"Automated Flow to Manage Dependencies between Different Flows – Serial & Parallel Execution of Design Steps"

A Major Report

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING
(VLSI Design)

Submitted By

Hitesh Dadlani
(19MECV06)

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

May 2021

CERTIFICATE

This is to certify that the project entitled “Automated Flow to Manage Dependencies Between Different Flows – Serial & Parallel Execution of Design Steps” submitted by Hitesh Dadlani (19MECV06), towards the partial fulfillment of the requirements for the degree of Master of Technology in VLSI Design, Nirma University, Ahmedabad is the record of work carried out by him under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.

Date

Dr. Vaishali Dhare	Dr. Usha Mehta
Internal Project Guide	PG Coordinator M.Tech - VLSI
Institute of Technology,	Institute of Technology,
Nirma University, Ahmedabad	Nirma University, Ahmedabad

Dr. Dhaval Pujara	Dr. Rajesh N Patel
Head of Department	Director
Department of Electronics & Communication Engineering	School of Engineering
Institute of Technology,	Institute of Technology,
Nirma University, Ahmedabad	Nirma University, Ahmedabad

Statement of Originality

I, Hitesh Dadlani, Roll No: 19MECV06, give undertaking that the M.Tech thesis entitled "Automated Flow To Manage Dependencies Between Different Flows – Serial & Parallel Execution Of Design Steps" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in Electronics & Communication Engineering (VLSI Design) of Institute of Technology, Nirma University, Ahmedabad, contains no material that has been awarded for any degree or diploma in any university or school in any territory to the best of my knowledge. It is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. It contains no material that is previously published or written, except where reference has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

Signature of Student

Date:

Place: Ahmedabad

Endorsed by

Dr . Vaishali Dhare

(Signature of Guide)

Acknowledgments

It gives me immense pleasure in expressing thanks and profound gratitude to Dr. Vaishali Dhare, Electronics and Communication Department, Institute of Technology, Nirma University, Ahmedabad for her valuable guidance and continual encouragement throughout this work. The appreciation and continual support she has imparted has been a great motivation to me in reaching a higher goal. Her guidance has triggered and nourished my intellectual maturity that I will benefit from, for a long time to come and also for his guidance throughout whole M.Tech.

It gives me an great pleasure to thank Dr. Usha Mehta, PG Coordinator of VLSI Design, Electronics and Communication Department, Institute of Technology, Nirma University, Ahmedabad for her kind support and guidance throughout whole M.Tech.

It gives me an immense pleasure to thank Dr. Dhaval Pujara, Hon'ble Head of Electronics and Communication Department, Institute of Technology, Nirma University, Ahmedabad for his kind support and providing basic infrastructure and healthy research environment.

I would like to thank Mr. Jebin Vijai (Engineering Manager) and Mr. Sayan Banerjee (CAD Engineer) at Intel Technology India Private Limited, Bangalore, for their suggestions and the clarity of the concepts on the topic that helped me a lot in this work. Also I am thankful to my team mates for their valuable guidance and prompt help.

I would also thank the Institution, all faculty members of Electronics and Communication Engineering Department, Nirma University, Ahmedabad for their special attention and suggestions towards the project work.

Hitesh Dadlani
19MECV06

ABSTRACT

The EDA industry has evolved rapidly over the last two decades. Designer's productivity has raised due to improved EDA tools over the last quarter of the century. Moore's law was made more effective and efficient with help of EDA tools. EDA tools have become essential in the design of chips with high complexity.

The first integrated chips were small-scale integration chips where the gate count was very small. With the advancement of technology, designers were able to place circuits with thousands of gates on a chip. As the number of gate counts increased design process started getting complex, designers felt the need to automate the process. Developers moved to the development of EDA tools in the design flow. The development of EDA tools in design flow depends on four major interrelated principles. The four principles on which EDA tools are developed are modularity, integrity, operate incrementally, have less runtime.

This Project introduces automated flow for front-end tools which eased the process of front-end tools execution. In the front-end domain, there are nearly forty tool flows. This project covers the various static flows used in the front-end domain. The automated flow will manage dependencies between these flows for serial & parallel execution depending on the input file. The automated flow comes with many features that can reduce the execution time and engineering efforts. The features include the dependency stage, different modes of execution, simple input format for the project, and tool configurations. The developed automated flow supports the standard environment used for INTEL projects. The automated flow is developed using PERL. Regression Testing is done to ensure the tool meets the required quality goals.

Contents

List of Figures

NOMENCLATURE

Abbreviation

RTL	Register Transfer Level
DRC	Design Rule check
LVS	Layout Rule Check
CDC	Clock Domain Crossing
ASIC	Application-Specific Integrated Circuit
VLSI	Very Large Scale Integration
CAD	Computer-Aided Design
EDA	Electronic Design Automation
PERL	Practical Extraction and Report Language

Chapter 1

Introduction

Electronic Design Automation (EDA) plays a major role in improving VLSI chip design. These complex chips are part of our day-to-day life hence EDA tools helps in improving the human life. Electronic system is modeled at the different abstraction level.   The electronic design automation tools work across all abstraction levels such as ESL, RTL, gate level, switch level, or physical level. EDA helps us to automate the VLSI design system by verifying correctness of design against specification at all the abstraction level. EDA takes the ESL plan through different stages such as synthesis, verification steps and testing the fabricated electronic system to guarantee that it meets the required quality goals.

EDA consist of set of hardware and software co-design, synthesis, verification tools that check the electronic system level design. EDA tools are used to translate the the behavior description to register transfer level (RTL) and then takes register transfer level through verification and synthesis stages at gate level and switch level. After synthesis we get a gate level netlist, which shows the interconnection of various components used in our design and eventually produces physical design (GDSII) format in that last stage of the tapeout, the engineer performs wafer processing, packaging, testing, verification and delivery to the physical IC. GDSII is the file produced and used by the semiconductor foundries to fabricate the silicon and handled to client.

1.1 VLSI ASIC FLOW

Today, ASIC design flow is a very complex and mature process in silicon design. The ASIC design flow and its different stages in VLSI designing that we describe are based on accepted methods and best practices in ASIC chip plans. This part attempts to explain various stages in the ASIC design flow, beginning fromASIC plan idea and moving from requirements(specification) to benefits.

1) **Specifications:-** This is the first stage in the design process where we define the important parameters of the system that must be designed into specification.

   Specifications details can be as follows

- Goals and requirements of the design.
- Functionality of chip
- Performance figures like Area and power

- Technology requirements like size and space

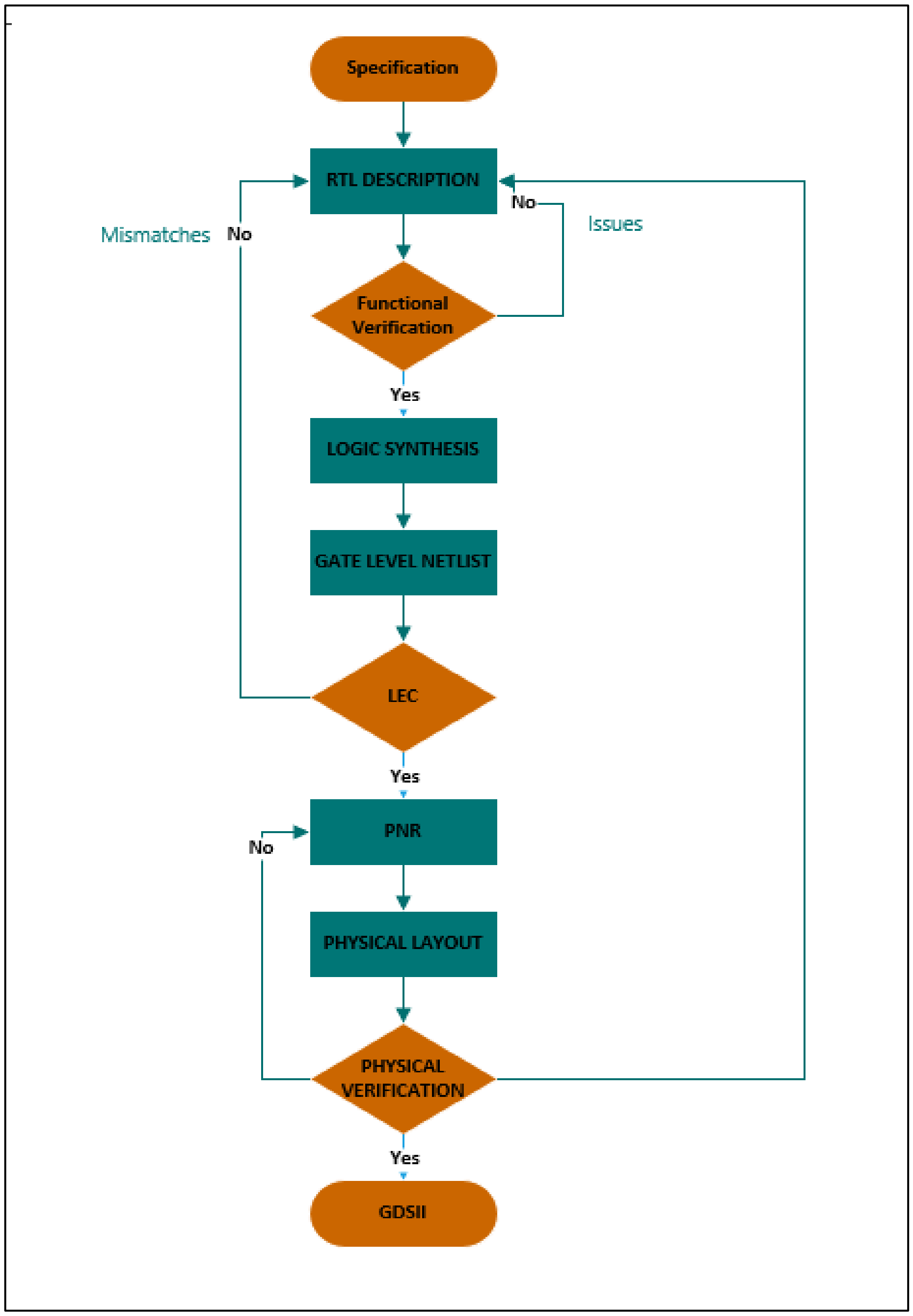


Fig 1: VLSI ASIC FLOW

2) **RTL DESCRIPTION:** Given the specification of the design, we need to describe a design and make the RTL code in one of the Hardware Description Language (HDL) either Verilog or VHDL. These two HDLs are utilized most generally. When the RTL code and testbench are produced, the RTL group deals with RTL description – they translate the RTL code into a gate-level netlist utilizing a synthesis tool that meets required timing constraints.

3) **FUNCTIONAL VERIFICATION:-** This step confirms the functionality of design by applying the different stimulus to the design check the output.It is step where the design group and RTL group come into the cycle where they produce testbench in SVA. This is called as behavior Simulation.

   There are two kinds of tools for simulation:

   a) **Functional Simulation Tools**: After functionally verify the design it will tell us the logic we implemented is matching our specification.

   b) **Timing Simulation Tool**: Verifies that circuit configuration meets the timing constraints prerequisites and affirms the design is liberated from circuit signal delay.

      The tools used for performing this steps are SYNOPSYS VCS, MENTOR GRAPHICS QUESTASIM

4) **LOGIC SYNTHESIS:** Upto the stage 3 technology was not included however from logic synthesis step, there is a dependence of technology that we are utilizing. The logic synthesis tool translate your Verilog/VHDL code into a Gate level netlist.

   The instruments utilized for this progression are -SYNOPSYS DESIGN COMPILER, CADENCE-GENUS, MENTOR GRAPHICS-LEONARDO SPECTRUM.

5) **GATE LEVEL NETLIST:** After Logic synthesis we get an gate level netlist, which shows the interconnection of different segments utilized in our design.

6) **LOGIC EQUIVALENCE CHECK(LEC):** LEC is the process of equivalence check among pre and post plan design. This equivalence checks if your design is logically right. If its not , at that point the engineer needs to again do the logic synthesis and make the design logically right, else we proceed onward to physical design.

The devices utilized in this progression are SYNOPSYS-FORMALITY, CADENCE-CONFORMAL, MENTOR GRAPHICS-QUESTASLEC.

7) **PLACE & ROUTE:** Placement is the way toward placing standard cells in row. A bad placement situation requires larger area and furthermore degardes the performance. Different variables, similar to the timing requirements, the net lengths and thus the connections of cells, leakage power should be taken consideration. It eliminates timing voilation.

   Routing is essentially the interconnection of the wires. Effective routing is must for the design, since it diminishes the voilation as well as improves the signal delay.

   The tools required for performing this step are CADENCE-INNOVUS, SYNOPSYS-IC COMPILER,MENTOR GRAPHICS OLYMPUS.

8) **PHYSICAL LAYOUT:** After Placement and Routing we will have the physical layout of our plan. We need to do actual physical verfication of our design. We have to do physical verification of our layout, which is what discussed in the next step in detail.

9) **PHYSICAL VERIFICATION:** After routing, ASIC plan goes through three stages of verification, known as signoff checks. This stage assists with checking whether the layout working the manner in which it was intended to. The accompanying checks are followed to avoid any mistakes not long before the tape out:

   a) Layout versus schematic(LVS) is a cycle of checking that the geometry/layout coordinates the schematic/netlist.

   b) Design rule checks(DRC) is the way toward matching that the calculation in the GDS file adheres to the rules given by the foundry.

1.2 Front End Flow

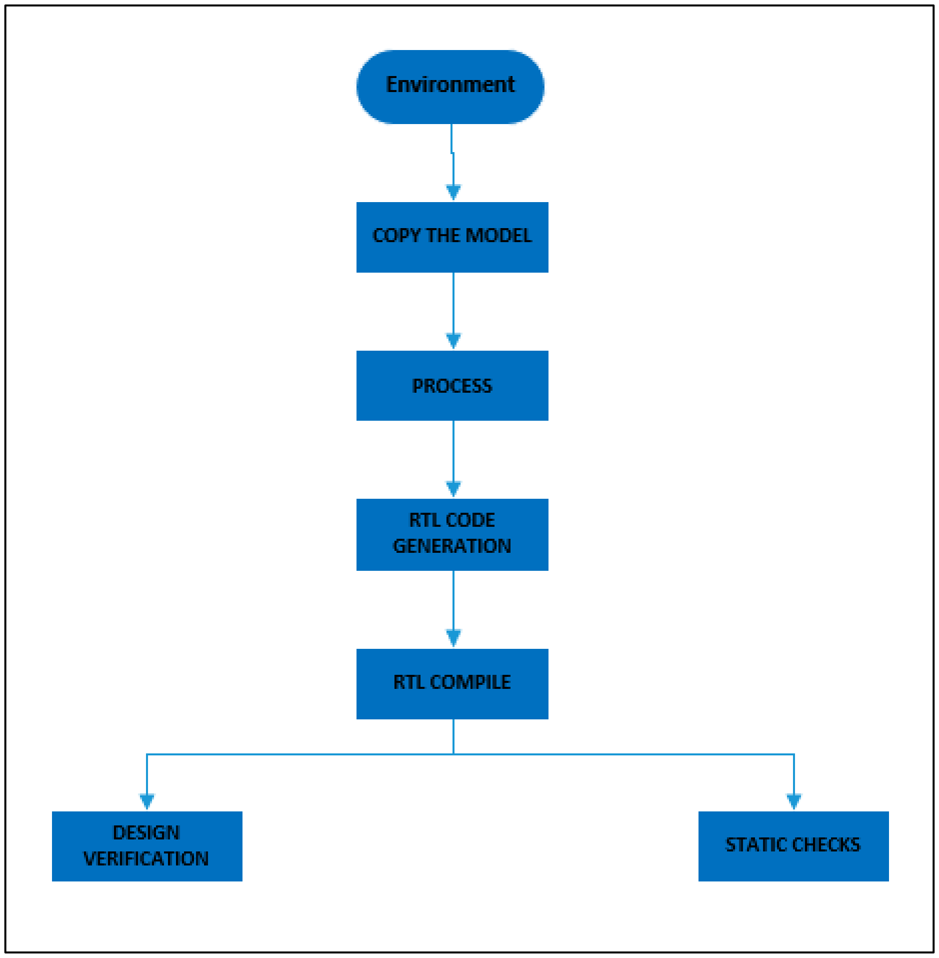


Fig 2: Front End Flow

1) Environment :- Enter your project setup,copy your remote repository and set the working directory and run the flows.Configure the tool version and configuration
2) Copy the model:- Data Management refers to the flow supporting code management integrating register transfer level code changes and release the models. Modern software and hardware relies on the Mangament system similar to Github to provide a mechanism for engineers to write code in parallel share therir work with colleagues.
3) RTL CODE GENERATION:-There is tools which works as a wrapper for generating the RTL by integrating corekits components of Ips using third party coretools and then generated the UPF for RTL generation.

4) RTL COMPILE:- There is tool flow which allows the customer to perform compilation and elaboration only. In this flow user need only input file to perform compilation and elaboration effectively with highest quality.
5) Static Checks: Lint is a static check at the RTL level. It checks the connectivity and synthesizability of design along with RTL quality and checks the structural connectivity.There is cdc checks which will perform clock domain crossing checks using third party tool.Then we perform Low power checks.
6) Design Verification:-This flow is primarly used by validation engineers. It is also used by design engineers to ensure their design are building correctly and passing acceptance testing.

1.3 Problem statement:

In front-end domain there are various stages. Each stage requires different tools for execution of design steps. The front-end domain has more than forty flows of execution of these design steps. If an individual wants to run more than one flow on the IP manually then he /she must be expert of that tools. So, here we are going to write an automated flow which can manage dependencies between these flow for serial & parallel execution depending on input file. The flow is written in PERL scripting so that the overall human efforts and time can be reduced.

1.4 Overview of Thesis

First Chapter of the thesis is the introduction which gives the information about the EDA tools, VLSI ASIC Flow, Front End Flow, Problem statement. Chapter Two is about the Literature survey. Chapter Three about the static flows used in the front-end domains. Chapter Four is about Flow Manager how to write input files, Rules of Input file, Work Flow for Flow Manager, Flow Chart, Results. Chapter Five is regarding the regression testing, regression testcase flow. Chapter Six is about Conclusion and Future Scope.

Chapter 2

Literature Survey

Since RTL is the earlier stage in design abstraction where the design can be modified however much as could reasonably be expected according to satisfying our requirements. For additional setup measures in the pipeling the design flexibility reduced by step by step. Once we enter in post silicon stage we can't modify our the module or design.In design stage RTL is most important stage. There are two challenges we encounter during RTL signoff stage.

1. Developing different nature of SOC which extends unpredictability of the chip gathering and approval.
2. The idea of the third party intellectual property uses should be confirmed and ensured.

Checks should run at RTL abstraction level which recognizes thousands of issues which may be considered as bug in RTL. The amount of time required to run checks at RTL Level is significantly less compared to post silicon level. If we run the RTL checks at rtl abstraction level we can fix the bugs early in design which will requires less time. If we find the bugs at post silicon level in RTL,we need to run all the progressions and update at whatever point. It is good to run the RTL quality checks at earlier stage of design which will reduce entire cycle time.

The RTL checks all the possibility of code wherever there are chances if bug in syntax, clock crossing from one domain to other. RTL Quality checks are categorized as follows:

1. Lint checks
2. CDC checks
3. Power intent
4. Power consumption and power reduction

1) **Lints Checks** : Linting checks implies playing out a static analysis utilizing a RTL descriptive language which contains a bunch of rules which are predefined and rules that mirror a proper coding practice. At the point when any of the standard is penetrated the apparatus reports the mistakes regarding a report which contains all itemized infringement list arranged as far as the standard ID. It assists with getting countless bugs with a not many emphasess, which spares time and cost both. A portion of the linting apparatuses accessible in market are Leda RTL checker by Synopsys, Spyglass by Atrenta and so on The essential progression of the relative multitude of devices are same. They perform build up minds RTL and report any blunders and admonitions

present in term of rule IDs. So it is simpler to break down in light of the fact that once we know the fix for a specific standard, we can resolve all mistakes under that specific principle bunch which spares a great deal of time.

2) **CDC Checks** :- The CDC checks full form is Clock Domain Crossing Checks. The most important check performed on RTL is Clock Domain Crossing (CDC). At the point when we manage various check areas in SOCs, there the idea on clock domain crossing or CDC comes into picture. Since as the SOC multifaceted nature is expanding step by step, the quantity of clock frequencies utilized are additionally getting increased. So huge number of signals cross different clock limits and collaborate with modules present in other clock frequency domain. It should be taken care of with so much accuracy that we should guarantee by appropriate watches that the signals are crossing securely and there is no useful mistakes expected due to these intersections. These checks are done on RTL utilizing numerous instruments. CDC check addresses issues like metastability, information misfortune and information incoherency which are possible issue in RTL.

3) **Power Intent**:- Reducing the gate length step by step device size is shrinking. So optimizing our design for leakage and dynamic power encourages us decrease the power and energy utilization and furthermore reducing packaging cost. Yet, these low power techniques additionally will in general complicate our verification checks and they present risk during amalgamation and actual usage. Full chip recreations are definitely not successful functional technique for confirming the present enormous, complex plans now a days. There are numerous EDA tools accessible in market to check these power intent compatibilities with RTL and if there is any mismatch then those are accounted for so that necessary change should be possible to the UPF what's more, again we will check. These tools are Spyglass-LP by Atrenta and VC-LP by Synopsys.

4) **Power consumption and power reduction:-** The advantage of using mix of low-power components along with low-power plan strategies play major role in today’s technology nodes. For sparing battery life in both portable and non-portable devices. As we are reducing the transistor gate length the static power dissipation is considerable dominant over dynamic power as innovation hubs finding an alternative way for power management techniques. Following are the power reduction techniques

a. Gated Power Domain
b. Multi VDD
c. Multi-threshold CMOS
d. Clock gating
e. Dynamic voltage and frequency scaling

We can evaluate the introduction of the strategy used of power reduction techniques at the RTL stage utilizing Electronic Design Automation (EDA) device accessible for power assessment. Apache has given us a tool named power artist used for power calculation. The EDA tool has capability of calculating average power, instantaneous power. Saurabh Verma, et all [7] discussed basics of clock domain crossing (CDC) problem. Metastability, data loss, data incoherency are main issue of CDC. Synchronizers are used to resolve the problem of metastability presented by Ginosar [3] has shown how metastability is degrades the performance of circuit This paper portrays an approach that will guarantee that the circuit has been planned appropriately to deal with the clock domain crossing issues. A step-by-step approach for checking clock domain intersections is depicted here. Customary check strategies like simulation and static timing analysis are not adequate to distinguish a wide range of issues, which can happen in clock domain crossing. The issues that can happen rely upon the sorts of clock domain crossing. Additionally, the answers for those issues are likewise extraordinary and consequently the verification methods required are distinctive too. A portion of the fundamental issues of clock area intersections have been examined here. The solution for those issues is likewise talked about and a verification methodology is proposed which will guarantee that information is effectively moved across clock domain.

Additionally, power estimation should be possible at RTL stage. D. L. Liu, et all [4] has examined with respect to Power utilization requirements in CMOS VLSI chips. Additionally, M. Nemani et all [6] have proposed High-level area and power consumption for VLSI circuits. The area complexity of multifaceted nature of multi-output Boolean capacities. This depended on changing the multi-output function to an identical single-yield work. The step 1 is checking the synchronize clocks if the synchronize clocks are not present, we need to add the synchronizers or the clock edges may be very close. Separately synchronizes the converging signals. If the clock edges are very close to each other the chances of data loss increase.

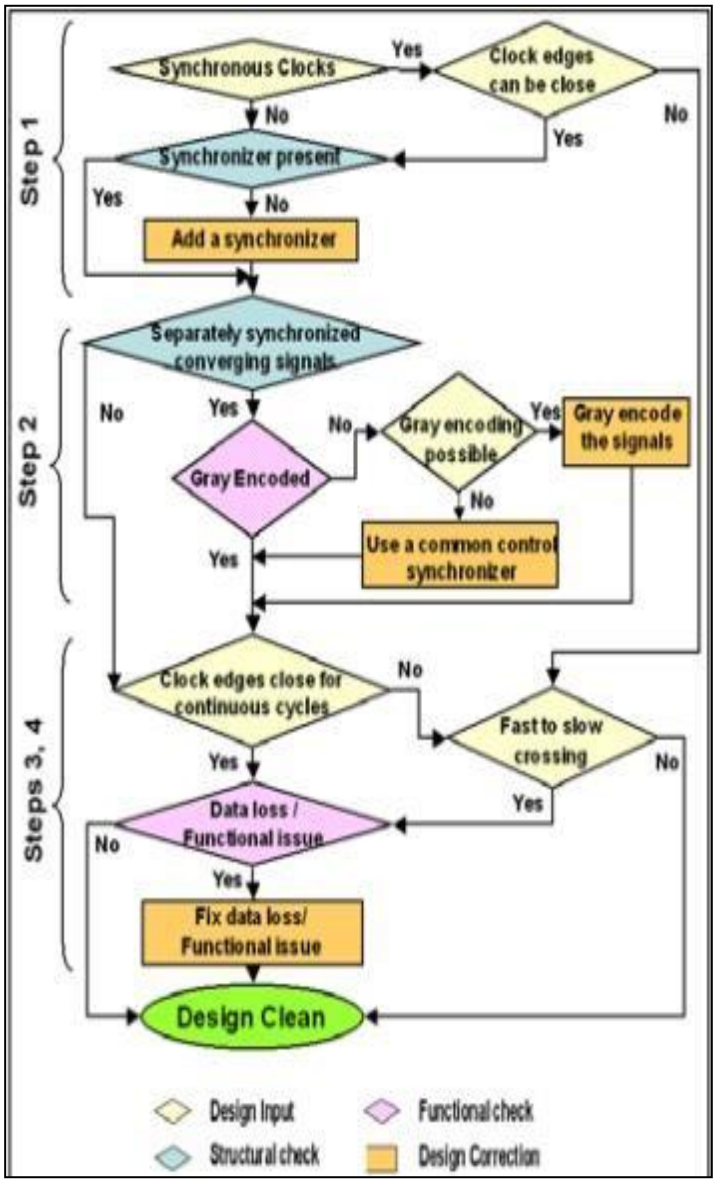


Fig 3 flow of verification methodology

The advantage of this model is that it tends to be effectively described, and it likewise offers a characteristic structure to represent sharing happening in a multi-output work. We have additionally proposed a methodology for assessing expected Cavg to convert a gate counts estimate of area complexity into an estimation of complete capacitance. The predicted capacitance was at that point joined with normal activity assessments to get high level power estimation

Chapter 3
Static Checks on RTL

3.1 Lint Flow

Originally, Lint was the name attached to a Unix utility that could flag non-portable or suspicious C code. The name derives from unwanted bits of fluff from material. Lint is classified as a static analysis tool in that it does not execute the code, instead examining it based on a set of rules. Today, Lint is applied to many different languages including those used in the design of electronic systems. As soon as RTL designers start writing code, they will begin to introduce unintended errors. To eliminate these errors, designers will use a variety of tools to ensure the code is correct before hand-off. Functional errors are typically caught by a mix of static tools (auto-formal and assertion-based) and dynamic tools such as simulation. To eliminate these mistakes, engineers should utilize an asset conveyed to guarantee that the code is precise before hand-off. The process flow can be seen below in figure.

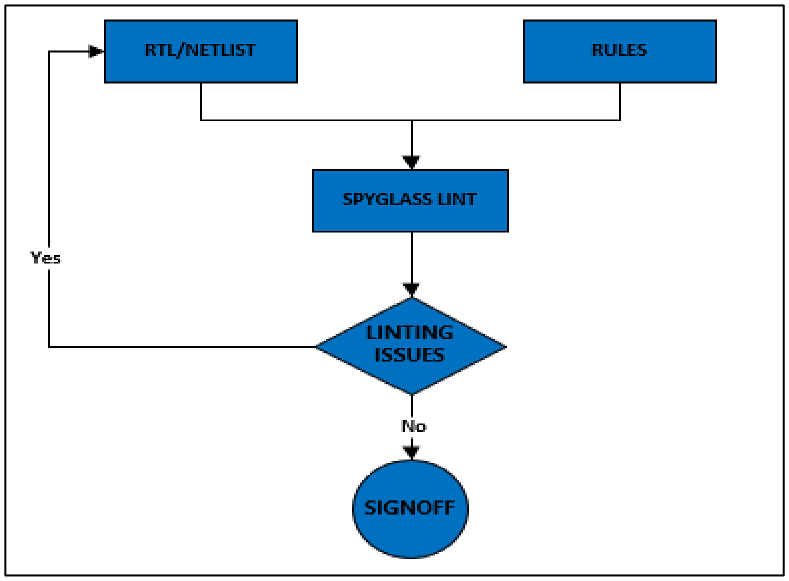


Fig 4: Lint Flow

Linting has the advantages in that it can deliver feedback about troublesome and even dangerous coding styles that would take a much longer time to uncover using simulation. With the right lint tool, you can catch the “low-hanging fruit” before tackling functional errors. With Lint it is easy to caught bugs earlier .

3.2 CDC (Clock Domain Crossing) Flow

SOC are getting complex step by step because of the different functionality which is being added routinely. This is occurring because of the expansion of functionality to the chips. Presently we will talk about one situation when the transfer of data is finished starting with one clock domain then onto other. This transition of data leaves us with a few difficulties. Presently the primary test is the verification and validation of clock domains. This has become the significant test in the verification of design. The way of transferring data from starting with one clock domain then onto the next is known as clock domain crossing.

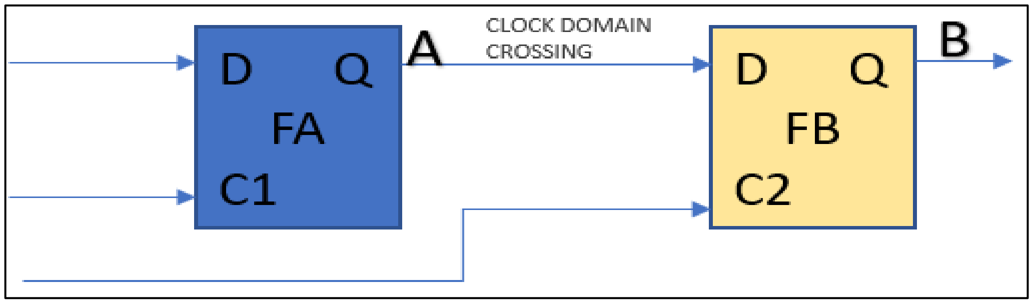


Fig 5 : Clock Domain Crossing

In the figure over, the signal A which begins from clock domain C1 and the desire is that this must be appropriately caught in clock domain C2. Based on the connection between the given two clocks, the issue emerges when the data is moved starting with one clock domain then onto the other domain. The arrangement shifts from issue to issue. There are some customary strategies, for example, simulation and static timing analysis yet these are sufficiently not adequate to confirm that the data is reliably and dependably transferred structure one clock domain to another clock domain.

1) Metastability
2) Data loss
3) Data incoherency

Apparatuses Used to Check Clock Domain Crossings at RTL Stage. There are different companies offering tools to check the clock space crossing issues. The progression of the devices is basically same for all. A portion of the tools are as under.

1) Questa CDC® coach designs 19

2) spyglass CDC® Atrenta

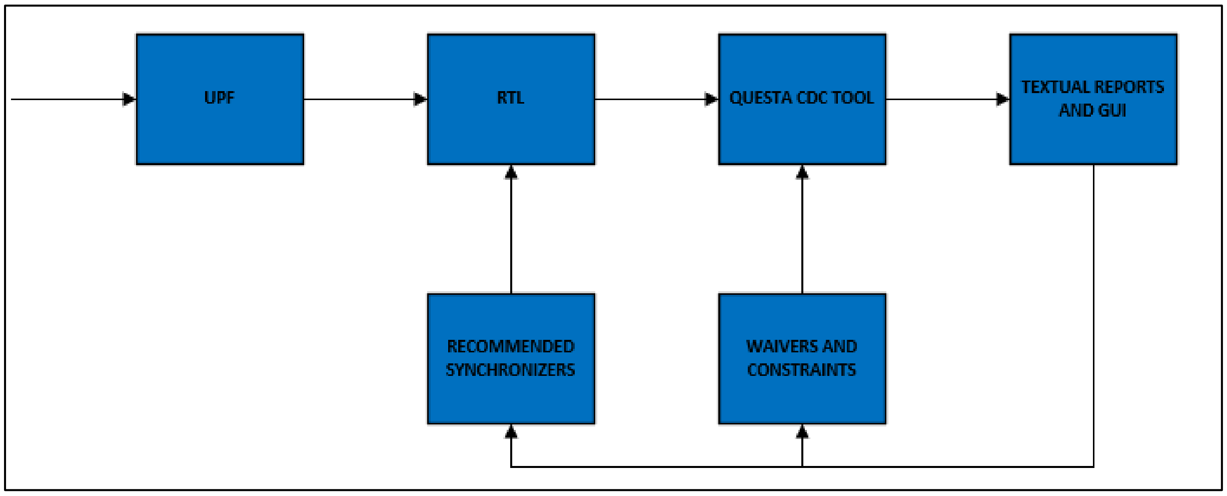


Fig 6 : Questa CDC Flow

The Questa CDC gadget takes the UPF document and RTL as the data sources. The device has inside set up to check all signs which cross in the between of the clock areas. So when we run the apparatus in the wake of stacking our RTL, it checks for all conceivable clock area crossing signals and demonstrated by the arrangement all CDC ways are assembled in the report. The report can be found in the GUI

3.3 Power Estimation Flow on RTL

Low power consumptions are the most important factor of consideration now a days. Designs are becoming more complex and the component count, in turn transistor count is increasing. With the increase in device density the power consumption is also increasing. But there is always a limitation to the power supply. The less is the power consumed the more is the battery life and the more reliable the device is. Now a days there are many low power methodologies implemented in VLSI designs to reduce static and dynamic power consumption like clock gating, power gating etc. For all the implementations separate circuitry are added to RTL making the total DUT more complex and more time taking to analyze. Like the early detection of CDC bugs are done by doing a CDC checks on RTL using different tools, it is also necessary to do a power estimation also in early phase of design cycle. Because if the required power consumption range is not satisfied then we cannot go ahead with that design. If at any post silicon stage, we catch that power criteria is not met and we need to redesign then we have to go to the RTL again do required coding which is a very large amount of re-do task. It takes lot of iteration to finally get the design ready with desired power intent. Power analysis at RTL reduces lot of redesigning efforts, time, and cost too. So, RTL is the best stage to do all analysis.

The power check involves estimation of different power consumed in a design like static power, dynamic power, clock power, latch power, leakage power etc. If any power reduction methodology is used in our design, then it power analysis also gives us the efficiency of that implementation. So, we can compare between different methods and decide which technique is more reliable for our design. So, RTL power estimation helps designer a lot. Tools used to check power estimation at RTL are as follows:

1) Design Power (Synopsys)
2) PowerArtist (ANSYS Apache)

Power Analysis Flow

The essential flow with respect to how the PowerArtist tool works and how it do analysis to produce results about in terms of various power reports is portrayed in the accompanying figure.

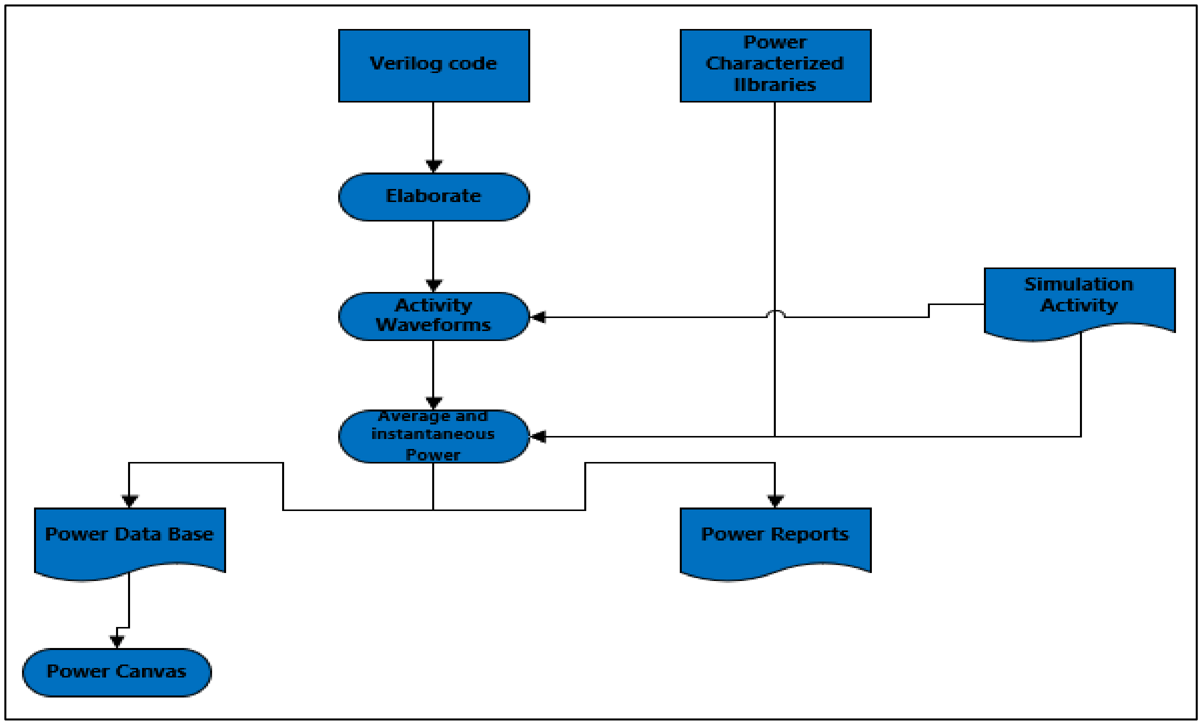


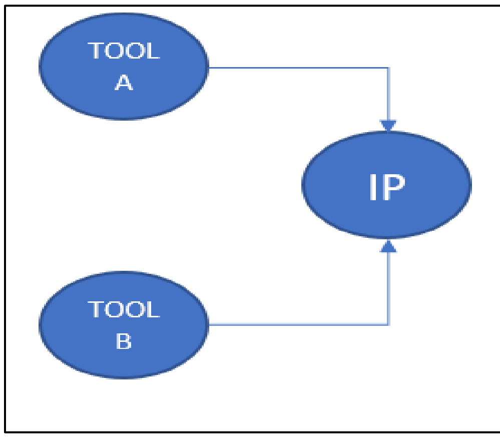
Fig 7: Power Artist Flow

The Design is coded though Hardware using VHDL/Verilog, for that design UPF is specified as power intent. Both UPF and RTL is Passed through static checker in order to check the syntax and semantics of the code. Then it is followed by simulation in order to check the functionality of the design. If that is true then it is passed to next level for synthesis, if it fails then it is again passed back for correction. In order to check UPF power Intent tool is used then if all of these things are true i.e., if it possesses all of these qualities then it is passed into the synthesis tool and to obtain the synthesis result i.e., gate level netlist then power, Area and Timing parameters are measured as per the requirements.

Chapter 4

Automated Flow for Front End Tools

The automated flow works as flow manager for all front- end tools. This flow takes input in initialization file format and generates files depending on different mode of execution. There are nearly forty tools in front end domains how flows helps in reducing the execution time is as follows:



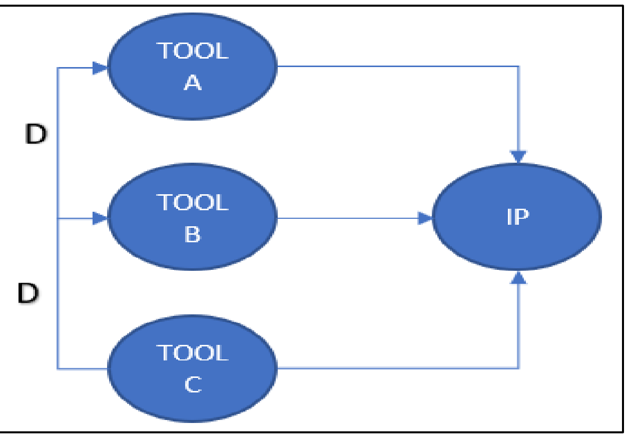


Fig 8 : Sequential Manner

Fig 9 : Automated Flow

Case I : In this case if the person wants to run a tools A and tool B on an IP. Then the execution of tool B will only start after the execution of tool A only. Thus in this case both tools will run in sequential manner.

Case II: In this case depending on the input file the execution of tools will take place. In this case tools will run in parallel mode depending on the dependencies mentioned in the input file.

The work flow for flow manager is as follows:

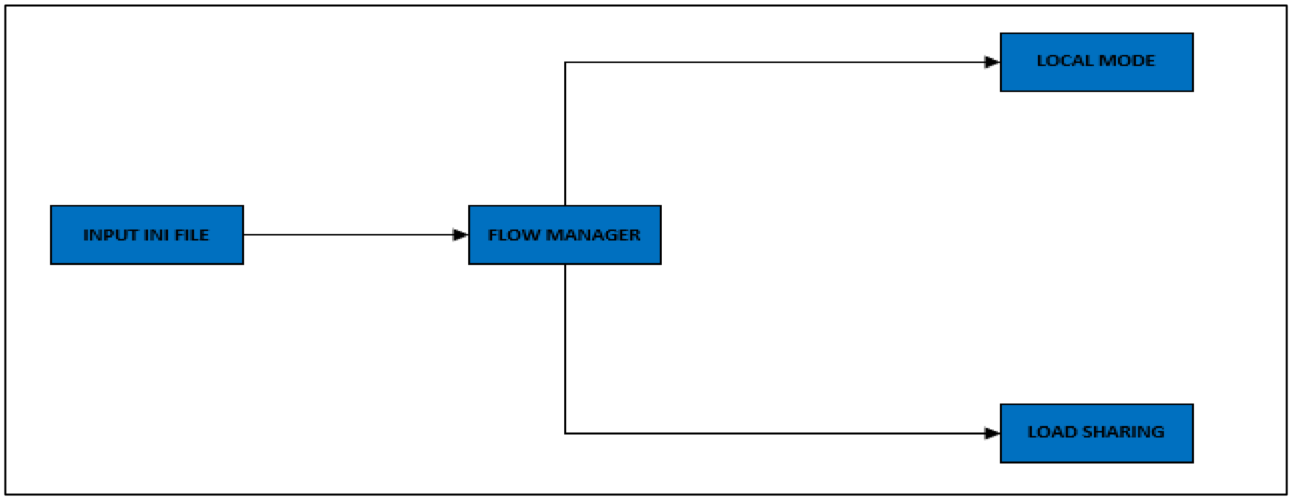
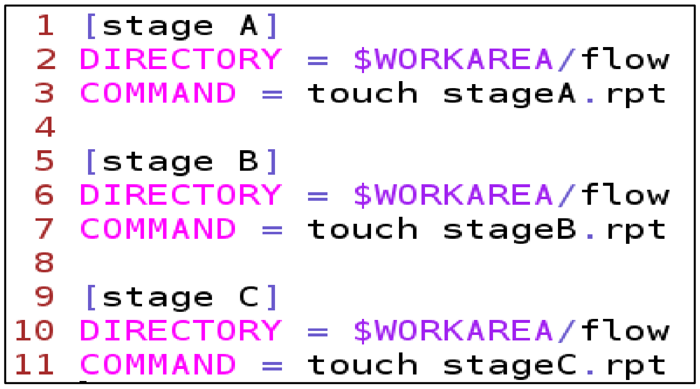


Fig 10 : Work Flow for Automated Flow

4.1 Input file:

The utility requires a input file, which defines set of tasks to be executed. An input file is a file containing a set of directives used by a make build automation tool to generate a target/goal. The input file contains a list of rules. These rules tell the system what commands you want to be executed. The example of Inputfile is shown below:



```
[stage A]
DIRECTORY = $WORKAREA/flow
COMMAND = touch stageA.rpt

[stage B]
DIRECTORY = $WORKAREA/flow
COMMAND = touch stageB.rpt

[stage C]
DIRECTORY = $WORKAREA/flow
COMMAND = touch stageC.rpt
```

Fig 11: Input File

The first line is the tool name one needs to run. Then set the workarea where you want to run the tool. The command required for running a tool.

4.2 Flow Chart for Flow Manager Script:-

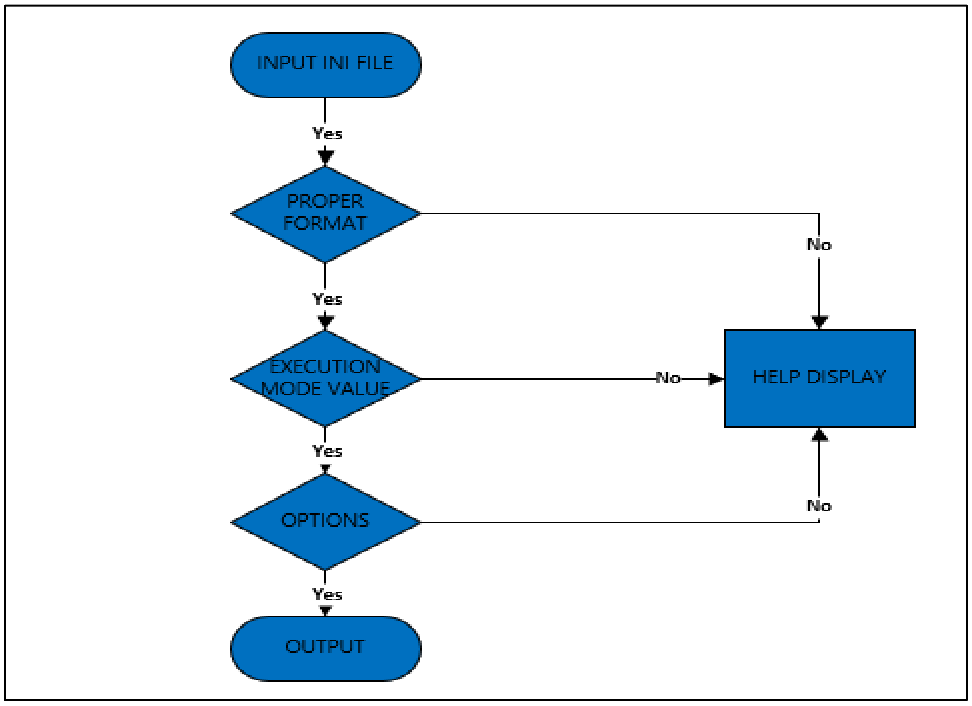


Fig 12: Flow Chart Of Script

The Script will check first the input file. If we provide a wrong input file then it will display an error message and display the help command. There are certain rules for input file format. In inut file stage should be in square brackets only. The absolute path is not supported by input file. Then it will check for modes for execution. There are mainly two modes the execution of flows on local machine or load sharing machine. If this command is not given properly then it will display the error message and help command. If modes of execution are true then it will check for options value such (-stage,+stage,-number of task). If iption values are true then output is generated else it will display an error message if options are not given proper format.

4.3 Results in Local Machine:-

Stages	Individual Stage time	RUNTIME IN SEQUENTIAL	RUN TIME IN PARALLEL MODE
STAGE A	**5 mins**	**Total Time = (STAGE A + STAGE B + STAGE C)** **Total Time = 5+ 25 +30 =60 mins**	**Total Time = STAGE A + max time (STAGE B , STAGE C)** **Total Time = 35 mins**
STAGE B	**25 mins**		
STAGE C	**30 mins**		

In the results we can see when all stages are running in sequential we will consider summation time of all the stages while in case of parallel, we will consider max time STAGE A+ maxtime(STAGE B, STAGE C). Execution time has been reduced by 42% when flows are running in parallel.

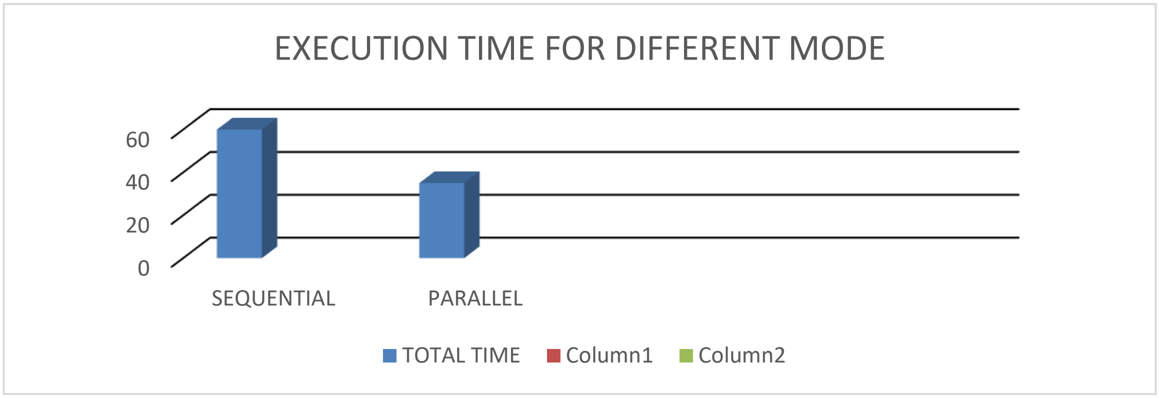


Fig 13: Comparison of runtime in sequential and parallel mode

Chapter 5

Regression Analysis

Regression Analysis is also called as multivariate analysis is a discovery testing method. It is used to check any change in the the flow code doesn't influence the current features of the flow. Regression Analysis guaranteeing that the new code has no bugs, less execution time and easy to debug. Regression analysis is a sort of programming analysis. Testcases are rerun at time of new release to check the features are implemented as per our specification and flow is not breaking anywhere due to new changes and new changes has not conveyed any bugs. When there is huge change in the current flow code functionality, regression analysis is required. It is method of re-checking the unaltered code is executing properly along with the new changes

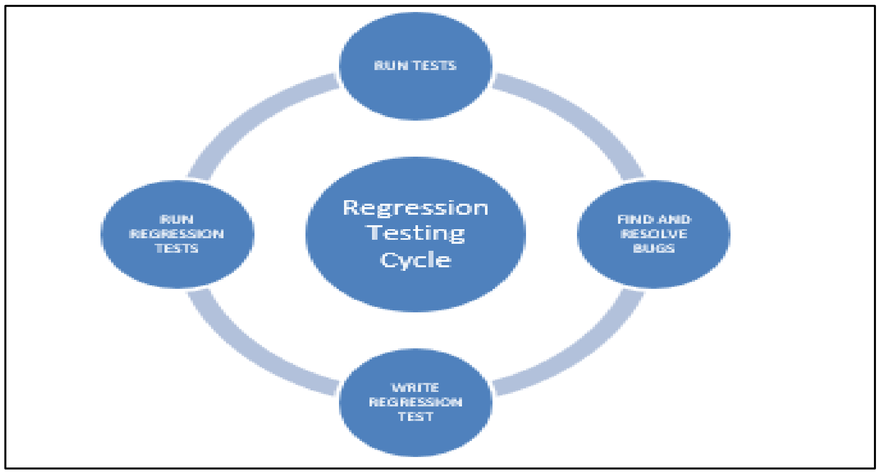


Fig 14 : Regression Analysis

5.1 Regression Analysis is required to be done at following scenarios:-

- New features are added to flow.
- When Changes required.
- When the defect fixed or environment changes

5.2 Regression Analysis Execution

The requirement for regression analysis comes when new features (enhancements), bug fixes, reduce the execution time and evacuation of current functionality occurs in flow code. These changes may affect the current functionality of the system. Regression Analysis becomes compulsory in that situation.

Regression Analysis can be executed using the following steps:

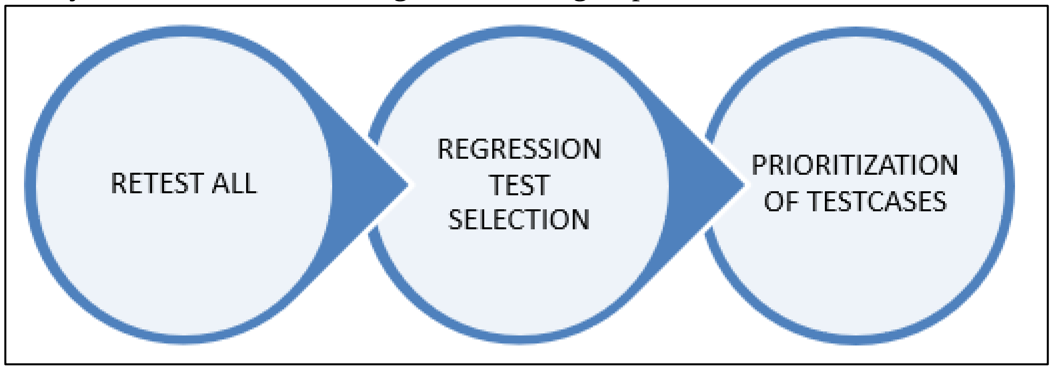


Fig 15 : Regression Testing Plan

RETEST ALL

Re-Test is one of the ways to do deal with relapse testing. In this methodology, all the experiment suits ought be re-run. Here we can characterize re-test as when a test comes up short, and we decide the reason of the failure is a code issue. The issue is accounted for, we can expect another form of the product wherein imperfection fixed. For this situation, we should execute the test again to confirm that the issue is fixed. Some will allude to this as affirmation testing. The re-test is very pricey, as it needs assets and time.

1. Regression Analysis Selection:

In this statergy, a selected experiment suit will run not the whole experiment suit directory. The selected suits are categorized as follows

Reusable Test cases.

Obsolete Test cases.

5.3 Regression Analysis Flow

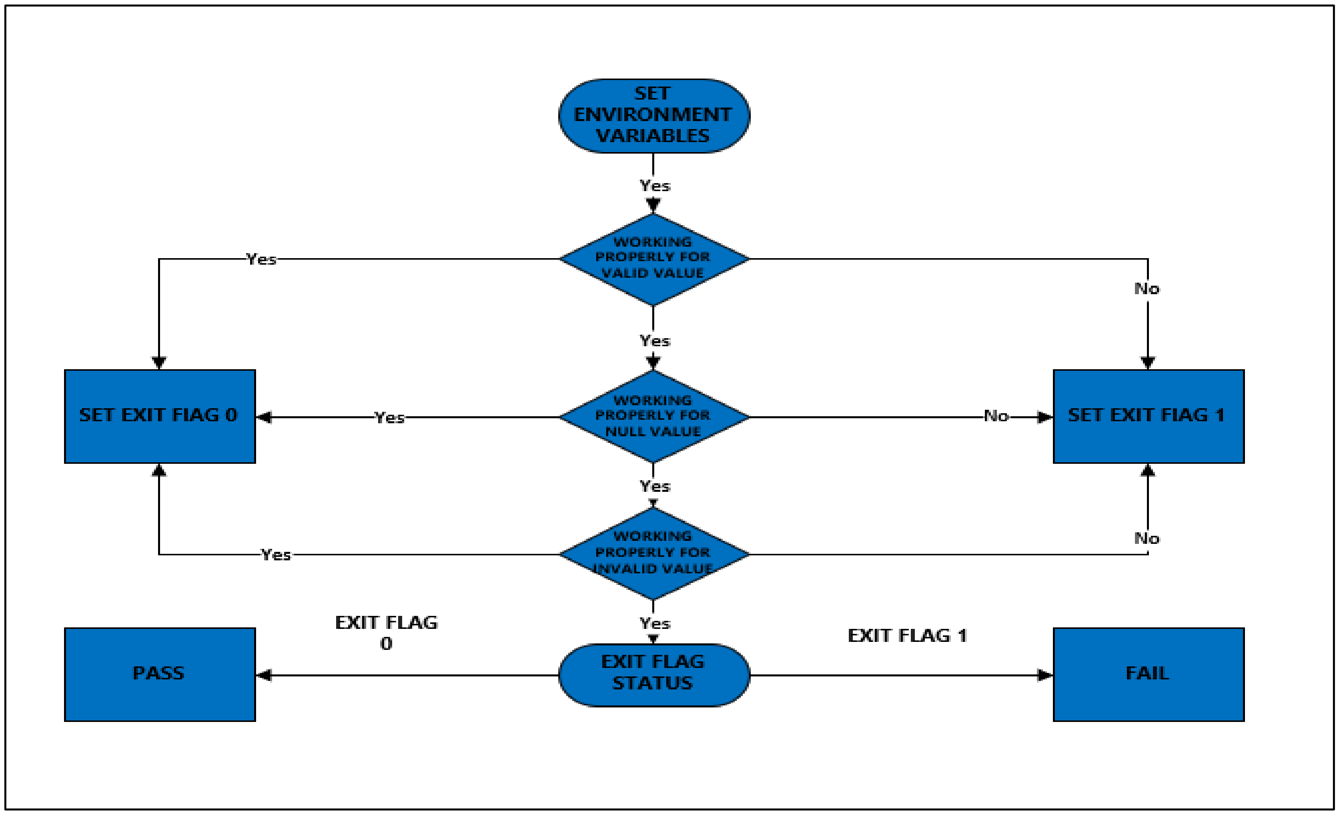

SET ENVIRONMENT VARIABLES
Yes
WORKING PROPERLY FOR VALID VALUE
Yes
No
Yes
WORKING PROPERLY FOR NULL VALUE
SET EXIT FIAG 0
Yes
No
SET EXIT FIAG 1
Yes
WORKING PROPERLY FOR INVALID VALUE
Yes
No
Yes
EXIT FLAG 0
EXIT FLAG STATUS
EXIT FLAG 1
PASS
FAIL

Fig 16: Regression Testcase Flow

- First setup the environment variables and enter the project setup.
- Check the input configuration by giving the proper value. If the config variable is behaving properly then set the exit flag variable as zero else one.
- Check the input configuration by giving the empty value. If the config variable is behaving properly then set the exit flag variable as zero else one.
- Check the input configuration by giving the invalid value. If the config variable is behaving properly then set the exit flag variable as zero else one.
- Now check the status of the exit flag if it is one at end of all scenarios the testcase will fail and if it is zero testcase will pass.

5.4 Advantages of Regression Analysis

- With help of regression analysis the quality of product got improved
- We can get confidence that new enhancement donot hinder our previous functionality and implemented properly.
- We can automate the regression to run in parallel which will reduce the execution time.

Chapter 6

Conclusion and Future Scope

The basic agenda of this thesis is to understand the available methods of execution of static check tools used in front end domain. From the results of the manual execution of these tools on local machine, there is noticeable reduction in execution time. It requires engineering efforts and consume more time. By implementing automation flow, less efforts are required, have better accuracy, and saved execution time.

We have successfully automated the front-end tools flow after doing the study of the front-end tools. Perl script has been written for the automated flow. The new features we are going to add in this script. We can proceed for regression testing of these script soon and enhance the features depending on request of different teams and provide deployment services for these scripts and also perform regression testing.

References

[1] E.Cummings, C. (2008). Clock Domain Crossing (CDC Design & Verification Techniques) using System Verilog. *SNUG*.

[2] Karimi, N. (septemeber 2013). Detection, Diagnosis and Recovery from Clock Domain Crossing Failures in Multiclock SoCs. *IEEE transcation on computer-aided design of Integrated circuits and systems*, (p. Vol 32 no 9).

[3] Kim, H.-K. (2013). Testing of synchronizers in asynchronous FIFO. *IEEE Trans.*, (pp. 49-72).

[4] Liu, D. (1994). Power Consumption Estimation in CMOS VLSI chips. *IEEE Journal*, pp 663-670.

[5] Nemanu, M. (1999). High-level area and power estimation of VLSI circuits. *IEEE Trans.Computer-Aided Design*, (pp. 697-713).

[6] Verma, S. (n.d.). *Understanding clock domain crossing issues.* Atrenta.

[7] https://alison.com/course/fundamentals-of-perl-programming

[8] https://learn.perl.org/

[9] https://www.learn-perl.org/en/

Turnitin Originality Report

Processed on: 12-May-2021 11:25 IST
ID: 1584193065
Word Count: 5774
Submitted: 1

Report By Hitesh Dadlani

Similarity Index	Similarity by Source	
9%	Internet Sources:	9%
	Publications:	0%
	Student Papers:	N/A

7% match ()
Patra, Smrutisikta. "RTL Design Quality Checks for Soft IPs", 2016

1% match (Internet from 12-Nov-2020)
https://www.einfochips.com/blog/asic-design-flow-in-vlsi-engineering-services-a-quick-guide/#:~:text=ASIC%20design%20flow%20is%20a%20mature%20and%20silicon-pro

1% match (Internet from 01-Apr-2021)
https://dzone.com/articles/how-does-the-asic-design-flow-cycle-works

“Automated Flow To Manage Dependencies between Different Flows – Serial & Parallel Execution of Design Steps” A Major Report Submitted in partial fulfillment of the requirements for the degree of MASTER OF TECHNOLOGY IN ELECTRONICS AND COMMUNICATION ENGINEERING (VLSI Design) Submitted By Hitesh Dadlani (19MECV06) Electronics & Communication Engineering Department Institute of Technology Nirma University Ahmedabad - 382 481 May 2021 1 ABSTRACT The EDA industry has evolved rapidly over the last two decades. Designer’s productivity has raised due to improved EDA tools over the last quarter of the century. Moore’s law was made more effective and efficient with help of EDA tools. EDA tools have become essential in the design of chips with high complexity.The first integrated chips were small-scale integration chips where the gate count was very small. With the advancement of technology, designers were able to place circuits with thousands of gates on a chip. As the number of gate counts increased design process started getting complex, designers felt the need to automate the process. Developers moved to the development of EDA tools in the design flow. The development of EDA tools in design flow depends on four major interrelated principles. The four principles on which EDA tools are developed are modularity, integrity, operate incrementally, have less runtime. The development of EDA tools in design flow depends on four major interrelated principles. This Project introduces automated flow for front-end tools which eased the process of front-end tools execution.This project covers about the various static flows used in front end domain.The automated flow will manage dependencies between these flow for serial & parallel execution depending on input file. The automated flowcomes with many features that can reduce the execution time and human efforts. The features includes dependency stage, different modes of execution, simple input format for project and tool configurations. The developed automated flow supports the standard environment used for INTEL projects.The automated flow is developed using Perl. 2 Contents

NOMENCLATURE Abbreviation RTL DRC LVS CDC ASIC VLSI CAD EDA PERL Register Transfer Level Design Rule check Layout Rule Check Clock Domain Crossing Application-Specific Integrated Circuit Very Large Scale Integration Computer-Aided Design Electronic Design Automation Practical

Extraction and Report Language 5 Chapter 1 Introduction Electronic Design Automation (EDA) is at the center of innovation propels in improving human existence and utilize each day. Given an electronic framework demonstrated at the electronic framework level (ESL), EDA automates the plan and test cycles of checking the rightness of the ESL plan against the specification of the electronic framework, taking the ESL plan through different synthesis and verification steps, lastly testing the fabricated electronic framework to guarantee that it meets the details and quality prerequisites of the electronic framework. The electronic framework can likewise be a printed circuit board (PCB) or essentially IC. EDA consist of of set of hardware and software co-plan,synthesis,verification flows that check the electronic system level design ,translate the the electronic system level design to register transfer level and then takes register transfer level through verification and synthesis stages at gate level and After synthesis we get a gate level netlist, which shows the interconnection of various components used in our design and eventually produces physical design (GDSII) format in that last stage of the tapeout, the engineer performs wafer processing, packaging, testing, verification and delivery to the physical IC. GDSII is the file produced and used by the semiconductor foundries to fabricate the silicon and handled to client. 1.1 VLSI ASIC FLOW Today, ASIC design flow is an exceptionally full grown cycle in silicon plan. The ASIC design flow and its different stages in VLSI designing that we describe are based on accepted methods and best practices in ASIC chip plans. This blog endeavors to clarify various stages in the ASIC design flow, beginning from ASIC plan idea and moving from requirements(specification) to benefits. 1) Specifications:- This is the first stage in the design process where we define the important parameters of the system that must be designed into specification. Specifications details can be as follows • Goals and requirements of the design. • Functionality of chip • Performance figures like Area and power • Technology requirements like size and space Fig 1: VLSI ASIC FLOW 7 2) RTL DESCRIPTION: In view of the specification of the design we need to depict a plan and make a RTL code in one of the Hardware Description Language(HDL) either Verilog or VHDL. These two HDLs are utilized most generally. When the RTL code and testbench are produced, the RTL group deals with RTL description – they make an translation of the RTL code into an entryway level netlist utilizing a synthesis tool that meets required timing constraints. 3) FUNCTIONAL VERIFICATION:- This Step affirms the functionality of design by applying the different stimulus to the design check the output.It is step where the design group and RTL group come into the cycle where they produce testbench in SVA. This is called as behavior Simulation. There are two kinds of tools for simulation: a) Functional Simulation Tools: After functionally verify the design it will tell us the logic we implemented is matching our specification. b) Timing Simulation Tool: Verifies that circuit configuration meets the timing constraints prerequisites and affirms the design is liberated from circuit signal delay. The tools used for performing this steps are SYNOPSYS VCS, MENTOR GRAPHICS QUESTASIM 4) LOGIC SYNTHESIS: Upto the stage 3 technology was not included however from logic synthesis step, there is a dependence of technology that we are utilizing. The logic synthesis tool translate your Verilog/VHDL code into a Gate level netlist. The instruments utilized for this progression are -SYNOPSYS DESIGN COMPILER, CADENCE-GENUS, MENTOR GRAPHICS-LEONARDO SPECTRUM. 5) GATE LEVEL NETLIST: After Logic synthesis we get an gate level netlist, which shows the interconnection of different segments utilized in our design. 6) LOGIC EQUIVALENCE CHECK(LEC): LEC is the process of equivalence check among pre and post plan design. This equivalence checks if your design is logically right. If its not , at that point the engineer needs to again do the logic synthesis and make the design logically right, else we proceed onward to physical design. 8 The devices utilized in this progression are SYNOPSYS-FORMALITY, CADENCE- CONFORMAL, MENTOR GRAPHICS-QUESTASLEC. 7) PLACE & ROUTE: Placement is the way toward placing standard cells in row. A bad placement situation requires larger area and furthermore degardes the performance. Different variables, similar to the timing requirements, the net lengths and thus the connections of cells, leakage power should be taken consideration. It eliminates timing voilation. Routing is essentially the interconnection of the wires. Effective routing is must for the design, since it diminishes the voilation as well as improves the signal delay. The tools required for performing this step are CADENCE-INNOVUS, SYNOPSYS-IC COMPILER,MENTOR GRAPHICS OLYMPUS. 8) PHYSICAL LAYOUT: After Placement and Routing we will have the physical layout of our plan. We need to do actual physical verfication of our design. We have to do physical verification of our layout, which is what discussed in the next step in detail. 9) PHYSICAL VERIFICATION: After routing, ASIC plan goes through three stages of verification, known as signoff checks. This stage assists with checking whether the layout working the manner in which it was intended to. The accompanying checks are followed to avoid any mistakes not long before the tape out: a) Layout versus schematic(LVS) is a cycle of checking that the geometry/layout coordinates the schematic/netlist. b) Design rule checks(DRC) is the way toward matching that the calculation in the GDS file adheres to the rules given by the foundry. 9 1.2 Front End Flow Fig 2: Front End Flow 1) Environment :- Enter your project setup,copy your remote repository and set the working directory and run the flows.Configure the tool version and configuration 2) Copy the model:- Data Management refers to the flow supporting code management integrating register transfer level code changes and release the models. Modern software and hardware relies on the Mangament system similar to Github to provide a mechanism for engineers to write code in parallel share therir work with colleagues. 3) RTL CODE GENERATION:-There is tools which works as a wrapper for generating the RTL by integrating corekits components of Ips using third party coretools and then generated the UPF for RTL generation. 10 4) RTL COMPILE:- There is tool flow which allows the customer to perform compilation and elaboration only. In this flow user need only input file to perform compilation and elaboration effectively with highest quality. 5) Static Checks: Lint is a static check at the RTL level. It checks the connectivity and synthesizability of design along with RTL quality and checks the structural connectivity.There is cdc checks which will perform clock domain crossing checks using third party tool.Then we perform Low power checks. 6) Design Verification:-This flow is primarly used by validation engineers. It is also used by design engineers to ensure their design are building correctly and passing acceptance testing. 1.3 Problem statement: In front end domain there are various stages. Each stage requires different tools for execution of design steps. The front end domain has more than forty tools of execution of these design steps. If an individual wants to run all tools on the IP manually then execution time will be more. Customers facing an issue with running the tools on IP. Customer send to us for evaluation and request us to automate the front end tools flow. So here I am going to write an automated flow which can will manage dependencies between these flow for serial & parallel execution depending on input file. The flow is written in perl scripting so that the overall human efforts and time can be reduced. 1.4 Overview of Thesis First Chapter of the thesis is the introduction which gives the information about the EDA tools, VLSI ASIC Flow, Front End FlowProblem statement Chapter Two is about the Literature survey. Chapter Three is about the static flows used in the front end domains. Chapter Four is about Flow Manager how to write Input files , Rules of Input file , Work Flow for Flow Manager, Flow Chart , Results Chapter Five is about Conclusion and Future Scope. 11 Chapter 2 Literature Survey Since RTL is the earlier stage in design abstraction where the design can be modified however much as could reasonably be expected according to fulfilling our necessities. For additional setup measures in the stream the plan adaptability diminishes bit by bit. Once we enter in post silicon stage we can’t modify our the module or design.In design stage RTL is most important stage. There are two challenges we encounter during RTL signoff stage. 1. Developing different nature of SOC which extends unpredictability of the chip gathering and approval. 2. The idea of the third party intellectual property uses should be confirmed and ensured. Checks should run at RTL abstraction level which recognizes thousands of issues which may be considered as bug in RTL. The amount of time required to run checks at RTL Level is significantly less compared to post silicon level. If we run the RTL checks at rtl abstraction level we can fix the bugs early in design which will requires less time. If we find the bugs at post silicon level in RTL,we need to run all the progressions and update at whatever point. It is good to run the RTL quality checks at earlier stage of design which will reduce entire cycle time. The RTL checks all the possibility of code wherever there are chances if bug in syntax, clock crossing from one domain to other. RTL Quality checks are categorized as follows: 1. Lint checks 2. CDC checks 3. Power intent 4. Power consumption and power reduction 1) Lints Checks : Linting checks implies playing out a static analysis utilizing a RTL descriptive language which contains a bunch of rules which are predefined and rules that mirror a proper coding practice. At the point when any of the standard is penetrated the apparatus reports the mistakes regarding a report which contains all itemized infringement list arranged as far as the standard ID. It assists with getting countless bugs with a not many emphasess, which spares time and cost both. A portion of the linting apparatuses accessible in market are Leda RTL checker by Synopsys, Spyglass by Atrenta and so on The essential progression of the relative multitude of devices are same. They perform build up minds RTL and report any blunders and admonitions 12 present in term of rule IDs. So it is simpler to break down in light of the fact that once we know the fix for a specific standard, we can resolve all mistakes under that specific principle bunch which spares a great deal of time. 2) CDC Checks :- The CDC checks full form is Clock Domain Crossing Checks. The most important check performed on RTL is Clock Domain Crossing (CDC). At the point when we manage various check areas in SOCs, there the idea on clock domain crossing or CDC comes into picture. Since as the SOC multifaceted nature is expanding step by step, the quantity of clock frequencies utilized are additionally getting increased. So huge number of signals cross different clock limits and collaborate with modules present in other clock frequency domain. It should be taken care of with so much accuracy that we should guarantee by appropriate watches that the signals are crossing securely and there is no useful mistakes expected due to these intersections. These checks are done on RTL utilizing numerous instruments. CDC check addresses issues like metastability, information misfortune and information incoherency which are possible issue in RTL. 3) Power Intent:- Reducing the gatelength step by step device size is shrinking. So optimizing our design for leakage and dynamic power encourages us decrease the power and energy utilization and furthermore reducing packaging cost. Yet, these low power techniques additionally will in general complicate our verification checks and they present risk during amalgamation and actual usage. Full chip recreations are definitely not successful functional technique for confirming the present enormous, complex plans now a days. There are numerous EDA tools accessible in market to check these power intent compatibility with RTL and if there is any mismatch then those are accounted for so that necessary change should be possible to the UPF what's more, again we will check. These tools are Spyglass-LP by Atrenta and VC-LP by Synopsys. 4) Power consumption and power reduction :- The advantage of using mix of low-power components along with low-power plan strategies play major role in today’s technology nodes. For sparing battery life in both versatile and non-verstaile devices . As we are reducing the transistor gate length the static power dissipation is considerable dominant over dynamnic power as innovation hubs finding an alternative way for power management techniques. Following are the power reduction techniques 13 a. Gated Power Domain b. Multi

VDD c. Multi-threshold CMOS d. Clock gating e. Dynamic voltage and frequency scaling We can evaluate the introduction of the strategy used of power reduction techniques at the RTL stage utilizing Electronic Design Automation(EDA) device accessible for power assessment. Apache has given us a tool named power artist used for power calculation. The EDA tool has capability of calculating average power, instantenous power. Saurabh Verma, et all [7] discussed basics of clock domain crossing (CDC) problem. Metastability, data loss, data incoherency are main issue of CDC. Synchronizers are used to resolve the problem of metastability presented by Ginosar [3] has shown how metastability is degrades the performance of circuit This paper portrays an approach that will guarantee that the circuit has been planned appropriately to deal with the clock domain crossing issues. A step by step approach for checking clock domain intersections is depicted here. Customary check strategies like simulation and static timing analysis are not adequate to distinguish a wide range of issues, which can happen in clock domain crossing. The issues that can happen rely upon the sorts of clock domain crossing. Additionally, the answers for those issues are likewise extraordinary and consequently the verification methods required are distinctive too. A portion of the fundamental issues of clock area intersections have been examined here. The solution for those issues are likewise talked about and a verification methodology is proposed which will guarantee that information is effectively moved across clock domain. Additionally power estimation should be possible at RTL stage. D. L. Liu, et all [4] has examined with respect to Power utilization requirements in CMOS VLSI chips. Additionally M. Nemani et all [6] have proposed High-level area and power consumption for VLSI circuits. The area complexity of multifaceted nature of multi-output Boolean capacities. This depended on changing the multi-output function to an identical single-yield work.the step 1 is check the synchronize clocks if the synchrconize clocks are not present we need to add the synchronizers or the clock edges may be very close.Separately synchronizes the converging signals. If the clockedges are very close to each other the chances of data oss increases.if the data loss in=s find in the design you need to clean it and then only design becomes clean. 14 Fig 3 flow of verification methodology The advantage of this model is that it tends to be effectively described, and it likewise offers a characteristic structure to represent sharing happening in a multi-output work. We have additionally proposed a methodology for assessing expected Cavg to convert an gate counts estimate of area complexity into an estimation of complete capacitance. The predicted capacitance was at that point joined with normal activity assessments to get high level power estimation 15 Chapter 3 Static Checks on RTL 3.1 Lint Flow Lint was initially the name applied to a Unix utility, which would signal nonportable or faulty C code. The name gets from material with superfluous bits of fluff. Lint is inspected as a static method in which it doesn't execute the code, but instead tests it observes an collection of rules. Today, Lint is applied to various languages incorporating those utilized in the arranging of electronic frameworks. When RTL engineers begin composing code, they will begin making unintended mistakes. To eliminate these mistakes, engineers should utilize a asset conveyed to guarantee that the code is precise before hand-off. The process flow can be seen below in figure. Fig 4: Lint Flow Normally, functional mistakes are detected by a static tools (autoformal and statement based), and dynamic tools, for example, simulation tools. Linting has the advantages that can give contribution on problematic and even hazardous coding. types that may take far longer to find utilizing simulation. you will snatch the easy pickins prior to handling it precisely with the the proper lint device. 16 3.2 CDC (Clock Domain Crossing) Flow SOC are getting complex step by step because of the different functionality which is being added routinely. This is occurring because of the expansion of functionality to the chips. Presently we will talk about one situation when the transfer of data is finished starting with one clock domain then onto other. This transition of data leaves us with a few difficulties. Presently the primary test is the verification and validation of clock domains. This has become the significant test in the verification of design. The way of transferring data from starting with one clock domain then onto the next is known as clock domain crossing. Fig 5 : Clock Domain Crossing In the figure over, the signal A which begins from clock domain C1 and the desire is that this must be appropriately caught in clock domain C2. Based on the connection between the given two clocks, the issue emerges when the data is moved starting with one clock domain then onto the other domain. The arrangement shifts from issue to issue. There are some customary strategies, for example, simulation and static timing analysis yet these are sufficiently not adequate to confirm that the data is reliably and dependably transferred structure one clock domain to another clock domain. 1) Metastability 2) Data loss 3) Data incoherency Apparatuses Used To Check Clock Domain Crossings At RTL Stage. There are different companies offering tools to check the clock space crossing issues. The progression of the devices are basically same forall. A portion of the tools are as under. 17 1) Questa CDC® coach designs 19 2) spyglass CDC® Atrenta 3) VC CDC® Synopsys Fig 6 : Questa CDC Flow The Questa CDC gadget takes the UPF document and RTL as the data sources. The device has inside set up to check all signs which cross in the between of the clock areas. So when we run the apparatus in the wake of stacking our RTL, it checks for all conceivable clock areacrossing signals and as demonstrated by the arrangement all CDC ways are assembled in the report. The report can be found in the GUI 3.3 Power Estimation Flow on RTL Low Power Consumption is the primary factor of thought now a days. Configuration are ending up being more unusual and segment check, accordingly semiconductor tally is expanding. With the increment in gadget thickness power required is likewise more. In any case, there is reliably a limitation to the power supply. The less is the power devoured through the more is the battery life and the more solid the contraption is. By and by there are some low power techniques executed in VLSI configuration to diminish static and dynamic power utilization like clock gating, power gating, etc For all the use free equipment are added to RTL making the outright DUT more intricate and extra time needed to dissect. Like the early recognizable proof of CDC bugs are done by doing a CDC watches out for RTL using different apparatuses, it is moreover critical to do a power evaluation in like manner in early phase of configuration cycle. Since, in such a case that the vital power utilization range isn't satisfied then we can't proceed with that plan. In case at any post silicon stage we get that power necessity isn't met additionally, we need to redesign then we need to go to the RTL again do required coding which is a great deal of re-take care of errand.Power examination at RTL decreases parcel of upgrading endeavors, time, and cost as well. So 18 RTL is the best stage to do allexamination. The power check includes assessment of various power burned- through in a plan like static power, dynamic power, clock power, hook power, spillage power and so on Assuming any force decrease technique is utilized in our plan, it power examination likewise gives us the productivity of that execution. So we can analyze between various strategies and choose which procedure is more dependable for our plan. So RTL power assessment helps planner a ton to get wanted power expectation. Apparatuses Used To Check Power Estimation At RTL stage power assessment at RTL stage is finished with the assistance of a device. There are numerous instruments accessible in market. 1) Design Power (Synopsys) 2) PowerArtist (ANSYS Apache) 3) InCyte Chip Estimator (Cadence) Power Analysis Flow The essential flow with respect to how the PowerArtist tool works and how it do analysis to produce results about in terms of various power reports is portrayed in the accompanying figure. Fig 7 : Power Artist Flow The Design is coded though Hardware using VHDL/Verilog, for that design UPF is specified as power intent. Both UPF and RTL is Passed through static checker in order to check the syntax and semantics of the code. Then it is followed by simulation in order to check the functionality of the design. If that is true then it is passed to next level for synthesis, if it fails then it is again passed back for correction.. In order to check UPF power Intent tool is used then if all of this things are true i.e. if it possess all of these qualities then it is passed into the synthesis tool and to obtain the synthesis result i.e. gate level netlist then power , Area and Timing parameters are measured as per the requirements. 19 Chapter 4 Automated Flow For Front End Tools The automated flow works as flow manager for all front end tools. This flow takes input in initialization file formatand generates files depending on different mode of execution.There are nearly forty tools in front end domains how flows helps in reducing the execution time ia as follows: Fig 8 : Sequential Manner Fig 9 : Automated Flow Case I : In this case if the person wants to run a tools A and tool B on an IP. Then the execution of tool B will only start after the execution of tool A only. Thus in this case both tools will run in sequential manner. Case II: In this case depending on the input file the execution of tools will take place. In this case tools will run in parallel mode depening on the dependencies mentioned in the input file. The work flow for flow manager is as follows: Fig 10 : Work Flow for Automated Flow 20 4.1 Input file: The utility requires a input file, which defines set of tasks to be executed. A input file is a file containing a set of directives used by a make build automation tool to generate a target/goal. The input file contains a list of rules. These rules tell the system what commands you want to be executed.The example of Input file is shown below: Fig 11: Input File The first line is the tool name one needs to run. Then set the workarea where you want to run the tool. The command required for running a tool. 4.2 Flow Chart for Flow Manager Script:- Fig 12: Flow Chart Of Script 21 The Script will check first the input file. If we provide a wrong input file then it will display an error message and display the help command. There are certain rules for input file format. In inut file stage should be in square brackets only. The absolute path is not supported by input file. Then it will check for modes for execution. There are mainly two modes the execution of flows on local machine or load sharing machine. If this command is not given properly then it will display the error message and help command. If modes of execution are true then it will check for options value such (-stage,+stage,-number of task). If iption values are true then output is generated else it will display an error message if options are not given proper format. 4.3 Results in Local Machine:- Fig 13 : Parallel Execution Of Flows Fig 14: Execution of Single Stage In the results you can see all tools are running parallel at a same time and complete the execution of flows at same time. In second result you can see if you want to run particular stage you can also run the particular stage. 4.4 Advance Features:- • global_env : stage to declare environment variables intended to be set before every run • global_cfg : $flow. cfg variable override – applies to all $CMD • Load_settings : Global Netbatch settings applies to all stages • params: Section to specify any $param to use in the INI stage commands • include : section to include multiple INI files along with this top level INI file to create final.ini in the output area 22 Chapter 5 Regression Analysis Regression Analysis is also called as multivariate analysis is a discovery testing methods. It is used to check any change in the the flow code doesn't influence the current features of the flow. Regression Analysis guaranteeing that the new code has no bugs, less execution time and easy to debug. Regression analysis is a sort of programming analysis. Testcases are rerun at time of new release to check the features are implemented as per our specification and flow is not breaking anywhere due to new changes and new changes has not conveyed any bugs. When there is huge change in the current flow code functionality, regression analysis is required. It is method of re-checking the unaltered code is executing properly along with the new changes Fig 15 : Regression Analysis 5.1 Regression Analysis is required to be done at following scenarios:- ? New features are added to flow. ? When Changes required. ? When the defect fixed or environment changes 23 5.2 Regression Analysis Execution The requirement for regression analysis comes when new features (enhancements), bug fixes, reduce the execution time and evacuation of current functionality occurs in flow code. These changes may affect the current functionality of the system. Regression Analysis becomes compulsory in that situation. Regression Analysis can be executed using the following steps: Fig 16 : Regression Testing Plan 1. Re-test All: Re-Test is one of the ways to do deal with relapse testing. In this methodology, all the experiment suits ought be re-run. Here we can characterize re-test as when a test comes up short, and we decide the reason of the failure is a code issue. The issue is accounted for, we can expect a another form of the product wherein imperfection fixed. For this situation, we should execute the test againto affirm that the issue is fixed. Some will allude to this as affirmation testing.The re-test is very pricey, as it needs assets and time. 2. Regression Analysis Selection: In this startergy, a selected experiment suit will run not the whole experiment suit directory. The selected suits are categorized as follows Reusable Test cases. Obsolete Test cases. 24 5.3 Regression Analysis Flow Fig 17 : Regression Testcase Flow ? First setup the environment variables and enter the project setup. ? Check the input configuration by giving the proper value. If the config variable is behaving properly then set the exit flag variable as zero else one. ? Check the input configuration by giving the empty value. If the config variable is behaving properly then set the exit flag variable as zero else one. ? Check the input configuration by giving the invalid value. If the config variable is behaving properly then set the exit flag variable as zero else one. ? Now check the status of the exit flag if it is one at end of all scenarios the testcase will fail and if it is zero testcase will pass. 5.4 Advantages of Regression Analysis ? With help of regression analysis the quality of product got improved ? We can get confidence that new enhancement donot hinder our previous functionality and implemented properly. ? We can automate the regression to run in parallel which will reduce the execution time. 25 Chapter 6 Conclusion and Future Scope The basic agenda of this thesis is to understand the available methods of execution of static check tools used in front end domain. From the results of the manual execution of these tools on local machine, there is noticeable reduction in execution time. Current mode of work is to manually run these tools on each IP on single local machine in sequential manner. It requires human efforts and consume more time. By implementing automation flow, less efforts are required, and time is saved. We have successfully automated the front end tools flow after doing the study of the front end tools. Perl script has been written for the automated flow. The new features we are going to add in this script . We can proceed for regression testing of these script in the near future and also enhance the features depending on request of different teams and also provide deployment services for these scripts. 26 References [1] E.Cummings, C. (2008). Clock Domain Crosssing (CDC Design & Verification Techniques) using System Verilog. SNUG. [2] Karimi, N. (septemeber 2013). Detection, Diagnosis and Recovery from Clock Domain Crossing Failures in Multiclock SoCs. IEEE transcation on computer-aided design of Integrated circuits and systems, (p. Vol 32 no 9). [3] Kim, H.-K. (2013). Testing of synchronizers in asynchronous FIFO. IEEE Trans., (pp. 49-72). [4] Liu, D. (1994). Power Consumption Estimation in CMOS VLSI chips. IEEE Journal, pp 663-670. [5] Nemanu, M. (1999). High-level area and power estimation of VLSI circuits. IEEE Trans.Computer- Aided Design, (pp. 697-713). [6] Verma, S. (n.d.). Understanding clock domain crossing issues. Atrenta. [7] https://alison.com/course/fundamentals-of-perl-programming [8] https://learn.perl.org/ [9] https://www.learn-perl.org/en/ 27