NO
MaxDelay	0.01ns
SinkMaxTran	220ps
BufMaxTran	200ps
MaxSkew	20ps
NoGating	No

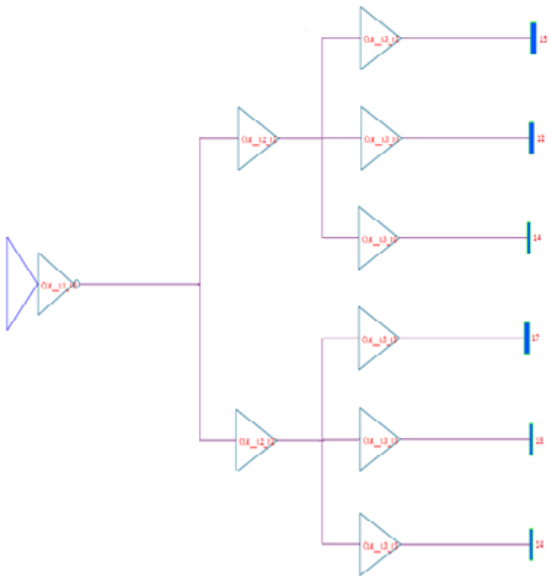


Figure 5.4 Symmetric Clock Tree with Manual CTS with the Algorithm

The minimization in the power of the clock network as per the algorithm is shown in the Figure 5.5. Here the power of the clock network has been reduced by reduced by 11.25% and the power of the total chip has been reduced by 5.76%. This Power reduction is due to the absorption of the proposed algorithm and the clock tree synthesis with manual buffering in the path of the clock network.

The power consumption of the chip with the different section of power consumption before the adoption of the proposed algorithm is tabulated in the Table 5.1 and the power consumption after the algorithm is adopted is tabulated in the Table 5.2. The different power consumption sections tabulated in the Tables 5.1 and 5.2 are the Internal Power, Switching Power, Leakage Power and the total power of the chip.

Table 5.1 Power Consumption of Chip before absorption of the proposed Algorithm

Total Internal Power			3.15701814	66.9067%	
Total Switching Power			1.56084418	33.0790%	
Total Leakage Power			0.00067648	0.0143%	
Total Power			4.71853883		
Group	Internal Power	Switching Power	Leakage Power	Total Power	Percentage
Sequential	1.605	0.1203	0.0001912	1.725	36.57
Combinational	0.9135	0.9153	0.0004395	1.829	38.77
Clock	0.6386	0.5253	4.578e-05	1.164	24.67
Total	3.157	1.561	0.0006765	4.719	100

Table 5.2 Power Consumption of the Chip after absorption of the proposed Algorithm

Total Internal Power			3.04575919	68.4845%	
Total Switching Power			1.40089619	31.4994%	
Total Leakage Power			0.00071507	0.0161%	
Total Power			4.44737047		
Group	Internal Power	Switching Power	Leakage Power	Total Power	Percentage
Sequential	1.656	0.154	0.0001951	1.81	40.7
Combinational	0.7751	0.829	0.0004858	1.604	36.08
Clock	0.6149	0.4179	3.415e-05	1.033	23.28
Total	3.046	1.401	0.0007151	4.447	100

The symmetric clock structure with symmetrical buffers is shown in Figure 5.5 where the buffers at different levels are of similar kind.

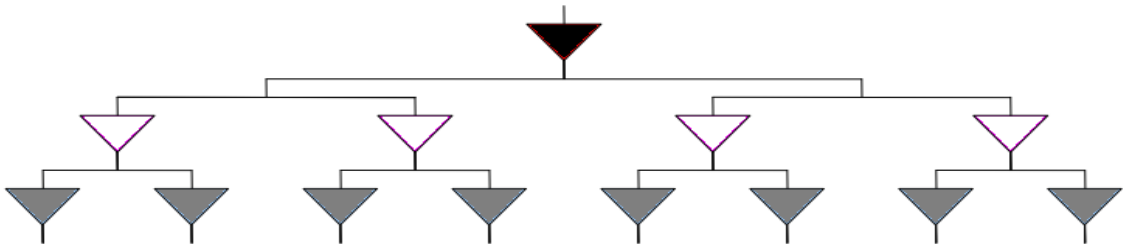


Figure 5.5 Example of a Symmetric Clock Tree

The benefits of maintaining a symmetric clock tree are given below

- All flops of a symmetric clock tree, traced back from the clock tree root are passing the same number of levels and the same cell references at each level.
- The clock tree is balanced at a specific corner which should fit all corners.
- To achieve global zero skew is easier in case of symmetric clock tree while for asymmetric to achieve zero skew a huge number of buffers will be used.

The clock tree with most common paths possible and most uncommon path possible is shown in Figure 5.6 and Figure 5.7.

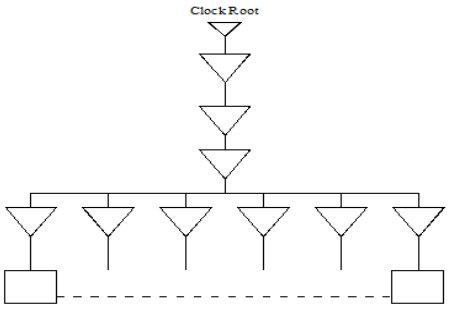


Figure 5.6 Clock tree with maximum common path

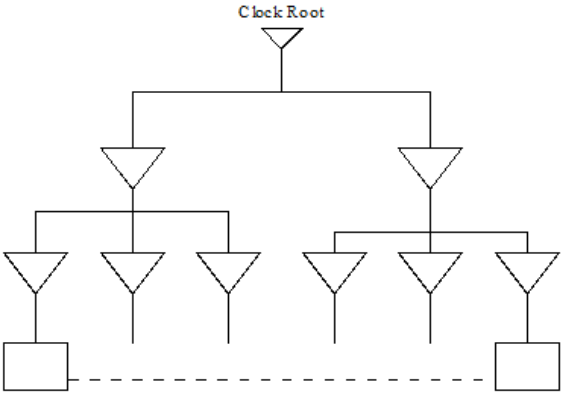


Figure 5.7 Clock tree with maximum uncommon paths

The benefits of maintaining most common clock path are that if the clock path from the root point to the sink point shares the most buffers, inverters and interconnects then if any variations occur with any cell or interconnect then that variation will be same for both the launching and capturing paths thereby making pessimism factor by the designers very less.

5.5 Different other clustering approaches for power saving in the clock network

There are many other techniques which are being used for the reduction of Dynamic power consumption in the design. The dynamic power Consumption is the power consumed when a cell switches its state from one level to another. So to reduce these unnecessary switching (if available) in the design some kind of power saving approaches are being used now a days. So these different techniques are

1 Clock Gating
2 Grouping of flops into a Multi Bit Flip Flop structure (MBFF)

3 Power gating

5.5.1 Clock Gating

Several techniques to reduce the dynamic power are developed, of which clock gating is predominant. Ordinarily, when a logic unit is clocked, its underlying sequential elements receive the clock signal, regardless of whether or not they will toggle in the next cycle. With clock gating, the clock signals are ANDed with explicitly predefined enabling signals. Clock gating is employed at all levels: system architecture, block design, logic design, and gates [3].

For this clock gating the flops are grouped with the activity driven [3] hence the groups are created with taking flops with similar activities and gated with the clock gates. So that when the flops see a similar kind of pattern then those flops can be cut from the clock network for saving the power as the clock network itself consumes up to 70% of the total power of the chip. Hence reduction of the clock network results in power saving. The clustering due to the power gating is shown in the Figure 5.8.

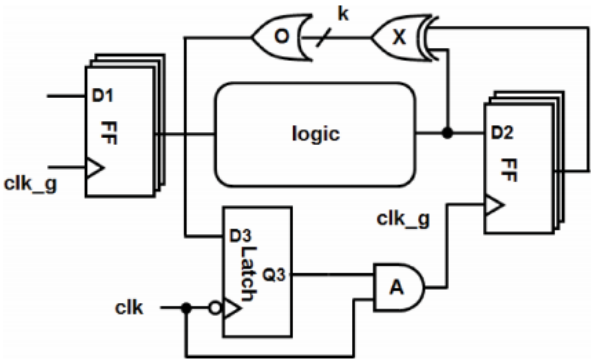


Figure 5.8 Clock gating Structure

5.5.2 MBFF Structure

This aproach [2] is a similar kind of our approach as this also clusters the flip flops into clusters for reducing the power and the structure is shown in the Figure 5.9.

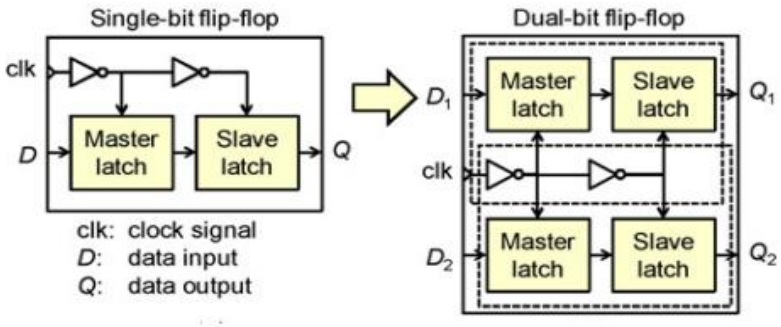


Figure 5.9 Structure of MBFF

5.5.3 Power Gating:

The strategy of power gating is to provide two power modes: a Low power Mode and another is an Active Mode. The goal of this power gating is to switch between these modes at the appropriate time and in the appropriate manner to maximize power savings while minimizing the impact on performance. For this power gating power switches are used from the library. And the size of the switches has impacts like described follows

Smaller Switches -> Smaller Area, larger resistance, good leakage reduction

Bigger Switches -> Larger Area, Smaller Resistance, low leakage reduction relatively.

While doing logic block level power gating the content of some registers need to be retained for which some state retention registers are used from the library. adding the retention registers increases the Area Overhead typically by 20%.

For considering the modules which are to be power gated is a basic step which is decided by the sleep control signals.

5.6 Conclusion

Here we discussed about the different aspects of designing to reduce the power consumption of the chip. As the power consumed in the clock network is 70-75% of the total power consumption hence some techniques with a new algorithm has been proposed for reduction in the power consumption in the clock network. With this algorithm the clock network was also built symmetrically with same kind of buffers at all levels.

Chapter 6

CONCLUSION AND FUTURE SCOPE

This topic concludes with a physical design of the processor including all the relevant steps of the physical design with proper synthesis with data path optimization. The design was optimized in all the steps for better performance. A clock tree synthesis was proposed with the register clustering algorithm with load balancing concept thereby creating a symmetric structure of clock tree and saving dynamic power consumption of the system. After that different other techniques for power savings have been described.

This algorithm has the potential for reducing the total power consumption in terms of power saving s in the clock network. This proposed algorithm needs other algorithms inclusion such as the wire-length minimization from the buffer point to the longest flip flop in the cluster and an algorithm for “how to maintain maximum common paths in the clock tree thereby reducing the OCV influence” into the existing; so that there will not be interconnect capacitance overload on the clusters thereby forming symmetric clock tree and finally with the maximum common path the OCV influence on the clock path can be reduced.

REFERENCES

[1] Ewetz, Rickard, and Cheng-Kok Koh. "Cost-Effective Robustness in Clock Networks Using Near-Tree Structures." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 34.4 (2015): 515-528.

[2] Deng, Chao, Yici Cai, and Qiang Zhou. "A register clustering algorithm for low power clock tree synthesis." Circuits and Systems (ISCAS), 2014 IEEE International Symposium on. IEEE, 2014.

[3] Wimer, Shmuel, and Israel Koren. "Design flow for flip-flop grouping in data-driven clock gating." Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 22.4 (2014): 771-778.

[4] Mehta, Ashish D., et al. "Clustering and load balancing for buffered clock tree synthesis." Computer Design: VLSI in Computers and Processors, 1997. ICCD'97. Proceedings., 1997 IEEE International Conference on. IEEE, 1997

[5] Jiang, Iris Hui-Ru, Chih-Long Chang, and Yu-Ming Yang. "INTEGRA: Fast multibit flip-flop clustering for clock power saving." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 31.2 (2012): 192-204.

[6] Shen, Weixiang, et al. "An effective gated clock tree design based on activity and register aware placement." Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 18.12 (2010): 1639-1648.

[7] Bhasker, Jayaram, and Rakesh Chadha. “Static timing analysis for nanometer designs: a practical approach.” Springer Science & Business Media, 2009.

[8] Hou, Wenting, Dick Liu, and Pei-Hsin Ho. "Automatic register banking for low-power clock trees." (2009): 647-652.

[9] Shih, Xin-Wei, et al. "Symmetrical buffered clock-tree synthesis with supply-voltage alignment." Design Automation Conference (ASP-DAC), 2013 18th Asia and South Pacific. IEEE, 2013.

[10] SELVANAND, P., and U. RAGAVENDRAN. "LOW POWER CLOCK TREE SYNTHESIS USING CLUSTERING ALGORITHM."

[11] Rajaram, Anand, and David Z. Pan. "Robust chip-level clock tree synthesis."Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 30.6 (2011): 877-890.

[12] Lin, Mark Po-Hung, Chih-Cheng Hsu, and Yao-Tsung Chang. "Post-placement power optimization with multi-bit flip-flops." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 30.12 (2011): 1870-1882.

[13] Wu, Guirong, et al. "An efficient clock tree synthesis method in physical design." Electron Devices and Solid-State Circuits, 2009. EDSSC 2009. IEEE International Conference of. IEEE, 2009.

[14] Kiong, Teng Siong, and Norhayati Soin. "Low power clock gates optimization for clock tree distribution." Quality Electronic Design (ISQED), 2010 11th International Symposium on. IEEE, 2010.

[15] SoC Encounter User Guide, www,supportcadence.com

[16] Yongseok Cheon, et al. "Power-Aware Placement." Proceeding of 42nd Annual design Automation Conference. ACM,2005

[17] Padmanabhan, Uday, Janet Meiling Wang, and Jiang Hu. "Robust clock tree routing in the presence of process variations." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 27.8 (2008): 1385-1397.

[18] Padmanabhan, Uday, Janet Meiling wang, and Jiang Hu."Robust clock tree routing in the presence of process variations." Computer Aided Design of Integrated Circuits and Systems, IEEE transactions on 27.8 (2008): 1385-1397.

[19] Lu, Bing, et al. "Process variation aware clock tree routing." Proceedings of the 2003 international symposium on Physical design. ACM, 2003.

[20] Padmanabhan, Uday, Janet M. Wang, and Jiang Hu. "Statistical clock tree routing for robustness to process variations." Proceedings of the 2006 international symposium on Physical design. ACM, 2006. [21] Chao, T-H., J-M. Ho, and Y-C. Hsu. "Zero skew clock net routing." Proceedings of the 29th ACM/IEEE Design Automation Conference. IEEE Computer Society Press, 1992.

[22] "VLSI Concepts". *Vlsi-expert.com*. N.p., 2016. Web. 31 May 2016

[23] Lu, Bing, et al. "Process variation aware clock tree routing." Proceedings of the 2003 international symposium on Physical design. ACM, 2003.

[24] "Microprocessor Design/Computer Architecture - Wikibooks, Open Books For An Open World". *En.wikibooks.org*. N.p., 2016. Web. 31 May 2016.

[25] "FLAGS Register". *Wikipedia*. N.p., 2016. Web. 31 May 2016