TestChip Automation

Project Report

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology

in

Electronics and Communication Engineering

(Communication Engineering)

By

PARTH BANUGARIA

(12MECC41)

Electronics and Communication Engineering Branch

Department of Electrical Engineering

Institute of Technology

Nirma University

Ahmedabad-382481

May 2014

TestChip Automation KPI monitoring through Technical Dashboard for SoC Management

Project Report

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology

in

Electronics and Communication Engineering

(Communication Engineering)

By

PARTH BANUGARIA

(12MECC41)

Internal Guide

Dr. Tanish Zaveri

External Guide

Ms. Sushma Mohan

Electronics and Communication Engineering Branch

Department of Electrical Engineering

Institute of Technology

Nirma University

Ahmedabad-382481

May 2014

Declaration

This is to certify that

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

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

- Parth Banugaria

12MECC41

Certificate

This is to certify that the Project entitled **"TestChip Automation KPI monitoring through Technical Dashboard for SoC Management"** submitted by **Parth Banugaria (12MECC41)**, towards the submission of the Project for requirements for the degree of Master of Technology in Communication of Nirma University, Ahmedabad is the record of study carried out by him under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.

Date: Place: Ahmedabad

Dr. Tanish Zaveri	**Dr. D.K.Kothari**
Internal Guide	**Program Co-ordinator**
Dr.P.N. Tekwani	**Dr K Kotecha**
Head of EE Department	**Director, IT,NU**

Acknowledgement

I am deeply indebted to my thesis supervisors Dr. Tanish Zaveri and Ms. Sushma Mohan, Project Manager ST Microelectronics for their constant guidance and motivation. I would like to express my gratitude and sincere thanks to Dr. P. N. Tekwani Head of Electrical Engineering Department and Dr. D. K. Kothari Coordinator M.Tech Communication Engineering program for allowing me to undertake this thesis work and for his guidelines during the review process. I would also like to thank Dr K Kotecha Director, IT, NU for allowing me to undertake this project work. I also wish to thank Rohit Murarka for his help and support. Without his experience and insights, it would have been very difficult to do quality work.

- PARTH BANUGARIA

(12MECC41)

List of Abbrevations

ASIC ..... Application Specific Integrated Circuit
CTS ..... Clock Tree Synthesis
DEF ..... Design Exchange Format
DRC ..... Design Rule Check
GDS ..... Geometric Data Stream
HDL ..... Hardware Description Language
KPI ..... Key Performance Indicators
LEF ..... Layout Exchange Format
LVS ..... Layout vs Schematics
PnR ..... Placement and Routing
RnD ..... Research and Development
RTL ..... Register Transfer Logic

Abstract

The project deals with automation and customization of the tool used for SoC design flow management. Automation will give an efficient way to analyse SoC design flow data, FrontEnd as well as BackEnd flow, generated from different EDA tools. PinPoint is a web based tool which will provide an efficient way to analyse design data synchronised with different EDA tools. PinPoint will capture complex information and will provide an easy way to find violations from design data. SoC Encounter, ICC, Olympus, SignOff and FrontEnd flow are currently aligned with PinPoint.

For different EDA tools, KPI matrices are defined on the basis of user specifications and project needs in which data is extracted from different files design files and reports. Each of these files and reports will be having different formats and our goal is to align PinPoint to extract data from design area. Data extracted from design area is presented on web based dashboard in interactive and creative way such that designer can analyse it and make modification in design accordingly to meet design constraints.

Contents

Declaration iii

Certificate iv

Acknowledgement v

List of Abbrevations vi

Abstract vii

List of Figures xi

1 Introduction 1

1.1 SoC Design 1

1.2 FrontEnd Flow 2

1.3 BackEnd FLow 3

1.4 ASIC Flow 4

2 Design Flow 6

2.1 Register Transfer Level (RTL) 8

2.2 Synthesis 8

2.2.1 Translation 9

2.2.2 Optimization 10

2.2.3 Mapping 10

2.3 Floorplan 11
2.4 Placement 12
2.5 Clock Tree Synthesis 13
2.6 Routing 14
2.7 Extraction 16
2.8 Verification 17
2.8.1 Formal Verification 17
2.8.2 Physical Verification 18
2.8.2.1 Design Rule Checks (DRC) 18
2.8.2.2 Layout v/s Schematic (LVS) 18
2.8.3 NETLIST CREATION 19
2.8.4 HANDOFF 20
2.8.5 PLACE and ROUTE 20
2.8.6 SIGNOFF 20

3 WorkFlow of Technical Dashboard 21
3.1 Problem Statement 21
3.1.1 Design Issues 22
3.1.2 Management Issues 22
3.2 Solution 22
3.3 WorkFlow of PinPoint 24

4 Dashboard For SoC Management 25
4.1 DASHBOARD 25
4.2 Key Performance Indicators 26
4.3 SnapShot Details 27
4.4 Result 28

5 Implementation and Results 29
5.1 Steps Integration 29

5.1.1 Placement and Routing 29
5.1.2 SignOff 30
5.1.3 FrontEnd 31
5.2 Results 31

6 Conclusion and Future Scope **32**
6.1 Conclusion 32
6.2 Future Work 32

List of Figures

1.1 FrontEnd FLow 3
1.2 BackEndFlow 3

2.1 RTL2GDSII flow 7
2.2 Synthesis Process 9
2.3 Placement 12
2.4 Clock Tree Synthesis 14
2.5 Routing Flow 15
2.6 Physical Verification Tasks 19

3.1 Design and Management Issue 22
3.2 Automation Overview 23
3.3 PinPoint Flow 23
3.4 PinPoint WorkFlow 24

4.1 PINPOINT Dashboard 26
4.2 Key Performance indicators 27
4.3 Layout Details 27

5.1 PnR Steps 30
5.2 Graphical Comparision 30
5.3 FrontEnd Steps 31

Chapter 1

Introduction

System-on-Chip (SoC) refers to the technology of integrating all components of an elec- tronic system into a single integrated circuit. Over the last several years, SoCs have become increasingly complex. This is primarily because of the increasing number of multimedia and real time applications supported by modern day SoCs. Due to these applications, the demand for high performance IPs and SoCs has increased. Consequently the number and types of IPs developed to execute these applications have increased over the years.

1.1 SoC Design

System on chip (SoC) refers to integrating all components of a computer or other electronic system into a single integrated circuit (chip). It may contain digital, analog, mixed-signal, and often radio-frequency functions all on a single chip substrate. With the development of VLSI, multi-million transistors can be embedded on a System on chip to meet increasing demands of more functionality and application support at a smaller chip cost with guaranteed reliability. Enhanced functionality in turn envisages an increase in density of components per sq mm and this can be achieved by moving to technology platforms like 65, 55 or 40 nm and even below. Diverse application support requirements demand the need to integrate complex

designs with a standard defined interface to ease SoC integration. This in turn requires a robust verification and a reliable testability scheme to guarantee reliable performance in the field even with varying environmental constraints.

In order to meet diverse SoC application requirements, it is not practical to design the whole system from scratch. Hence the approach is to integrate complex blocks that have been individually designed. These blocks are called intellectual property (IP) cores. An IP core is a reusable unit of logic, cell, or chip layout design and is the intellectual property of one party. IP cores may be licensed to another party or can also be owned and used by a single party alone.

For designing a particular SoC, besides the design, Specifications and architecture, we require the necessary IP cores. To meet the ever demanding needs of increased CPU performance and varying application requirements it becomes intuitive to build a diverse IP portfolio either through an in house development or by buying IPs from vendors which suit the network on chip protocol requirements as mandated by the SoC integration.

This design methodology where the basic components or IP cores are reused is called semi custom design. It reduces the design time and cost involved. In semi custom design it is possible to have a separated design approach for the front end (FE) dealing with logic and timing modeling of the design and back end (BE) dealing with the physical design.

1.2 FrontEnd Flow

Front-end means the design conducted by the ASIC designers, which usually includes HDL description; Simulation and Functional verification; Synthesis (net list). This step is finished by the ASIC sign-off point, so that the verified design will be delivered to the ASIC foundry.[4]

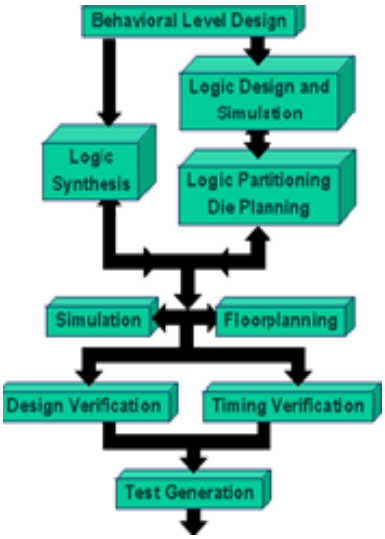

Figure 1.1: FrontEnd FLow

1.3 BackEnd FLow

Back-end tasks are usually includes Power Rail Design; Clock Tree generation; Congestion; Placement and Route; I/O and block Placement; Coupling; Hot spots; DRC. However, the overall timing and functional requirements are actually defined in the previous front-end phase so that the back-end phase will make sure after finishing back-end tasks, timing and functionalities will be the same.[4]

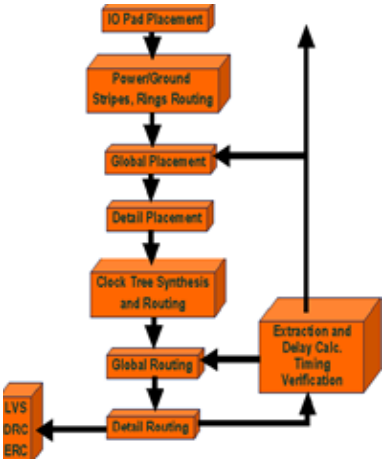

Figure 1.2: BackEndFlow

1.4 ASIC Flow

In integrated circuit design, RTL is a level of abstraction used in describing the operation of a synchronous digital circuit. In RTL design, a circuit's behavior is defined in terms of the flow of signals (or transfer of data) between hardware registers, and the logical operations performed on those signals.

The Logic Synthesis tool synthesizes RTL (register transfer level) code and converts it into a gate-level netlist, optimized for speed and size requirements defined by the designs constraints. The input to the synthesis tool is the RTL description of each digital design block (VHDL, Verilog), while the output is the gate level netlist describing the connection between gates. A Netlist/Gate-level netlist is the end result of the Synthesis process. This netlist contains information on the cells used, their interconnections, area used, and other details.[4]

1. Standard cell libraries :It describe the layout of basic combinational and sequential logic gates like Inverters , Buffers, ANDs/ ORs/ NANDs etc. Standard cell libraries are generally called reference libraries. These reference libraries are technology specific and are usually created and provided by an external ASIC vendor or internal library group. In addition to standard cell libraries, reference libraries also contain pad cell libraries for signal IO and power/ground pad cells, as well as macro cell or IP libraries for reusable IP like RAMs, ROMs, and other predesigned, standard, complex blocks.

2. Technology Libraries : It describe the structure, function, timing and environment of the ASIC Technology being used. The .libs is defined on various PVT (Process Voltage Temperature) conditions. This contains information about manufacturing grid, Metal layer creation and spacing rules, Pitch, Via rules, Metal thickness rules etc.

3. Synopsys Design Constraints : SDC is a format used to specify the design intent, including the timing, power and area constraints for a design. SDC is

TCL [Tool Command Language] based

4. The Design Exchange Format : The DEF file contains the netlist as well as the physical design data such as placement and bounding box data, routing grids, power grids, pre-routes and rows. DEF can also include additional information such as routing grids, power grids, pre-routes, and rows. This physical data is an optional input that would typically be available if this were a redesign of an existing device.

5. Graphic Design System : GDS or GDSII, is a database file format which is the de facto industry standard for data exchange of integrated circuit or IC layout artwork. It is the detailed physical view of all the design components used in the design. The data can be used to reconstruct all or part of the artwork to be used in sharing layouts, transferring artwork between different tools, or creating photo masks.

6. Library Exchange Format : Abstract physical views of all the design components used in the design. This information includes Library data in ASCII format, Wire unit parasitic information, Detailed cell size and pin location, Routing level information and Macro cell definitions

Chapter 2

Design Flow

Certification is the process of validating Hard and Soft IPs for CAD view compliance on various platforms (Synopsys, Cadence, Magma etc.). Design flow includes all the processes starting from RTL of the design and finishing at routing, verification and signoff checks to chip level design.

The basic certification flow generally called as RTL2GDSII flow consists of following steps:

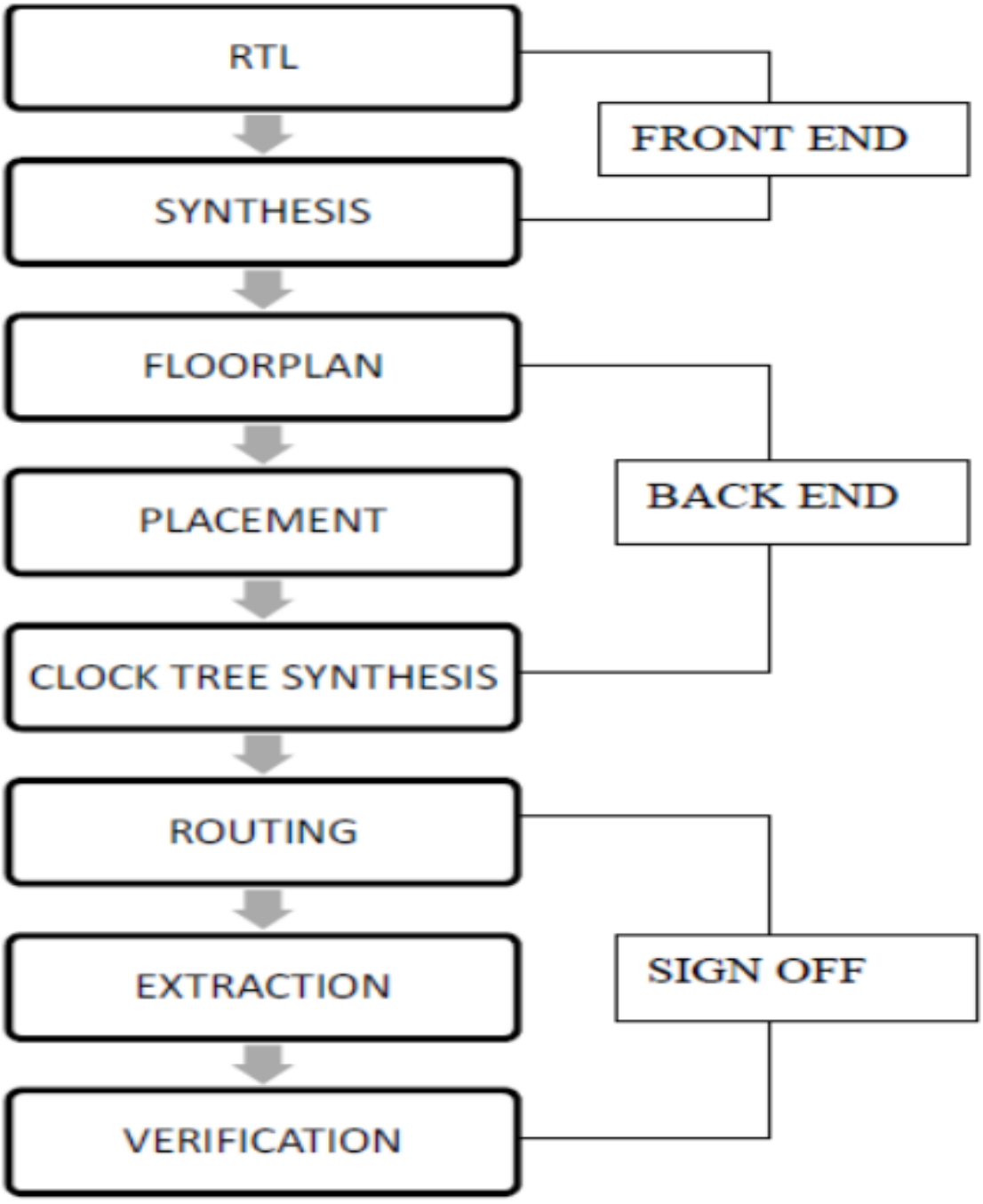

Figure 2.1: RTL2GDSII flow
[4]

2.1 Register Transfer Level (RTL)

In integrated circuit design, register transfer level (RTL) description is a way of describing the operation of a synchronous digital circuit. In RTL design, a circuit's behaviour is defined in terms of the flow of signals (or transfer of data) between hardware registers, and the logical operations performed on those signals. Register transfer level abstraction is used in hardware description languages (HDLs) like Verilog and VHDL to create high-level representations of a circuit, from which lower-level representations and ultimately actual wiring can be derived. RTL netlist is technology independent.[4]

2.2 Synthesis

Now the process starts at the design level. Synthesis is that part where we are provided with the constraints at the chip level. For example all the timing and area. Then according to that we have to write RTL (register transistor logic). After writing RTL we generate one netlist. That is called GDS (graphical design system). RTL is technology independent and netlist is technology dependent. That's why we generate the netlist. While synthesis the inputs thus given are RTL, Library of cells, Timing constraints. Logic synthesis is a process in which we design an optimized gate level representation of a hardware circuit. We use standard cells like AND, OR etc to form that gate level representation. Hardware circuits can't be designed directly with gates so first we need some process or way to get the expression which we have to design using gates.[4]

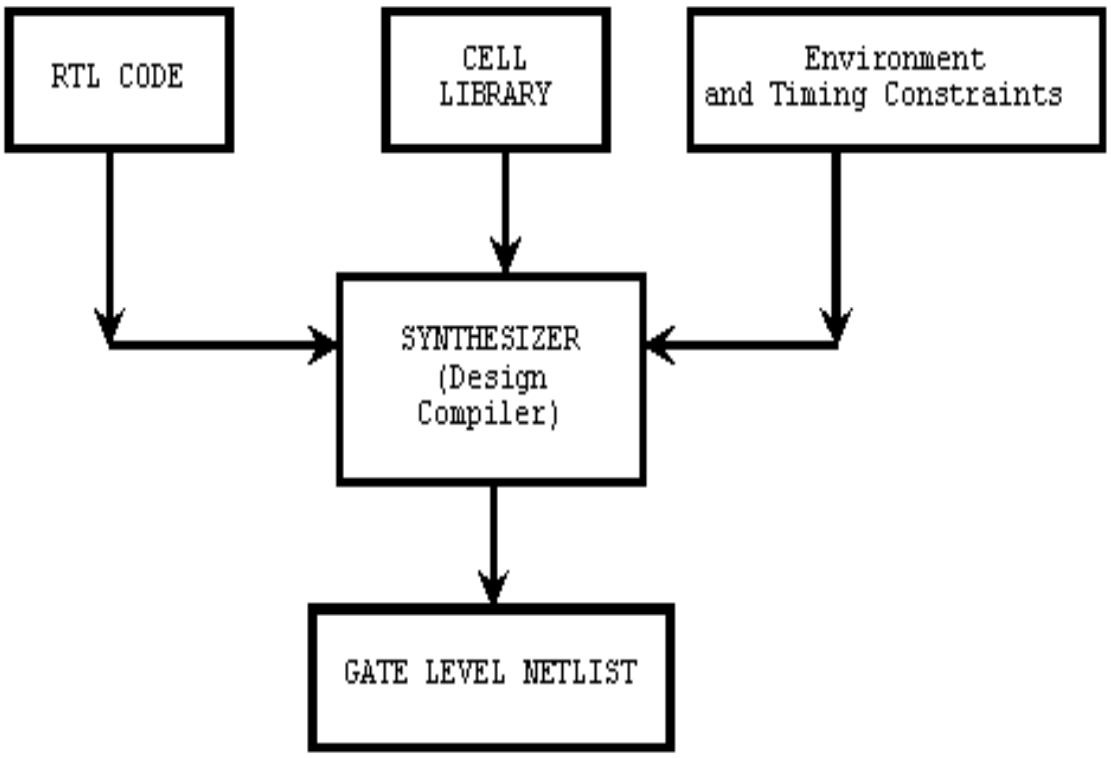

Figure 2.2: Synthesis Process
[4]

Synthesis process consists of 3 sub processes:

1. translation
2. optimization
3. mapping

2.2.1 Translation

So how we convert the analog circuit in the gate level expression. The first step is translation. In this process we use a hardware description language in which we write the architectural description of the analog circuit. And its output is a Boolean expression. Hardware description language can be like verilog or VHDL. We write the architectural description and constrains if specified in the verilog and based

on those it gives a Boolean expression that we have to design using gates. These constrains are timing, area etc. these constrains are very important and based on these the expression is modified.

2.2.2 Optimization

It is a process in which we try to reduce the Boolean expression that we get from the verilog by the techniques that we have. This optimization process is constraint driven. The constraints are the ones that are input to the synthesis tool. These constrains are timing, area etc.

2.2.3 Mapping

The next job that synthesis tool performs is to map the reduced Boolean expression interims of the standard cells that we have in the library. And the tool is designed in such a way that it maps the function in terms of cells that are available in the cell library. So after these three what we gate is netlist. It means the tool will give us output interims of cells which we have to use and in what number we have to use to realize the Boolean expression. From above we concluded that input to synthesis tool is the hardware description and the cell library and the output is the netlist.

Synthesis is done mainly because of the following three reasons:

1. Abstraction

2. Portability

3. Design tricks

Abstraction: Generally the design entry in the high level is easy, so that the focus is on higher level issues; synthesis enables design entry at higher level. All calculations of constraints like fan-out, load capacitance etc is not possible manually.

Because then we will have to calculate for each path separately and then find the largest one or the mean of all. With synthesis these calculations are easily performed by the tool.

Portability: As HDL is not technology dependent. So we can implement circuits for any library, i.e. if we are using tool then we don't have to do all the process again if we are using different technologies like 45 nm or 65 nm. But if we are performing manually then we have to calculate separately for every library which is a very hectic task.

Design Tricks: For design implementation we need to get the best possible combinations. The tool can implement a lot of optimization techniques and will take care of issues like logic reduction, loads, fanouts etc. Manually performing this job is not easy.

The output of the synthesis process is Gate level Netliist, which is used as input in the BackEnd Flow.

2.3 Floorplan

It is the first step in Backend flow, in Backend flow the design will be implemented physically using the logic representation, gate level netlist of the design. Floor planning is the exercise of arranging blocks of layout within a chip to minimize area or maximize speed. It can be called, allotment of the silicon real estate to different blocks in the design. The input to the floor planning is typically the dimension of the chip, the area constraints. In this step the PG connections will be given to the top level as well as for the standard cells and will be laid down with the help of straps. Positioning of analog blocks and hard macros are done manually in a given area of chip because they are sensitive to electromagnetic environment. Power grid management is done in the entire chip to supply power to the transistors except for hard blocks (e.g. DAC) around which rings is formed. IR drop analysis is done to make sure that power is supplied to every transistor.[4]

2.4 Placement

This step is meant to perform the placement. Placement is the task of placing Logic Cells or Gates in some pre-allocated rows to minimize area or cycle time. Cells are placed adjacent to each other so as to minimize the interconnections. Placement reads the PDEF, for placing the modules. Placement considers the modules that were placed during floorplanning and takes into account the hierarchy and connectivity of the design. It honors floorplanning constraints, including guides, regions, and fences. It places standard cell instances at legal locations where there should not be any DRC violations against preroutes or routing blockages. If we have macros in our design like PLL, ADC, DAC or IOs we have to first place them only after that we place the logic cells. The placing of these macros is more often done in a manual fashion. Manual placement is also possible for logic cells if we want say some of the cells to be assigned some predefined locations.[4]

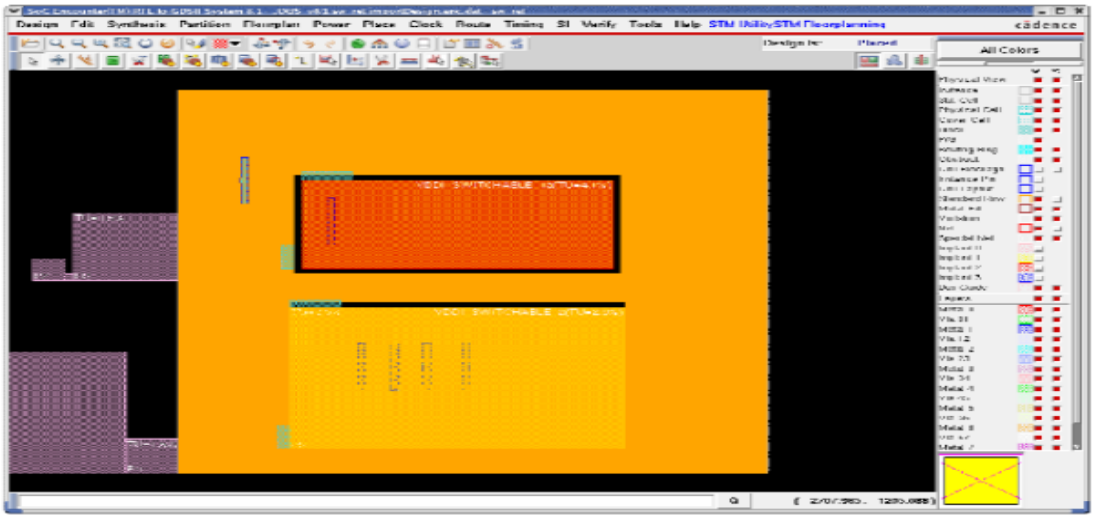

Figure 2.3: Placement
[4]

2.5 Clock Tree Synthesis

The main objective of Clock Tree Synthesis is to avoid the clock skew, which would create synchronization failure between different entities in the design. Method constructs a tree of clock buffers with some suitable geometry such that modules that communicate with each other receive well-defined clocks. Gated CTS processes the active gated clock net from root to the clock pin. Given the root net Gated CTS traverses through all the gated subnets, determines the structure of buffer trees and performs CTS to optimize skew. Need clock constraints (min and max insertion delay, max transition, max skew, Buffers and Inverters to be inserted), as an input to CTS. Clock tree synthesis ensures that clock signal must reach all sequential elements in the same clock tree at the same time by inserting clock buffers. Clock buffers are usually bigger in size and have shorter transition time as well as more rise and fall times. Clock nets are generally routed first and on higher metal layers with minimum detouring to give it the highest priority in routing.[7]

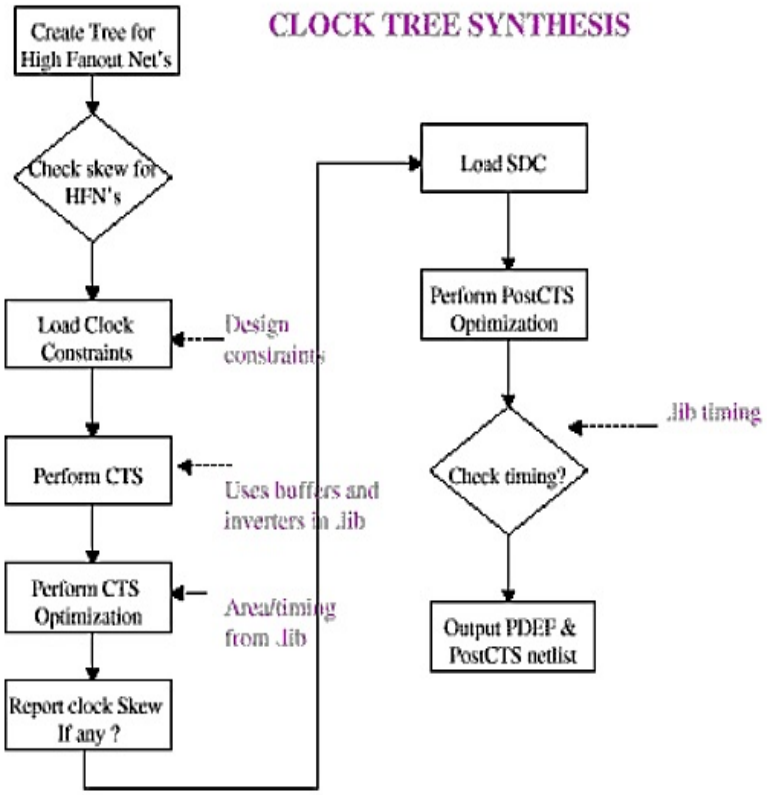

Figure 2.4: Clock Tree Synthesis
[7]

2.6 Routing

This step is meant to perform the full routing of the design and physically implement the design connectivity (between gates, I/O, blocks) using the given metal layers and vias. During routing it is important to make sure that the given constraints (i.e. timing, area, power etc) of the design are met after physically connecting the cells. Different products use different types of routing, they have different routers and algorithms achieve the above.[4]

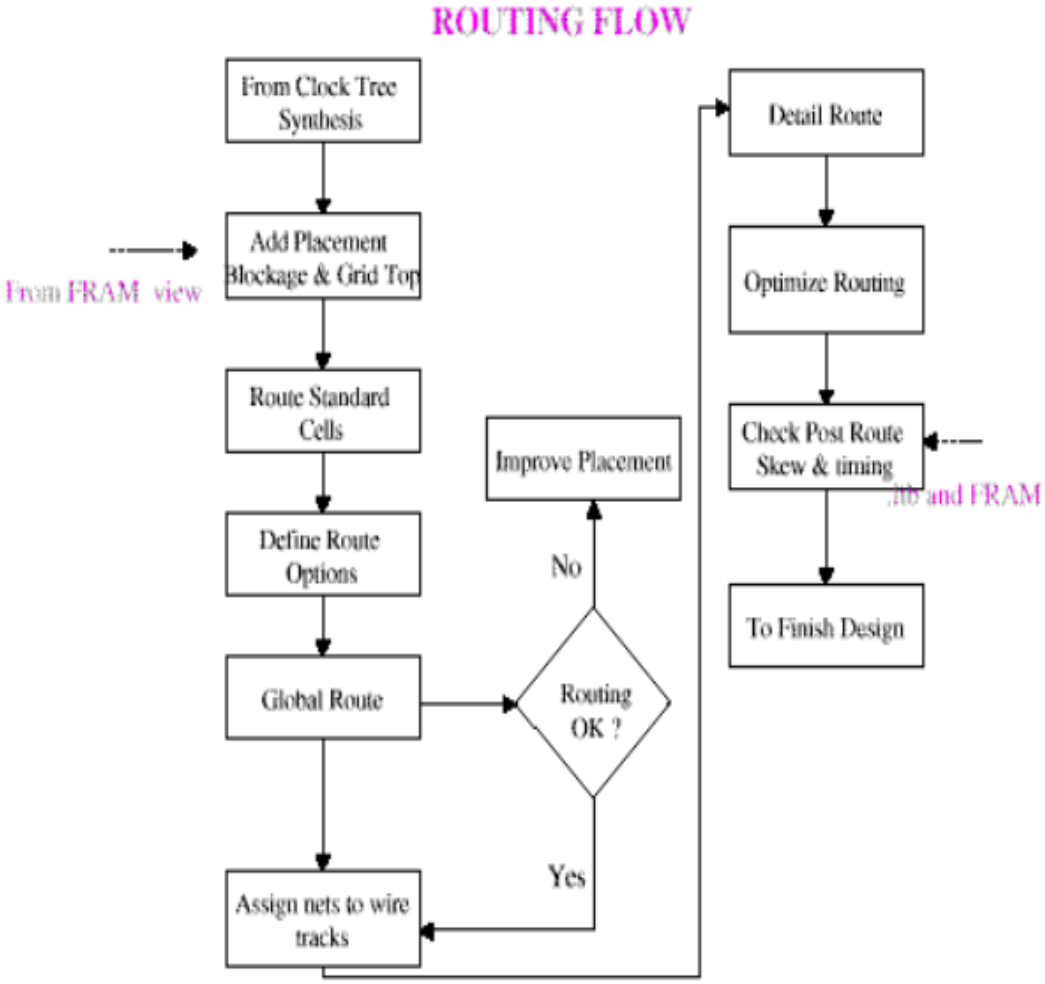

Figure 2.5: Routing Flow
[4]

Full routing consists of global and detailed routing.

1. During global routing phase, the router breaks the routing portion of the design into rectangles called global routing cells (gcells) and assigns the signal nets to the gcells. The global router attempts to find the shortest path through the gcells, but does not make actual connections or assign nets to specific tracks within the gcells. The Global routing is part of the block implementation and

the top-level implementation stages of the Encounter flow. We should run the router early in the design flow to identify and fix routability problems or avoid them altogether

2. During Detailed routing phase, the NanoRoute router follows the global routing plan and lays down actual wires that connect the pins to their corresponding nets. The detailed router creates shorts or spacing violations rather than leave unconnected nets. The router runs search-and-repair routing during detailed routing. During search and repair,it locates shorts and spacing violations and reroutes the affected areas to eliminate as many of the violations as possible. The primary goal of detailed routing is to complete all of therequired interconnect without leaving shorts or spacing violations.During detailed routing, the router divides the chip into areas called switch boxes (SBoxes), which align with the gcell boundaries. The SBoxes can be expressed in terms of gcells; The SBoxes overlap with each other and their size and amount of overlap might vary during search-and-repair iterations.

2.7 Extraction

After the full flow is performed the signoff is performed. Signoff is defined as the process of passing design to an ASIC vendor who guarantees a working Silicon. By signing off, LVS and DRC checks, performing parasitic extraction, timing and cross talk analysis are done. This is to confirm that after clock generation and routing no error occurs. And according to the errors the changes are made.

The process of signoff is performed after the place and route flow. The output files generated during the place and route flow serve as inputs to the signoff flow. These input files include the binary format layout file gdscad (GDS-II) generated during finish design, the DEF and LEF files generated during the routing phase.

Extraction involves extracting accurate information from routed design to check the correctness of implemented design (i.e. placement and routing). Extraction does

the following tasks:[5]

1. Produces accurate, full-chip extraction for noise, voltage (IR) drop, and timing verification.

2. Does selective extraction and netlisting for critical path analysis.

3. Provides a complete, production-proven solution for different design types, such as ASIC, full custom, microprocessor, memory, and analog designs.

2.8 Verification

[6] The Verification applications assist certification engineer in verifying the physical and electrical integrity of IC designs.

Generally verification is of two types: Formal verification and Physical verification.[6]

2.8.1 Formal Verification

This task checks the formal equivalence between initial verilog and final verilog. Here the synthesized netlist is compared with the RTL netlist and the routed netlist is compared with the mapped netlist. This is done to check if the RTL netlist and routed netlist are equivalent. However this method only verifies the functionality of the design.

Formal verification is an alternative to verification through simulation. Verification through simulation applies a large number of input vectors to the circuit and then compares the resulting output vectors to expected values. Formal verification utilizes mathematical techniques to compare the logic to be verified against either a logical specification or a reference design.

2.8.2 Physical Verification

Physical verification checks the physical implementation of the design i.e. layout or gds. The two main tasks performed in the physical verification are DRC and LVS.

2.8.2.1 Design Rule Checks (DRC)

Design Rule Checks (DRC) is the area of Electronic Design Automation that determines whether a particular chip layout satisfies a series of recommended parameters called Design Rules. A design rule set specifies certain geometric and connectivity restrictions to ensure sufficient margins to account for variability in semiconductor manufacturing processes, so as to ensure that most of the parts work correctly.

2.8.2.2 Layout v/s Schematic (LVS)

Layout v/s Schematic (LVS) checking software recognizes the drawn shapes of the layout that represent the electrical components of the circuit, as well as the connections between them. The software then compares them with the schematic or circuit diagram.

The inputs to DRC are an SVRF rule file and a layout database. The outputs are a run transcript and a DRC Results Database, with a DRC Summary Report.

The inputs to LVS are an SVRF rule file, a layout netlist or database, and a source netlist or database. The LVS outputs that always occur include a run transcript and an LVS Report. The process of verification is carried out using the Mentor Graphics Calibre tool.

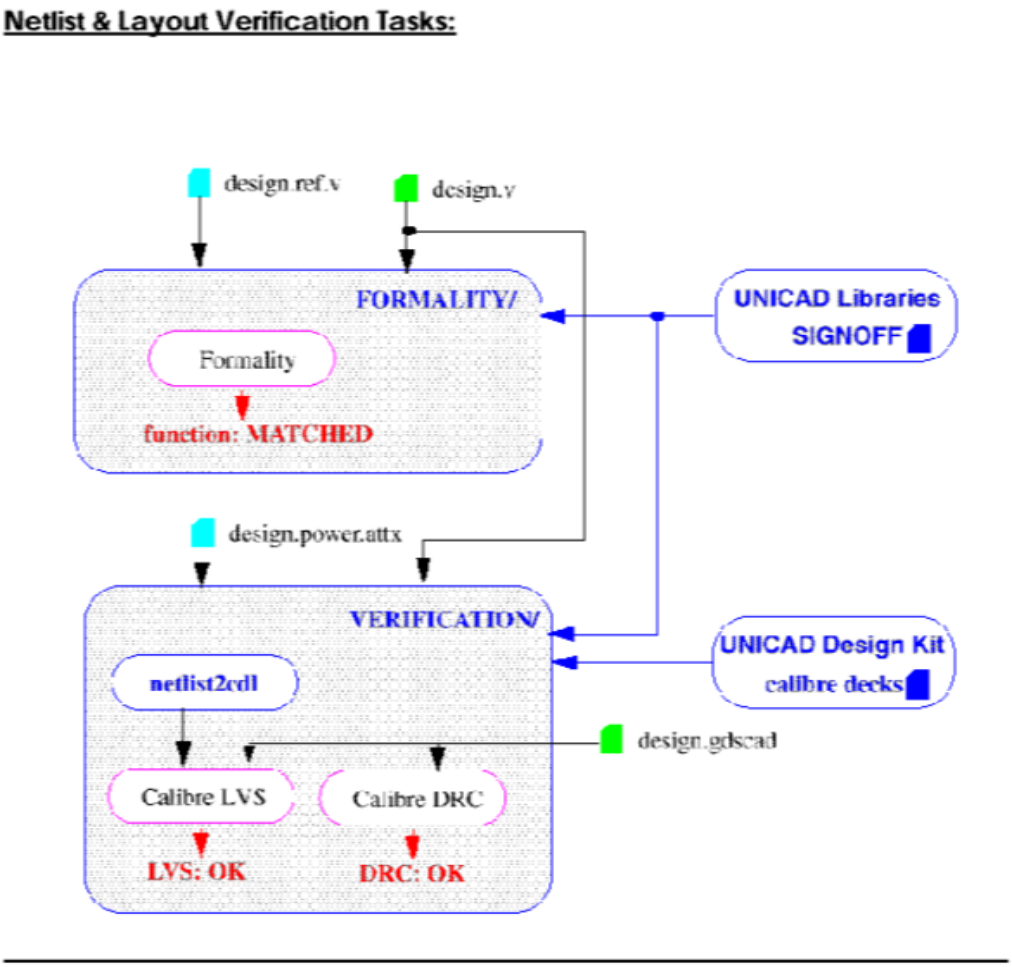

Figure 2.6: Physical Verification Tasks
[?]

2.8.3 NETLIST CREATION

1. FUSION : In-house developed tool to generate RTL

2. Design Compiler : Used for synthesis of RTL

2.8.4 HANDOFF

1. Synopsys Handoff Kit : Kit to control the automatic execution of Handoff checks.

2. Prime Time : To perform timing analysis and constraint checks on the netlist and constraints.

2.8.5 PLACE and ROUTE

1. Encounter Kit, ICC Kit, Sierra Kit : Kits to control the automatic execution of Placement And Routing

2. EncounterTechnoKit, SynopsysTechnoKit, SierraTechnoKit : Kits containing technology related information in the format required by PnR tools.

3. Encounter, ICC, Sierra : PnR tool from Cadence, Synopsys, Mentor respectively.

2.8.6 SIGNOFF

1. SignOff Kit : Kit to control the automatic execution of signoff checks

2. SignoffTechnoKit : Kit containing technology related information required by different Signoff tools

3. Calibre : Used for DRC and LVS

4. StarRCXT : Used for parasitic extraction

5. PrimeTime : For Delay calculation and timing analysis

Chapter 3

WorkFlow of Technical Dashboard

During SoC design cycle, there is need to communicate design progress across team. Also, ability to view historical data in order to gain insight about the big hairy issues in the design is of utmost importance. As design data is huge, its a tedious task to search relevant info in huge design data. Thus there is a need to rationalize and automate process in order to have better data management infrastructure.

A 24*7 web-based automated dashboard is an answer to this problem. It has standardized extraction process as well as monitoring process. Extracts relevant KPIs automatically makes it available on the web dashboard. This helps designers save time spent in

3.1 Problem Statement

Precise Statistics are key to our RnD effectiveness. Lot of energy spent in collecting Statistical Data over time. Need to Rationalize and Automate such process to save time and energy. The SoC Management issues can be broadly categorized in two sub-categories[3]

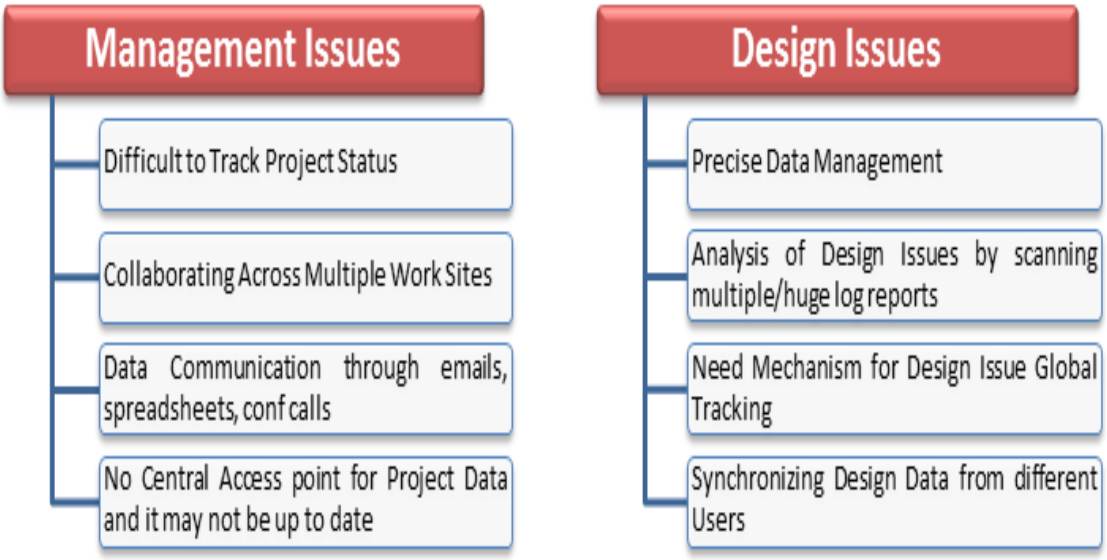

Figure 3.1: Design and Management Issue

3.1.1 Design Issues

For SoC design flow, it is very difficult for the team to synchronize design data and take critical decisions. Analysis of design is also a difficult task when data is coming from different users. Also global issue tracking is key requirement to make SoC design flow productive.[3]

3.1.2 Management Issues

Projects running on different sites are needed to synchronized in terms of complex data management. Medium for such communication is important and data provided may not be up to date.

3.2 Solution

Keeping in view issues as discussed above, there is a need to move to a platform which can perform the automatic extraction of meaningful information from huge

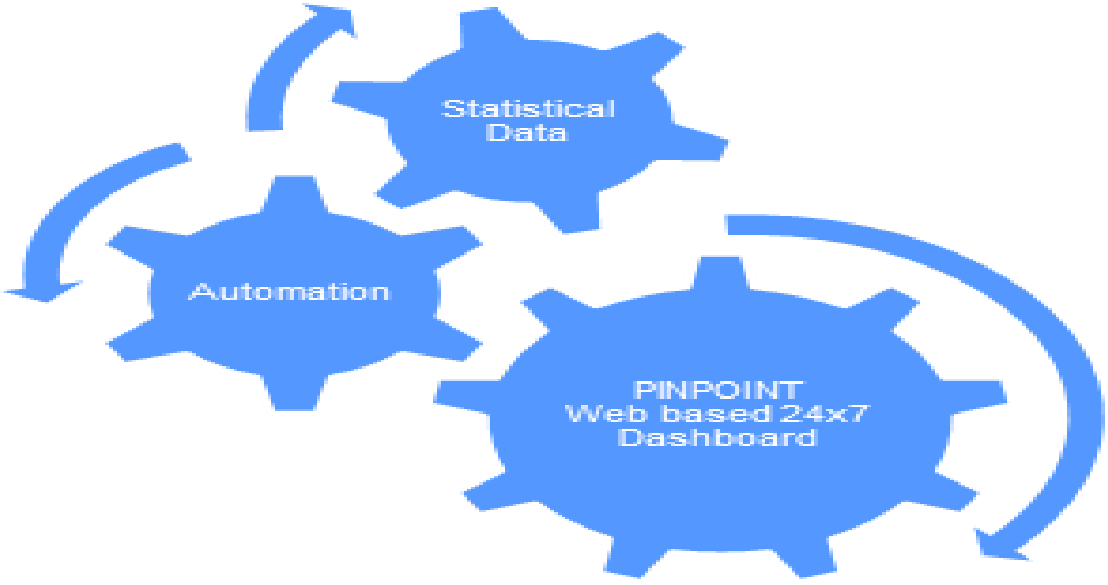

Figure 3.2: Automation Overview

set of data and make it available instantly to users and managers for quick analysis and decision making. We decided to develop a ST Kit around PinPoint (3rd Party Software) which can help us achieve our goals as could be depicted below.

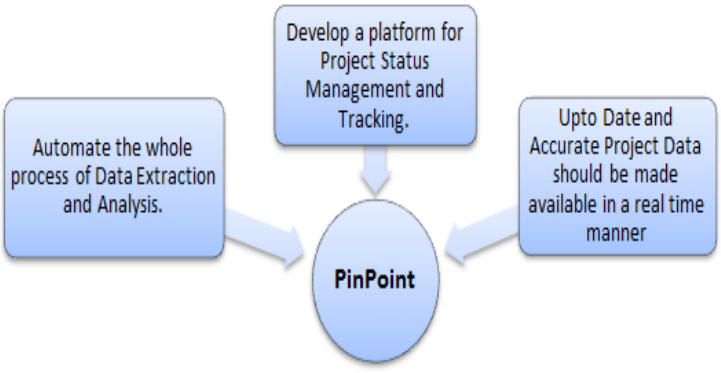

Figure 3.3: PinPoint Flow

3.3 WorkFlow of PinPoint

PinPoint is a three level dashboard which are Project, Block and SnapShot level , arranging reports and data in specific format. PinPoint extracts reports, logs, etc. from the user area and upload it to this three level dashboard which is web based 24x7 available. Currently it supports Encounter, ICC, Olympus, SignOff, Frontend. Dashboard is customizable as per project needs and KPI can be added accordingly and it extracts timing, clock tree, path group summary etc.

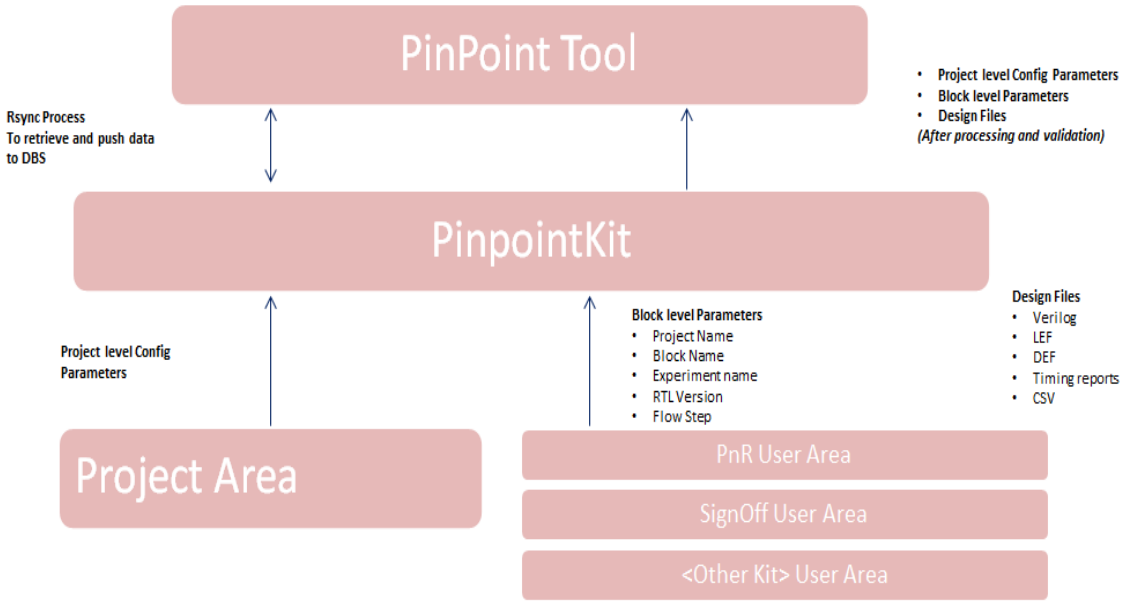

Figure 3.4: PinPoint WorkFlow

It also allows to compare KPI across different flow steps which will give better visibility regarding modification results. Also it provides graphical feature to compare results across different flow steps in the design flow. That will provide better discernibility to work on violations and changes to be made in the design. Layout details is also available with bucketing feature through which negative slack can be seen directly on dashboard.

Chapter 4

Dashboard For SoC Management

The main aim of the work is to provide an easy and efficient way to analyse the data that are generated from different tools in different files. This is achieved by web based dashboard that will extract the data from files generated by different tools.

4.1 DASHBOARD

Dashboard is designed as Decision support system (DSS), a system of supporting decision-maker to solve semi-structured and non-structured problems using data and model. Based on EDA tools, PINPOINT is designed which will support each step of design flow. Cadence, Mentor, ICC and Signoff are the vendors for which PINPOINT dashboard is available. Based on different blocks snapshopts for each step is taken from the system and data is available on dashboard with few minutes.[8]

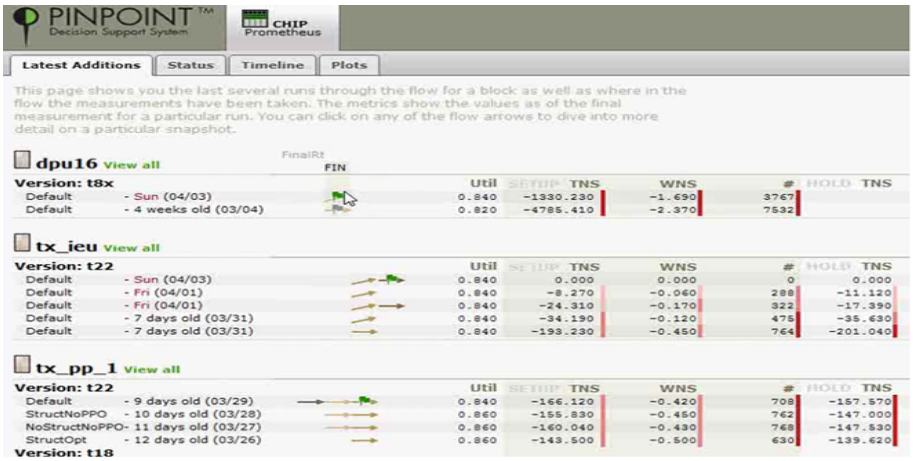

Figure 4.1: PINPOINT Dashboard

4.2 Key Performance Indicators

There are some parameters which are important from user point of view, such parametrs are shown on the dashboard. For PinPoint we can define each KPI in terms of steps of design flow.For example for PnR we are taking different snapshots for each 5 steps ie PreCTS, PostCTS, Route,PostRoute and PostIncrRoute. For each of these steps we are taking snapshots and checking the results and changes that are to be made for optimized performance. Similarly for SIGNOFF, data is extracted from global timing reports generated from primetime. PINPOINT also give flexibility for reading lef as well as def file which will provide user to view the path and blocks on Pinpoint itself. They provide the physical geometry of the block.[8]

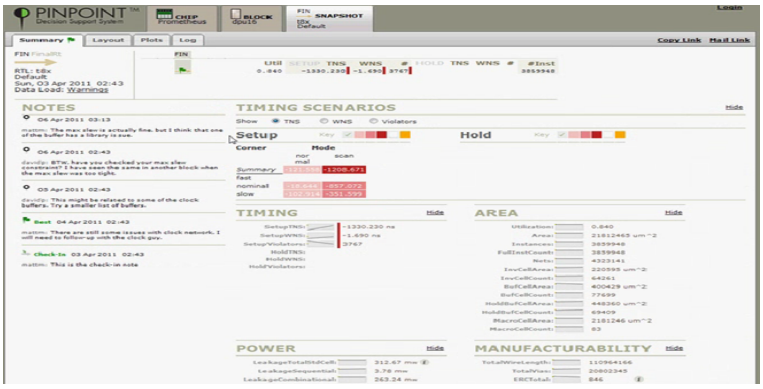

Figure 4.2: Key Performance indicators

4.3 SnapShot Details

Snapshots are taken of each steps of design flow cycle. If the block is on placement and routing stage, the five different snapshot is to be taken which are prects , postcts, route, postroute and postincroute. Each of these snapshots will tell the detail modification and changes made to reduce the violators in the block. For Timing Analysis,snapshot after SignOff stage is taken which will give general details about total negative slack and worst negative slack.[8]

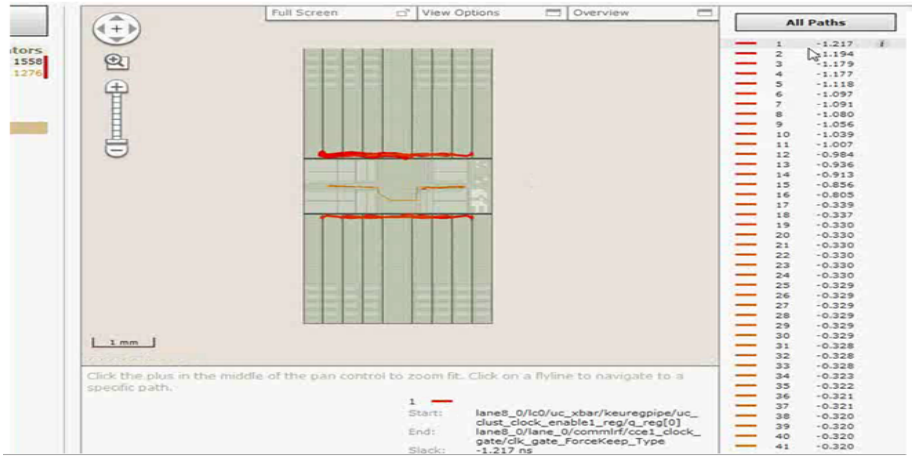

Figure 4.3: Layout Details

4.4 Result

PinPoint was deployed in various projects and was found very productive in terms of tracking and data management. Designers can easily track their data directly from dashboard without searching in huge area and can directly conclude on the problems. Also considering project leads, it will be easy for them to have a control over flow of the project and can be productive in terms of synchronization. Figure shows the dashboard which was developed for the boreal with design sync integration.

Chapter 5

Implementation and Results

Integrated Product Life Cycle management ties together product information related to critical processes and tasks through the development lifecycle, and provides visibility into potential problem areas that can be addressed early in the development process.

5.1 Steps Integration

PinPoint is integrated with BackEnd and FrontEnd flow of SoC design. Following are some examples showing the design step wise integration of PinPoint.[6]

5.1.1 Placement and Routing

Place and route is a stage in the design of printed circuit boards, integrated circuits, and field-programmable gate arrays. As implied by the name, it is composed of two steps, placement and routing. The first step, placement, involves deciding where to place all electronic components, circuitry, and logic elements in a generally limited amount of space. This is followed by routing, which decides the exact design of all the wires needed to connect the placed components.[7] PinPoint is aligned with almost all PnR tools available in the market. Including Soc Encounter, ICC and

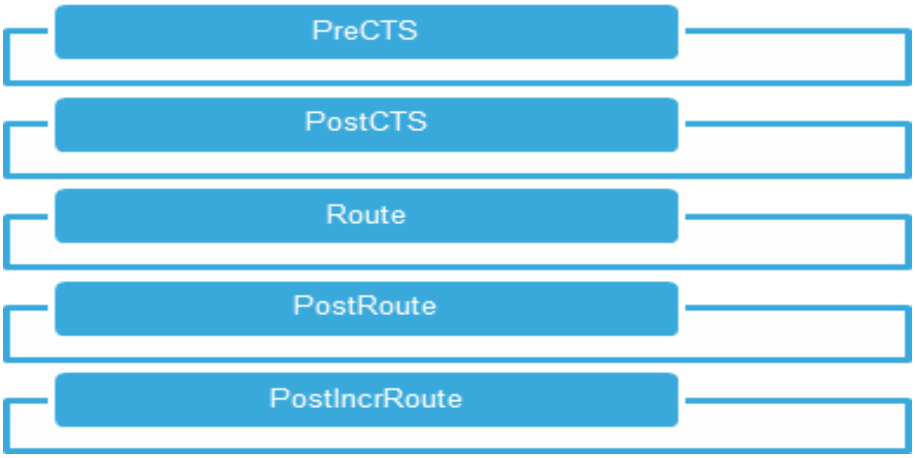

Figure 5.1: PnR Steps

Olympus. It is having a very good path bucketing feature which will help designers to seach for violated path from design. It also provides graphical analysis tool for comparing TNS and WNS over the snapshot

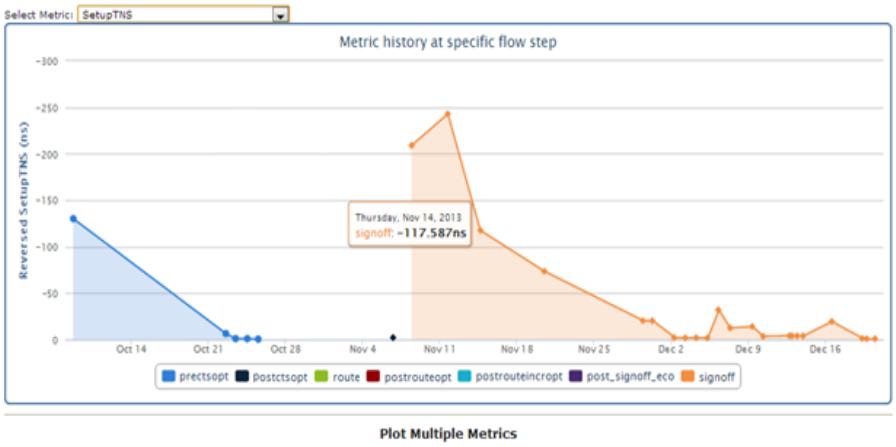

Figure 5.2: Graphical Comparision

5.1.2 SignOff

Static Timing Analysis is a method of computing the expected timing of a digital circuit without requiring simulation. STA plays a vital role in facilitating the fast

and reasonably accurate measurement of circuit timing. PinPoint provides very good timing analysis feature by which one can easily analyze and notify violations through timing report.[6]

5.1.3 FrontEnd

In the new release of PinPointKit, It provide support for FE Kit, extracting data directly from the CSV files. Below diagram shows flow steps added with PinPoint and new dashboard for FrontEnd Flow was designed.[6]

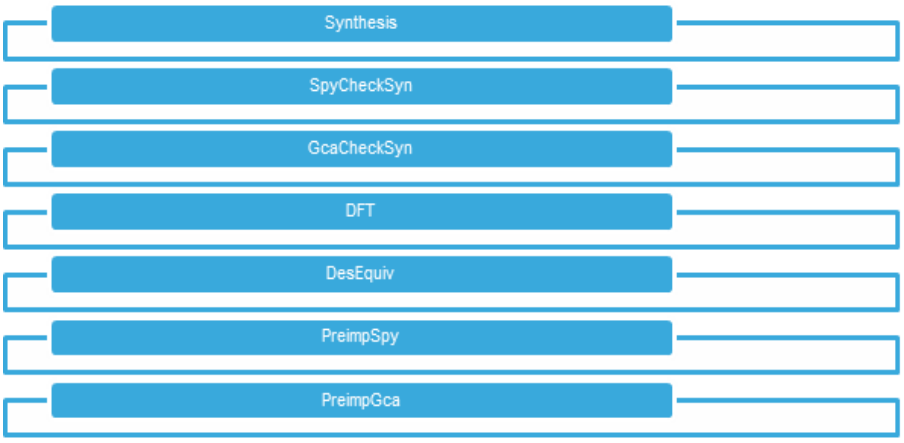

Figure 5.3: FrontEnd Steps

5.2 Results

By this methodology, a milestone is achieved through greater performance at the extended enterprise level, involving information management, program management and collaboration across separate groups and companies throughout the semiconductor development chain. It helps provide a single version of the truth for product information, enabling better process discipline and more efficient execution.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

This method provide an efficient way to track project results and to find issues and violations of design. It will provide an efficient approach to analyse design and make it more productive. Design and management approach from project tracking is improved and optimized to have better SoC product lifecycle management.

6.2 Future Work

Support for Power Reports from various tools and the concept of Generic Metrics will be considered in the future scope of this tool. PinPoint provides huge flexibility in terms to designing dashboard as the projects and designers requirement. Drag and Drop feature in KPI monitoring and flow steps control will be considered as future development.

References

[1] ST internal documents and Training manuals on LIBRARY BASICS

[2] ST internal documents and Training manuals on LIBRARY VIEWS

[3] ST internal documents and Training manuals on DESIGN FLOW INTRODUCTION

[4] ST internal documents and Training manuals on RTL TO GDS FLOW

[5] ST internal documents and Training manuals on PRIMETIME WORKSHOP

[6] ST internal documents and Training manuals on STATIC TIMING ANALYSIS BASICS

[7] ST internal documents and Training manuals on CLOCK TREE MANAGEMENT

[8] ST internal Tools

[9] Practical Programming in Tcl and Tk (4th Edition) by Brent B. Welch

[10] Tcl/Tk Cookbook by L. Sastry