TESTCHIP DESIGN AND VERIFICATION AUTOMATION

Major Project Report

Submitted In Partial Fulfillment of the Requirements for the Degree of

Master of Technology
in
Electronics & Communication Engineering
(Embedded Systems)

By

BHAUMIKKUMAR PATEL

(14MECE10)

Electronics & Communication Engineering Branch
Electrical Engineering Department
Institute of Technology
Nirma University
Ahmedabad - 382481

May-2016

TESTCHIP DESIGN AND VERIFICATION AUTOMATION

Major Project Report

Submitted In Partial Fulfillment of the Requirements for the Degree of

Master of Technology
in
Electronics & Communication Engineering
(Embedded Systems)

by

BHAUMIK PATEL
(14MECE10)

External Project Guide:

Mr. RAJESHKUMAR IMMADI
Project Manager,
ST Microelectronics Pvt Ltd,
Greater Noida

Internal Project Guide:

Prof. Y. N. TRIVEDI
Professor,
Institute of Technology,
Nirma University

Electronics & Communication Engineering Branch
Electrical Engineering Department
Institute of Technology
Nirma University
Ahmedabad - 382481
May-2016

Declaration

This is to certify that

1. The Project Report comprises of my original work towards the degree of Master of Technology in Embedded System 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.

Bhaumik H Patel

CERTIFICATE

This is to certify that the Project Report entitled **TESTCHIP DESIGN AND VERIFICATION AUTOMATION** submitted by **BHAUMIKKUMAR PATEL (14MECE10)**, towards the partial fulfillment of the requirements for the degree of **Master of Technology** in **"Embedded Systems", Nirma University, Ahmadabad** is the record of work carried out by his under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of our knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Date: Place: Ahmedabad

Dr. Y. N. Trivedi
Internal Guide

Dr. N. P. Gajjar
Program Coordinator

Dr. D. K. Kothari
Section Head, EC

Dr. P. N. Tekwani
Head of EE Dept.

Dr. P. N. Tekwani
Director, IT

CERTIFICATE

This is to certify that the Project Report entitled **TESTCHIP DESIGN AND VERIFICATION AUTOMATION** submitted by **BHAUMIKKUMAR PATEL (14MECE10)**, towards the partial fulfillment of the requirements for the degree of **Master of Technology** in **Embedded Systems, Nirma University, Ahmadabad** is the record of work carried out by his under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.

Mr. Rajeshkumar Immadi
Project Manager
ST Microelectronics Pvt Ltd
Greater Noida

Disclaimer

“The content of this thesis does not represent the technology, opinions, beliefs, or positions of ST Microelectronics Pvt. Ltd., its employees, vendors, customers, or associates.”

Acknowledgments

I would like to express my gratitude and sincere thanks to **Dr. D.K.Kothari**, Head of EC Department, and Dr. **N.P.Gajjar**, PG Coordinator of M.Tech Embedded Systems program for allowing me to undertake this thesis work and for his guidelines during the review process.

I take this opportunity to express my profound gratitude and deep regards to **Prof. Yogesh N. Trivedi**, guide of my major project for his exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessing, help and guidance given by him time to time shall carry me a long way in the journey of life on which I am about to embark.

I would take this opportunity to express a deep sense of gratitude to Project Manager **Mr. Balwant singh**, Senior Project Manager , ST Microelectronics Pvt Ltd. for his cordial support, constant supervision as well as for providing valuable information regarding the project and guidance, which helped me in completing this task through various stages.

I would also thank to **Mr. Rajeshkumar Immadi** Our project manager, my Project Mentor **Mr. Kapil Juneja** and **Nilanjan Guha** for always helping, give good suggestions and solving my doubts and guide me to complete my project in better way.

Lastly, I thank almighty, my parents and friends for their constant encouragement without which this assignment would not be possible.

Bhaumikkumar H PATEL
(14MECE10)

Abstract

Purpose of TestChip is to verify the invented Technology and IPs added in System On Chip(Soc) which implementation in silicon wafer. It includes all the IPs which are to be used for Researched Technology or System On Chip(SOC) applications. TestChip provides post silicon validation details included Library of Memory,RO,Standard-Cell,Analog or any other IP. Before any SOC/ASIC Mass Productions, Test Chip gives us better silicon parameters and testability.

Test Patterns are similar to test vectors except these are cycle based patterns. Patterns for the testing are deliver to the Fabrication after the tape-out. These Patterns are almost same except which IP are being tested so that to make these patterns automatically generated. Fulfillments of this tool gives better quality of test patterns,accurate and less time consuming to generate. It also helps in quick verification of RTL/netlist of any block or while chip.

RTL Generation is Generating Register Transfer Logic(RTL) of the multiple block in less time period. To make Register Transfer Logic(RTL) of the BLOCK automatically generated. They already have one automation for this but its using XML language for generate the Register Transfer Logic(RTL). When tool crash because of the lack input and any other reason its hard to debuge by designer. So develop one tool which is using tcl language for generating Register Transfer Logic(RTL). This thesis presents a test chip methodology and automatic generation of test pattern and Register Transfer Logic(RTL) to help achieve fundamental goals.

Contents

Certificate **i**

Declaration **ii**

Cerificate **iii**

Cerificate **iv**

Disclaimer **v**

Acknowledgments **vi**

Abstract **vii**

1 Testchip Introduction **1**

1.1 **Testchip:- Concept** 1

1.2 **Testchip Design Flow** 2

1.3 **Expectations of Testchip** 3

1.4 **Cost if bug detects at production time** 4

1.5 **Components added to Testchip** 4

1.6 **Problem Statement and Its Solution** 4

1.7 **Importance of Verification Environment Automation** 5

1.8 **About Project: Test Pattern Automation And RTL Generation Automation** 6

2 Standard cell And Memory IP **7**

2.1 **Standard Cell** 7

2.1.1 **ALLCELL Block** 7

2.1.2 **FDD Block** 8

2.1.3 **Retention Block** 8

2.2 **Memory IP** 8

2.3 **Static RAM** 9

2.3.1 **Single Port Random Access Memory (SPRAM)** 10

2.3.2 **DUAL Port Random Access Memory (DPRAM)** 10

3 TestChip Architecture **12**
3.1 **Purpose Of Testchip** 12
3.2 **Architecture of Testchip** 13
3.2.1 **TOP LEVEL** 13
3.3 **Standard cells** 15
3.3.1 **ALLCELL TOP** 17
3.3.2 **ALLCELL** 19
3.3.3 **FDD** 21
3.3.4 **RETENTION BLOCK** 23

4 Auto Test Pattern Generation Tool **24**
4.1 **Test Patterns of Standard Cell** 24
4.1.1 **Flow of Test Patterns** 26
4.2 **Need of Automation Tool** 27
4.2.1 **Problem Statement and Its Solution** 27
4.3 **Tool** 27
4.3.1 **Design sheet information:-** 28
4.3.2 **LIB information sheet:-** 29
4.3.3 **Feature of Tool** 30

5 RTL Generation Tool **31**
5.1 **Need of Automation Tool** 31
5.1.1 **Problem Statement and Its Solution** 31
5.2 **Tool** 32
5.2.1 **Feature of Tool** 32

6 Work Contribution and Result **34**
6.1 **Validation Flow** 34
6.2 **Simulation Graphs** 35
6.2.1 **SCAN 0, SCAN 1 and SCAN ALL** 35
6.2.2 **BYPASS MUX** 36
6.2.3 **BYPASS SCAN** 37
6.2.4 **FUNC SCAN AND FUNC MUX** 38

7 Tools **41**
7.1 **Incisive unified simulator** 41
7.2 **Shell Scripting** 42

Conclusion **43**

References **44**

List of Figures

1.1 Testchip Design Flow . . . 3

2.1 FDD Block Pin . . . 8
2.2 SINGLE PORT SRAM . . . 10
2.3 DUAL PORT RAM TYPE - 1 . . . 11
2.4 DUAL PORT RAM TYPE - 1 . . . 11

3.1 Maturity flow of Library and IP . . . 13
3.2 Testchip Architecture Block . . . 14
3.3 STD different level of heirarchy . . . 15
3.4 Top hierarchy . . . 16
3.5 ALLCELL top . . . 16
3.6 Input Block . . . 17
3.7 GP1 pinmap . . . 18
3.8 COMBO . . . 20
3.9 TRI . . . 21
3.10 FDD MUX . . . 22

4.1 Tool flow . . . 26
4.2 Tool structure . . . 28

6.1 Validation Flow . . . 34
6.2 ref chain flop for scan0 . . . 35
6.3 scan0 . . . 35
6.4 scan1 . . . 36
6.6 ref cahin bypass mux . . . 36
6.5 scan all . . . 37
6.7 bypasss mux . . . 37
6.8 bypasss mux . . . 38
6.9 bypass scan . . . 38
6.10 ref cahin func mux . . . 39
6.11 func mux . . . 39
6.12 ref cahin func scan . . . 40

6.13 func scan 40

List of Tables

7.1 NUC Command 41

Chapter 1

Testchip Introduction

RTL is large architecture of low level constructs called standard cell. This architecture is taken from standard-cell library consisting of pre-characterized collections of gates. Standard cell is typically particular to the planned manufacturer of the ASIC or System on Chip.

Standard Cell is fundamental cell which are used frequently in design of chip (e.g. OR, AND, XOR, XNOT, Inverter). It includes Latch and Flip ops types of storage elements. Common usage function are already includes in standard cells and used directly from standard cells library like element of XOR, XNOT, NAND, NOR. This library is specific to particular technology.

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.

The SoC containing test-structures for memories, mixed IPs, standard cells and sensors is used as test chip for post silicon validation. It provides assurance that functionality of IPs, standard cells and other components are correct and its maturity is high enough for the mass productivity.

1.1 Testchip:- Concept

There is large financial and time loss if any bugs or functional defect captured in new SOC/ASIC production. To resolve where Errors are occurred after packaging or at production level is very time consuming and might be unresolved for production level which suffers financial loss as well as takes time.

Testchip is way to mature latest technology to verify its functionality of different IP blocks and implementation of the logic on to the physical silicon chip. It makes good view in term of fabrication process of latest technologies or any SOC's Testability, Functionality

and Reliability.

Testchip also includes extended Testability of the Chip itself so that time to market consumes less and need inexpensive tester. It has very on-chip pattern generator and testing algorithms for memories. It ensures the maturity of the invented technology on silicon and finds out the fabrication defect for same.

Advantage

- It provides the best design flexibility for upcoming production of the chip.
- High performance and High density by resolving the problem which occurred at the Testchip.
- Highlighting CAD vs Silicon results differences of different IPs
- Effort to make Low Power Consumption for Latest Technology.
- Implement the Best Test Environment for the Market Deliver Chips.
- Gives Less Financial loss if Technology has any fabrication incompatibility.

Limitation

- Silicon wastage because testchip is not in any SOC applications so testchip targets to test as more as different IP in silicon area.

1.2 Testchip Design Flow

Technology development now not only driven to improve the circuit speed and density, but also concentrating on reducing the power consumption. Power is emerging as the most critical issue in system-on-chip design today so scaling would be mandatory and smaller dimension circuit have been preferred which lesser and lesser nm technology is chosen. It is important to have detailed understanding of the power consumption behavior of a chip.

The IC design process starts with a given set of requirements. After the development, this initial design is tested against the initial design requirements. When these requirements are not satisfied, the design must be improved. If such improvement is either not possible or too costly, then the requirements must be revised and the IC design process re-starts with the new modified requirements.

The failure to properly verify a design in its early phases typically causes significant and expensive re-design at a later stage, which ultimately increases the time to market. Thus, the verification of the design plays a very important role in every step. Fig.1.1 provides a view of the Very large scale integration (VLSI) design flow based on schematic capture systems. Although top-down design flow provides an excellent design process control, in reality, there is no truly unidirectional top down design flow. Both top-down and bottom-up

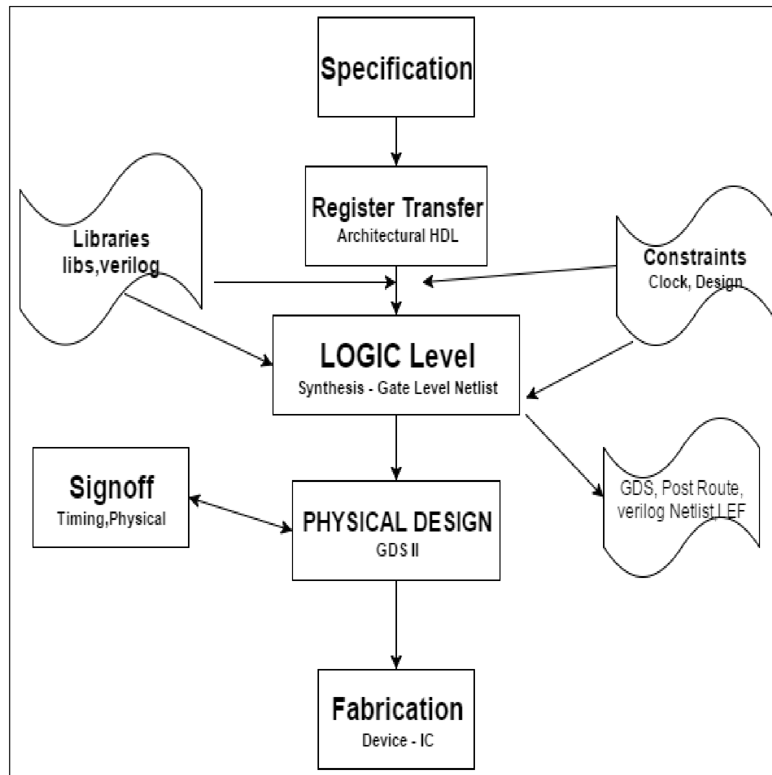

Figure 1.1: Testchip Design Flow

approaches have to be combined. For instance, if a chip designer defined architecture without close estimation of the corresponding chip area, then it is very likely that the resulting chip layout will exceed the area limit of the available technology. In such a case, in order to fit the architecture into the allowable chip area, some functions may have to be removed and the design process must be repeated. Thus, it is very important to feed forward low-level information to higher levels (bottom up) as early as possible.

1.3 Expectations of Testchip

As stated above that Testchip is chip to target testing of different IP in silicon wafer before any mass productions in SOC Integration/any new technology to set up for production. Testchip design flow helps to design flow of chip mass productions because of new technology cells and library is not mature at production level. The cycle time of the Testchip has to be tight so that necessary action can be taken to improve the IP or flow before the actual product out to the market. Design flow of testchip schedules from 1 to 4/5 months depending on the complexity of testchip.

1.4 Cost if bug detects at production time

SOC fabricates for mass production without made testchip and on silicon validation specification. Large Financial loss has to be suffered for company if any functional errors or bugs occur. It might be solved sometimes any minor errors at packaging level like any output have attenuated logic level.

The Pentium FDIV bug was a bug in the Intel P5 Pentium floating point unit (FPU). Because of the bug, the processor would return incorrect results for many calculations used in math and science. The error was rarely encountered by users. Intel were criticized very heavily and ultimately recalled the defective processors.

1.5 Components added to Testchip

- Memory IP
- Standard Cell
- Analog Mixed IP

1.6 Problem Statement and Its Solution

Test chip utilize before usage of any of IPs in SOC application, it would be tested first in silicon chip and its functionality is verified for physical silicon. Test chip is used for detailed information of the post silicon validation and used for better implement logic to the physical silicon chips. Therefore, SOC integration starts after testify with test chip. There is any delay in test chip development as SOC will have to be delayed. There is challenge to reduce time for test chip. It reduce time to market SOC or ASIC if it may reduce the time for test chip.

Any SOC/testchip reduce time to market by how fast verification is performed so reduce time to verified the Design of Test Chip is very important. Design of testchip is almost reached at least time. Challenge is be to verified testchip so that we can ensure that SOC might have been right functionality except any fabrication defects.

By Introducing AUTO TEST PATTERN GENERATION TOOL, It generates the test patterns automatically by giving the Design mapping Document of IP and Design activation IP Documents. Design mapping Document has the mapping information that which top pins related to the IP pins at the low level hierarchy and their execution cycle file values of top pins. Design activation IP Document has its block select value which used for active particular block and its down hierarchy block called as CUT where actual IPs and their supported driven blocks. Design mapping Document also has input data bus width and address bits.

Basically Test chip have different LIB cell IP. Every time test patterns have to written according to the Test chip. Now After developing AUTO TEST PATTERN GENERATION TOOL, Patterns are related to LIB not TEST CHIP so it would be easy to maintain patterns algorithm temple according to LIB because LIB cell IP are almost repeatedly in all TEST CHIP.

Tool makes the Patterns to not relate to Specific Test chip because the top of TEST CHIP always being changed, not Standard cell LIB IPs. Design Mapping Document of Specific Test chip is provided by the Test Chip Developers which are used to generate the mapping files in the patterns and other files and algorithms are always being generic to Standard cell LIB IP. Tool has the list of the patterns of algorithms and list of LIB information those are being likely to generate.

For RTL generate of the standard cell library they already have one tool for generate the RTL for Block. But on depending upon the changes include in .lib(synopsys standard format) file its make difficult recognize type of cell and segrate the cell information. Because all Tool used XML language for the generating RTL. As some changes in .lib file its also reflect that we have to change the tool respective. as language used is not as easy to learn which lead to include one automation engineer to maintain that tool. Also our team Testing standard cell which not include same strategy for testing. So they need to include that feature for future development if customer ask for that.

So I had developed tool which take input like some standard file (.lib and .db). and use the same tool(Design Compiler and Prime time) which can read that type of file and use language which is easy to understand by designer(tcl and shell). As some changes include as tool development (Design compiler and Prime time) its will not reflect for identify the cell type and its pin information.

As they are testing the different library like SVT, LVT. Those type of library we can not all combine. Also there are some different SPEC for testing the cell as per requirement we combine some library make one block for that cell.

RTL Generation Tool generate RTL of Standard Cell Test Chip automatically by giving library file (.lib and .db) for which we want to generate the RTL for Testing. Design document has library information with some reference library and reference cell like AND, OR, XOR, MUX, FLOP, CLOCK GATING CELL and BUSKEEPER cell. Design document has information about generate the RTL of the different block like for which library we want to combine the library make one block for testing.

1.7 Importance of Verification Environment Automation

As Testchip design flow is very tight and short, from Design, implementation, verification and sign-off complete very quick. As Verification is very time consuming depending on the

complexity of design. Verification would be made quickly if test patterns (test bench) for designs are created automatically. This type of automation is very time saving and effective for verification. Also, Quality of Verification is very much improved.

1.8 About Project: Test Pattern Automation And RTL Generation Automation

Testing semiconductor cell is increasingly important today because of the high density of current cell chips. In this thesis report we investigate on the various functional fault models present for todays cell technology and discuss about the ways and means to detect these faults. This tool generate scan pattern for any lib. This pattern test on testchip which is design for any particular LIB.

Writing RTL of the Standard cell Test Chip manually takes too much time and design cycle of any IP they cann't make. So its require to make its automated for writing RTL of design. if there is some change in spec after create RTL manually which will reflect in the making whole design new. This will not help for testing any IP so its necessary to write RTL of design automatically. I proposed with one solution to my sir that we can use tool like Design compiler and Prime time which read standard cell file which contain information about cell delay and functionality. if they standard cell team going to use different type of the version for creating .lib and .db file according to that version we use same for separate the cell which cell is corresponding to which category. For this is use some design compiler command for it. After separation of cell i write of RTL of testchip using C shell language.

Chapter 2

Standard cell And Memory IP

2.1 Standard Cell

Standard Cell is fundamental cell which are used frequently in design of chip (e.g. OR, AND, XOR, XNOT, Inverter). It includes Latch and Flip ops types of storage elements. Common usage function are already includes in standard cells and used directly from standard cells library like element of XOR, XNOT, NAND, NOR. This library is specific to particular technology.

Test chip contains the whole standard cells library in post silicon validation. A standard cell library is a consolidated data used in designing a SOC (system on chip). It is a collection of low level logic functions such as AND, OR, INVERTER, flip-flops, latches and buffers. These cells are realized as fixed height variable width full custom cells. The key aspect with these libraries is that they are of a fixed height, which enables them to be placed in rows, easing the process of automated digital layout. The cells are typically optimized full custom layouts, which minimize delay and area. Full-custom design is no longer feasible as Complexity of the design is continuously increasing. Standard cell contains layouts and power calculations.

To meet these test specifications, Presently have 3 kinds of STD cell block architectures:

- ALLCELL Block
- FDD Block
- Retention Block

2.1.1 ALLCELL Block

There can be several groups present at the top, which will constitute to groups G1, G2 etc. Designer can decide total number of libraries in a group based on some rules. The output of these groups which is DATAOUT is given to Output Block. There is muxing of

logic present in the Output Block which selects the suitable group output. The structure of Output block is not fixed and it depends on the number of libraries present in each group. Thus the final ALLCELL Top block will be giving an output which is DOUT.

2.1.2 FDD Block

FDD block was originally used for flops that have dual edge triggering capability. Now this block is also used for flops with some other special features and critical than the normal flop. Library is instantiated huge no. of times (1024) on a testchip.

The dual-edge triggered flip-flop exhibits low delay and small power consumption because the clocking is done at half the clock frequency as compared to single-edge triggered storage element. For these and some other special flops, there is requirement of large no. of instantiation. Also, the library is very small (24 -36).Hence, we cannot use SEQ block of these cells.

In FDD architecture the whole library is present in the place of CUT of ALLCELL block as shown in fig 2.1. Thus a whole library shares a Ref cell which takes the muxed output from the library. This block is then repeated as per the requirements.

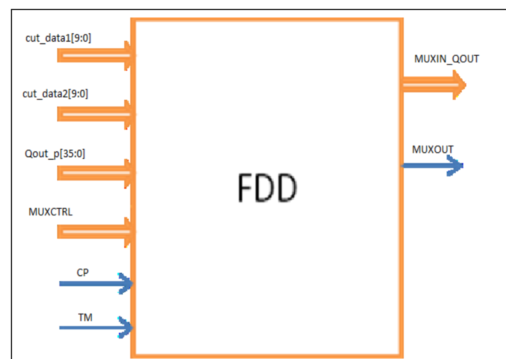

Figure 2.1: FDD Block Pin

2.1.3 Retention Block

Retention block is used to test Retention flops which retain their data.

Architecture and connectivity wise this block is just same as FDD block. Separate block other than FDD is used for these flops because the SLEEP pins of these flops have to be routed through always on cells.

2.2 Memory IP

SOC integration is not possible of without any memory component. Memory IP is very vital part of SOC and any reason of failure of memory IP result SOC application failure. Memory is being different types based on SOC application.

Semiconductor Memories are classified according to the type of datastorage and the type of data access mechanism into the following two main groups

- Non-volatile Memory (NVM) also known as Read-Only Memory(ROM) which retains information when the power supply voltage is off. With respect to the data storage mechanism NVM are divided into the following groups:
  - Mask programmed ROM. The required contents of the memory is programmed during fabrication,
  - Erasable PROM (EPROM). Data is stored as a charge on an isolated gate capacitor (floating gate). Data is removed by exposing the PROM to the ultraviolet light.
  - Electrically Erasable PROM (EEPROM) also known as Flash Memory. It is also base on the concept of the floating gate. The contents can be re-programmed by applying a suitable voltages to the EEPROM pins. The Flash Memories are very important data storage devices for mobile applications.
- Read/Write (R/W) memory, also known as Random Access Memory (RAM). From the point of view of the data storage mechanism RAM are divided into two main groups:
  - **SRAM (Static Random Access Memory)**
  - **DRAM (Dynamic Random Access Memory)**

Our Targets for test chip are SRAM and ROM memory. SRAM used as cache memory in micro-controllers , like the primary caches in powerful microprocessors, such as the x86 family, and many others,to store the registers and parts of the state-machines used in some microprocessors on application specific ICs, or ASICs (usually in the order of kilobytes).ROM is used extensively in graphic cards, hard disks, DVD drives, TFT screens.

2.3 Static RAM

The memory cell is a 6 transistor circuit which is a flip flop comprising two cross-coupled inverters and two access transistors, the access transistors turn on when the word line is selected (high) and its voltage rises to Vdd, and they connect the flip flop to the bit lines. Sizing of the transistors in the memory cells is very important especially for speed and chip cost.

The sense amplifier is important in the total performance of the SRAM chip since the sense delay time directly affects the access time. Sense amplifier is used to sense the small changes in voltage that results when a particular cell is switched onto the bit line. One stage differential pair of sense amplifier is utilized here. The sense amplifier circuit is controlled by a clock signal, which is synchronized with the pre-charging and word-line signals.

- **TYPE OF SRAM:-**
  - **Single Port static Random Access Memory (SPRAM)**
  - **Dual Port static Random Access Memory (DPRAM)**

2.3.1 Single Port Random Access Memory (SPRAM)

Single Port SRAM is simple SRAM contains single port to used for read and write operation as shown in fig 2.2. Clock pin , read and write operation are perform with its logic, is positive edge trigger. Every Posedge reflects the Input data bus to memory with address bus and respective of read or write signal.

- Memory has logic low control pin of CSN and WEN.

- WEN - Write Enable logic low Write when Logic low otherwise read.

- CSN - Chip select logic low Chip selected Write when Logic low otherwise disable.

- Memory has SLEEP which is used to switch off peripheral. Memory also has test mode,scan chain pins,special testing pins.

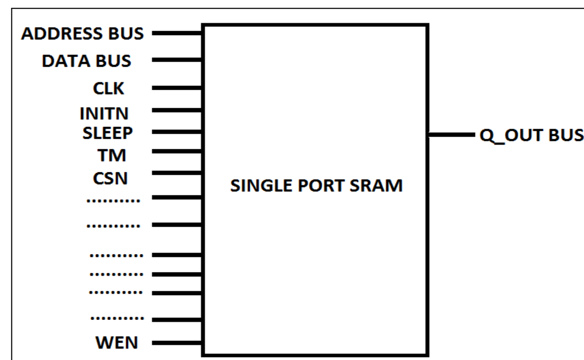

Figure 2.2: SINGLE PORT SRAM

2.3.2 DUAL Port Random Access Memory (DPRAM)

DUAL Port SRAM gives advantage of read and write perform simultaneously by adding extra data bus, address bus and extra control signal.In some application like Video RAM , Elevators to Robot Control, Commercial Aircrafts to Unmanned Flight Controls, Surveillance cameras to Night vision systems.Dual Port Memory has increased bandwidth approximately 2x the speed of a similar single port RAM.

DUAL Port Random Access Memory TYPE-1

This type of dual port memory has only one data port though it contains read and write different address bus as well as different chip selection bit as shown in fig 2.3. It has one write enable signal common for both.

When any address

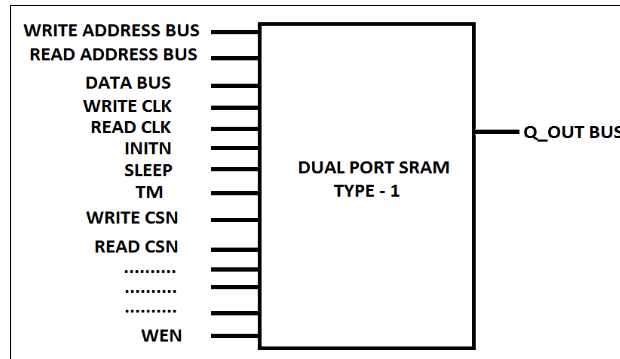

Figure 2.3: DUAL PORT RAM TYPE - 1

DUAL Port Random Access Memory TYPE-1

This type of dual port memory has two data port though it contains read and write different address bus as well as different chip selection bit. It also contains different write enable signal which gives permission to read and write at a time.

Memory has two data inputs port , two cheap selection enable , 2 write enable , 2 Address Bus as well as 2 Output Data ports. It is same like two SPRAM (Single Port SRAM) are going to attached together.

DUAL PORT SRAM port is independent of each other. Both port have their own WEN and CSN signals as well as their operating frequency.

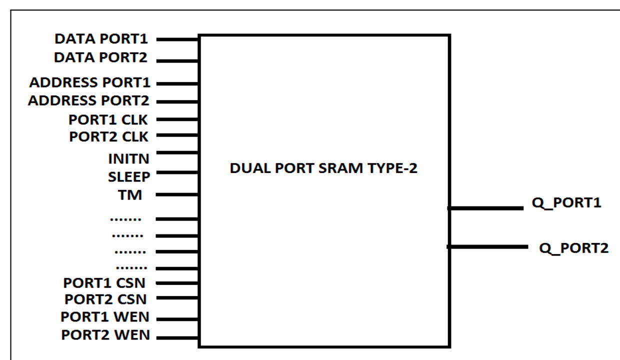

Figure 2.4: DUAL PORT RAM TYPE - 1

Chapter 3

TestChip Architecture

3.1 Purpose Of Testchip

Test chip is single chip contains different type of IP blocks which includes for silicon qualification and functionality test. It has RAM and ROM type memory includes single, dual port RAM. It has Memory-BIST and BISC for easy testing Methodology.

BIST has in-built Memory testing algorithms which enable by the registers performs the memory testing algorithms to all Memory IPs. Then output gives the bad and repair signal for that particular memory.

Test chip is special purpose silicon Qualification chip which performs too many testing on implemented blocks. It has tested with different Power, voltage and temperature. It has radiation test, life time test which ensures how to improved yield of our product. As much as CAD level data and actual data correlated that let yield and productivity of chip goes to high.

The list of blocks which includes in Test chip are as below

- Memory Blocks
- PLL Blocks
- Standard cell Blocks and RO Chain Blocks

Testchip is needed to give post silicon validation parameter to our customer. Testchip is testify that our IP, standard cell Library, Analog and mixed IPs performs normal expected functionality. Ideally, Testchip should be made first before any Maturity level of any technology library or SOC Integration but Practically TestChip will be made parallel with SOC integration. Test chips testing result and post silicon data comes before SOC final implementation. SOC modification does if TestChip results reflect it.

Flow of Maturity of Library is shown in fig 3.1 below :-

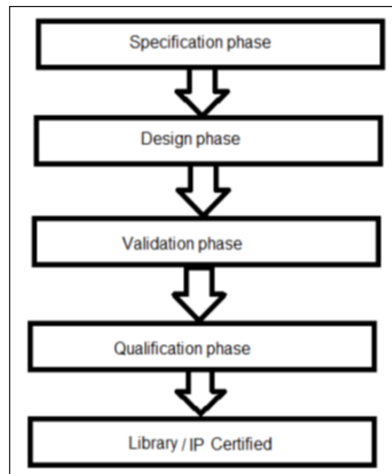

Figure 3.1: Maturity flow of Library and IP

- First step: - Define a new technology platform. There is no specification at this time and no library component. It takes list of component from old library.
- Second step: - Next, the specification is set and it includes all component of new technology platform (process, libraries, CAD tools) After Design is completed with help of library cells and components. Layouts and characterizing performed at this level. Some other components might be defined later also.
- Third step: - This level, the design are validated on the silicon. Here the Functionality and characterizing of library by testchip.
- Fourth step: - At last level, Testability is defined very well. It makes easy to test and must take less time. Yield and productivity must increase of chip. Library and IP is certified at this step with post silicon data.

Testchip is full custom chip from specification to tape-out as well as fully testable so that TestChip contains some of features give below which used accurate and speedy design generation. Automation of Design also can be used if these feature available.

- Repetitive design across Memblock
- Design Re-use and standardization
- Reduction of manual effort

3.2 Architecture of Testchip

3.2.1 TOP LEVEL

TestChip Architecture is shown as in fig 3.2 above with TOP Level hierarchy. Their functionality are given below

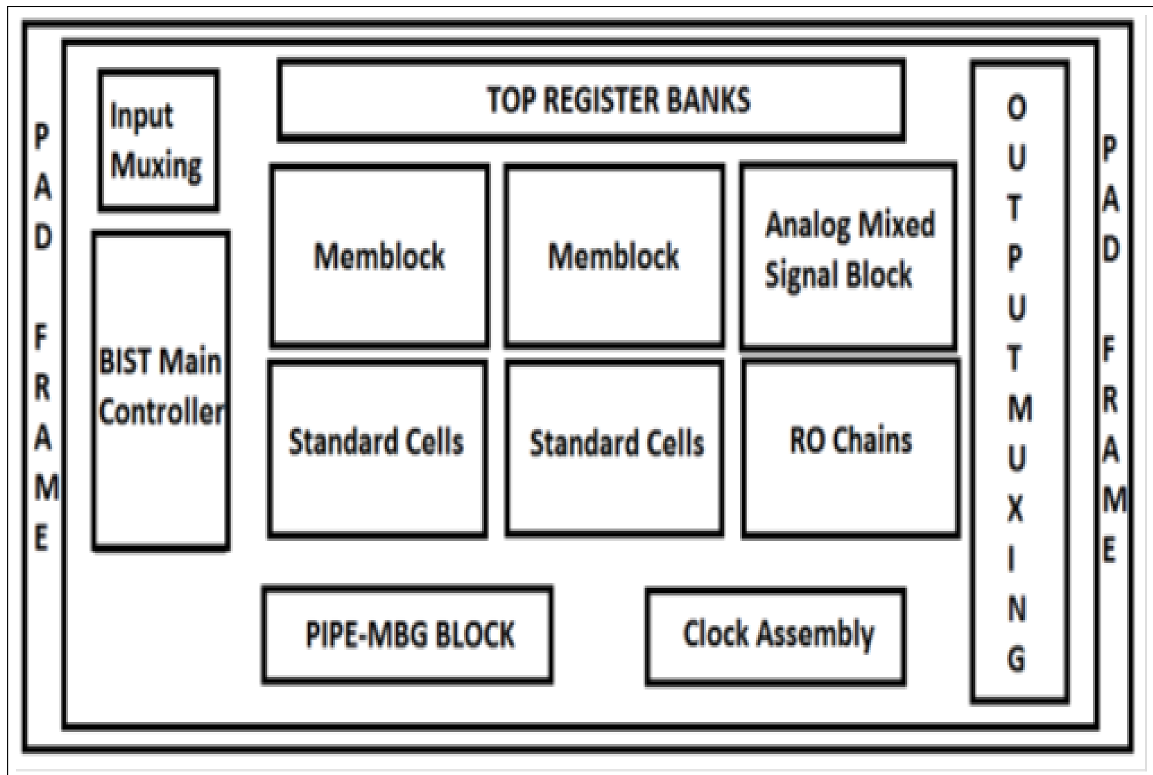

Figure 3.2: Testchip Architecture Block

- Input Muxing :- Used to take input from external world passing to downwards
- TOP Register Banks :- Register Bank has particular enable pins and IP selection pins of Memblocks and any other blocks
- CLK Assembly :- This block is useful for mapping which clock is used specific memblock and IP.
- BIST Central Controller :- BIST Controller controls all the signals which carried to BIST Wrapper around every Memory IP.
- Output Muxing :- This block is responsible to carried away output bus to the padframe.
- PIPE-MBG :- Pipling used for operating faster clock ans MBG(Macro Backgroung Logic) is used for generation of different type input combination.
- IG-Gating :- Memory has special IG pin used for blocking input data to the memory pins when enable.
- Output Muxing :- This block is responsible to carried away output bus to the padframe.

From TOP of TestChip, there are many blocks and blocks are divided as per design. For any selection of Blocks, BLOCKSEL [N-1:0] Register is placed in TOP Level of Testchip, where N is maximum block number. How to select BLOCKSEL is described after this section. For more testability, boundary scan is added to the top level input and output pins.

3.3 Standard cells

Below figure 3.3 show different hierarchical level of Standard cell architecture.

Top level consists of the input block, output block and number of the Group G1, G2 etc. Designer can decide total number of libraries in a group based on some rules.

H2 hierarchy is further sub divide of the different group G1, G2 etc. it consists G1_R0 G1_CTRL G1_R90 block.

H3 hierarchy is sub divide of the G1_R0 and G1_R90 Block. It consist different libraries.

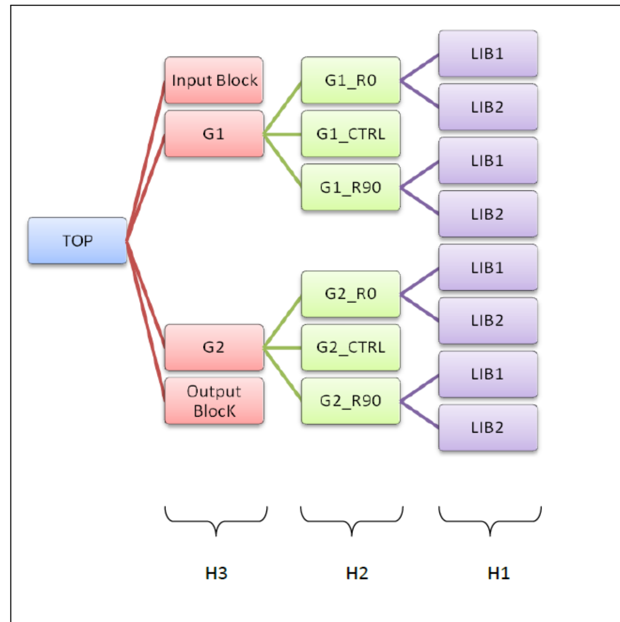

Figure 3.3: STD different level of heirarchy

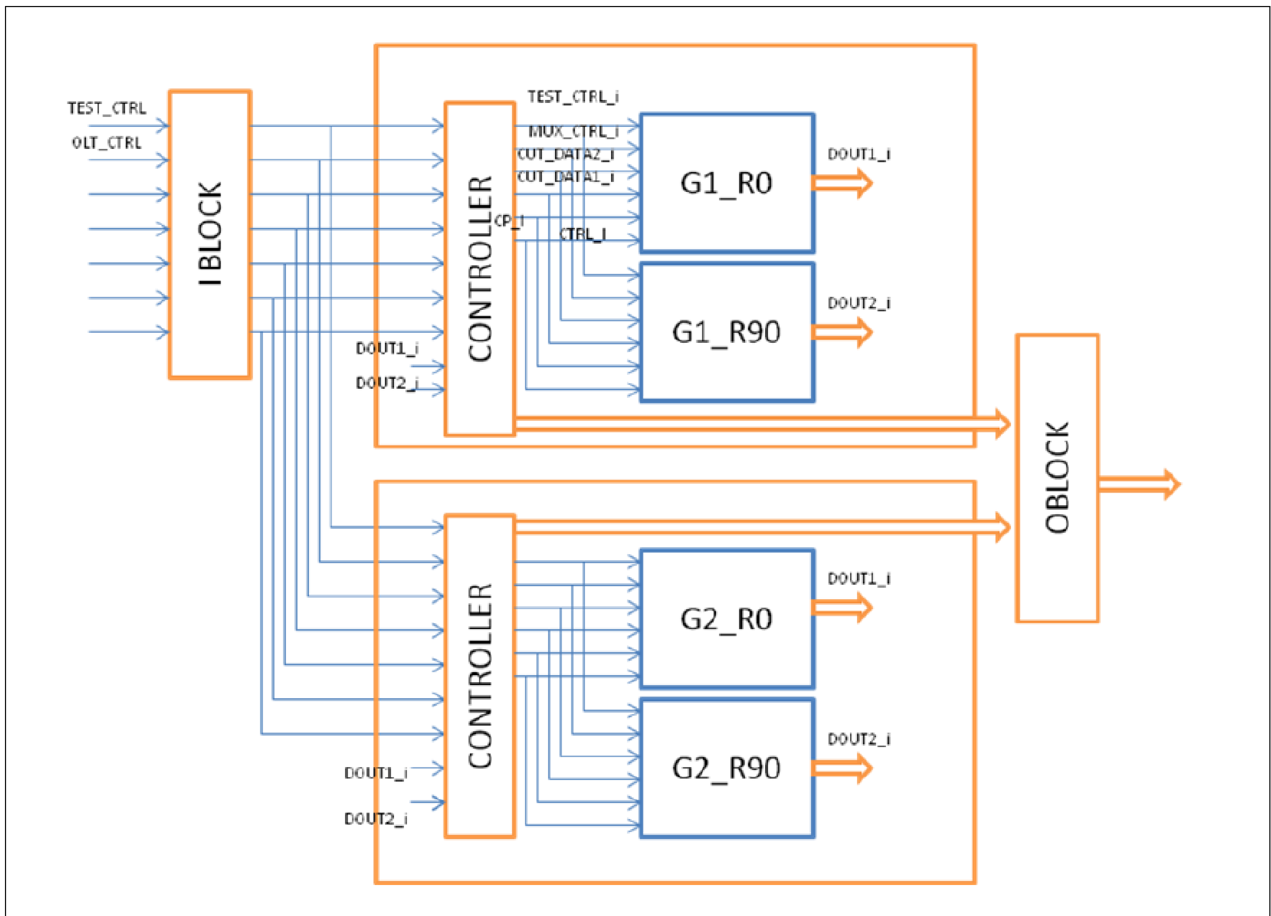

Figure 3.4: Top hierarchy

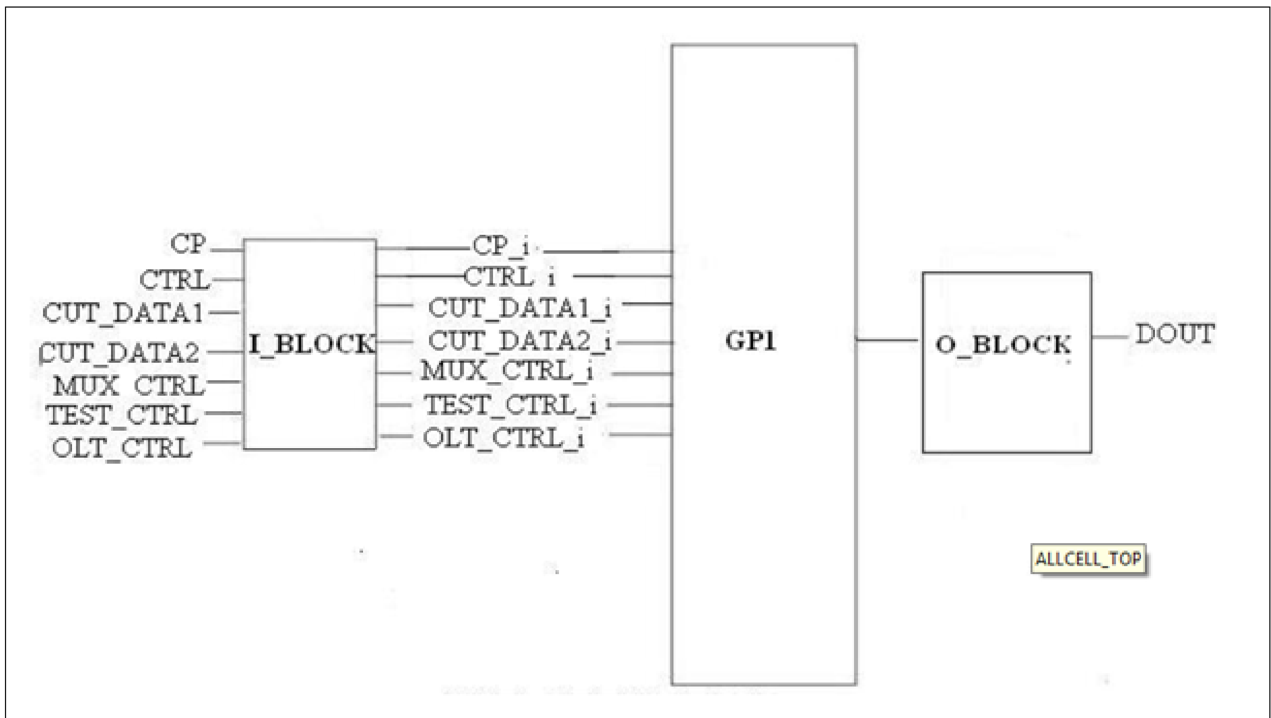

Figure 3.5: ALLCELL top

3.3.1 ALLCELL TOP

There can be several groups present at the top, which will constitute to groups G1, G2 etc. Designer can decide total number of libraries in a group based on some rules. The output of these groups which is DATAOUT is given to Output Block. There is muxing of logic present in the Output Block which selects the suitable group output. The structure of Output block is not fixed and it depends on the number of libraries present in each group. Thus the final ALLCELL Top block will be giving an output which is DOUT.

I_BLOCK

The ALLCELL TOP consists of Input Block, G1, G2 etc. and an Output Block. The Input Block is an interface block which is made up of buffers as shown in fig 3.6. The width of the inputs given to these buffers is as per the specification file provided to the designer. The output of this I Block is given as the input to various groups formed at the TOP.

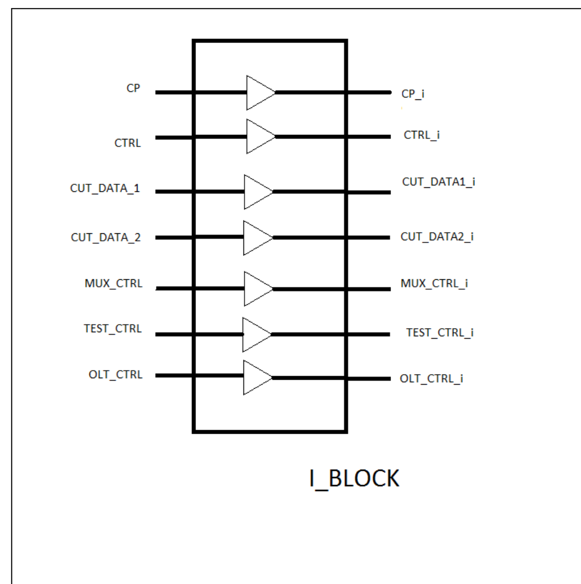

Figure 3.6: Input Block

GP1

Controller which is a FSM (Finite State Machine) along with G1_R0 and G2_R90 together forms a group called G1. This G1 is a group formed at the TOP. There can be several groups formed at the TOP based on the necessity.

The segregation of Libraries in different groups is of special importance. Following are the points that are to be taken care while creating the library groups:

- Separate netlist can be generated for these groups; hence those libraries which require dedicated power supply should be kept in Separate group.

- Libraries which cannot be routed together should be kept in separate groups.
- The no. of library instances in a group is limited to 32(16 each in R0 & R90).

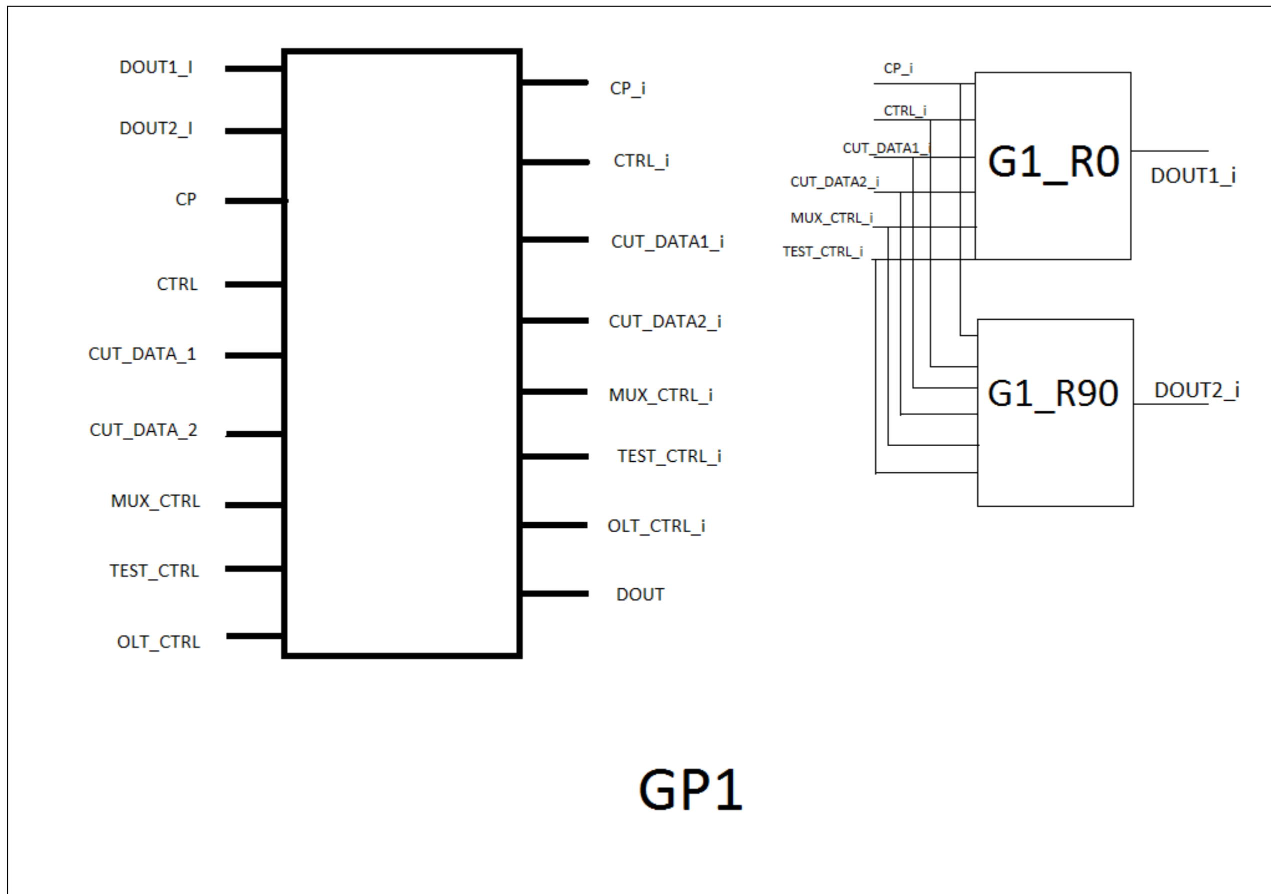

Figure 3.7: GP1 pinmap

G1_R0

One level above (GLOBAL_CELL) for libraries will be an Interface Unit. The Interface Unit will have a corresponding INI and RTL. The INI file contains the information regarding how the exporting is done at the top. The RTL file will be containing a module that defines the various inputs & outputs with their exact bus width as given in the specification.

There can number of libraries present which are termed as Lib1, Lib2, Lib3 etc. and each library can have several instances depending on the requirement (Max. 4). Below diagram shows 2 libraries (Lib1 and Lib2) have 2 instances each. The GP1_R0 (which is group formed at the top) will have a Reference Unit which will be having inputs such as CP, Cut_Data1 and Ctrl. The no. of library instances in a group is limited to 16.

G1_R90

G1_R90 is at same level of hierarchy as G1_R0. The no. of library instances in R90 & R0 orientation are provided in the input.csv.

REFERENCE UNIT

The Reference Unit consists of Reference Flop, Reference Mux, Reference Buffer, Reference and/or gate. The most reliable cell in the library is used as a Reference. The output of the library1 is MUXOUT & QOUT which together forms DOUT1. Thus DOUT1, DOUT2 etc. goes into Output Mux wrapper. The four bits coming from MUXOUT and four bits coming from QOUT that output is given to Output Mux Wrapper. Inside the Output Mux Wrapper there are muxs used along with MUX_CTRL 10th and 11th bit used for select line. This is done so that at a time only 2 bit output comes from the output of two muxs depending on the type of cells.

OLT Controller

OLT controller is a FSM which enables the operation life test [OLT] of STD cells. It contains an FSM which increments Input data going to the CUTs, updates MUXCTRL of the STD cell block libraries, and compares their outputs.

If the output obtained from any of the CUTs is found faulty, it flags a BBAD signal. When outputs have been compared for all cells (by varying MUXCTRL), for all the possible input combinations, BEND signal is generated indicating the end of OLT test (1 loop). This BEND signal is then again lowered to 0, to start another loop of testing. This way, Testing continues till the OLT time.

3.3.2 ALLCELL

The basic cells for ALLCELL are COMBO, SEQ, TRI, and CBUF. The main difference b/w these units are the type of CUTS (cell under test).

COMBO

This contains cells which include AND, OR, INVERTS, BUFFERS etc. and these cells are called as CUT (cell under test). The basic architecture is shown in fig 3.8.

Main components of this unit are:

- Cell under test [CUT]
- Reference cell
- Outmux

These components are connected in the way as shown. Presently all the Cuts share the inputs. Outputs from these cuts are applied to the D0 input of the bypass mux. Bypass mux, as the name suggests, is given a bypass input at D1. Through this input we can test the D-Q path of ref flop to which the output of bypass mux is applied.

Reference cell comprises of a bypass mux and a flop. Ref flops are connected in a scan chain such that, Q output of each flop is connected to TI Input of next flop. The no. of ref flops for each CUT depends upon type of block i.e. [Allcell, FDD]

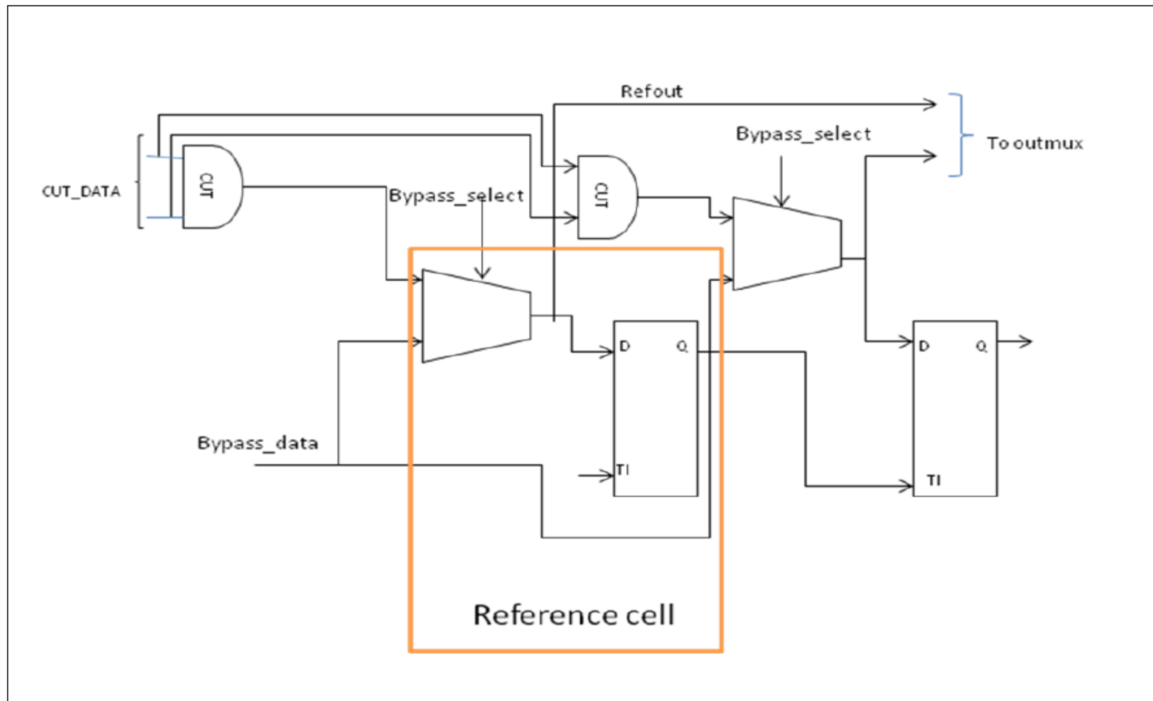

Figure 3.8: COMBO

SEQ

In SEQ the cells are flops. CP¡0¿ from Interface unit is given to these flops as clock input. CP¡1¿ from Interface unit is given to Reference flops as clock inputs. Hence clocks of CUTS and Ref cells are independent from top.

The reset used in SEQ will be present in CUT_DATA2 bus (this bus contains extra pins, special signals). The TI (Test Input) and TE (Test Enable) is present in Test_Ctrl bus. There is also a Reference Cell at the output. Rest of the components and paths in SEQ block are same as in COMBO.

TRI

TRI block contains tri state cell as CUTS. The tri state cell has an enable E and CUT_Data1 as input. The output of the tri cell is given to a bus keeper as well as to a mux of the Reference Cell. Consider if there are two thresholds zero and one then if an intermediate value occurs the bus keeper keeps the previous stored logic. Rest of the components and paths in SEQ block are same as in COMBO.

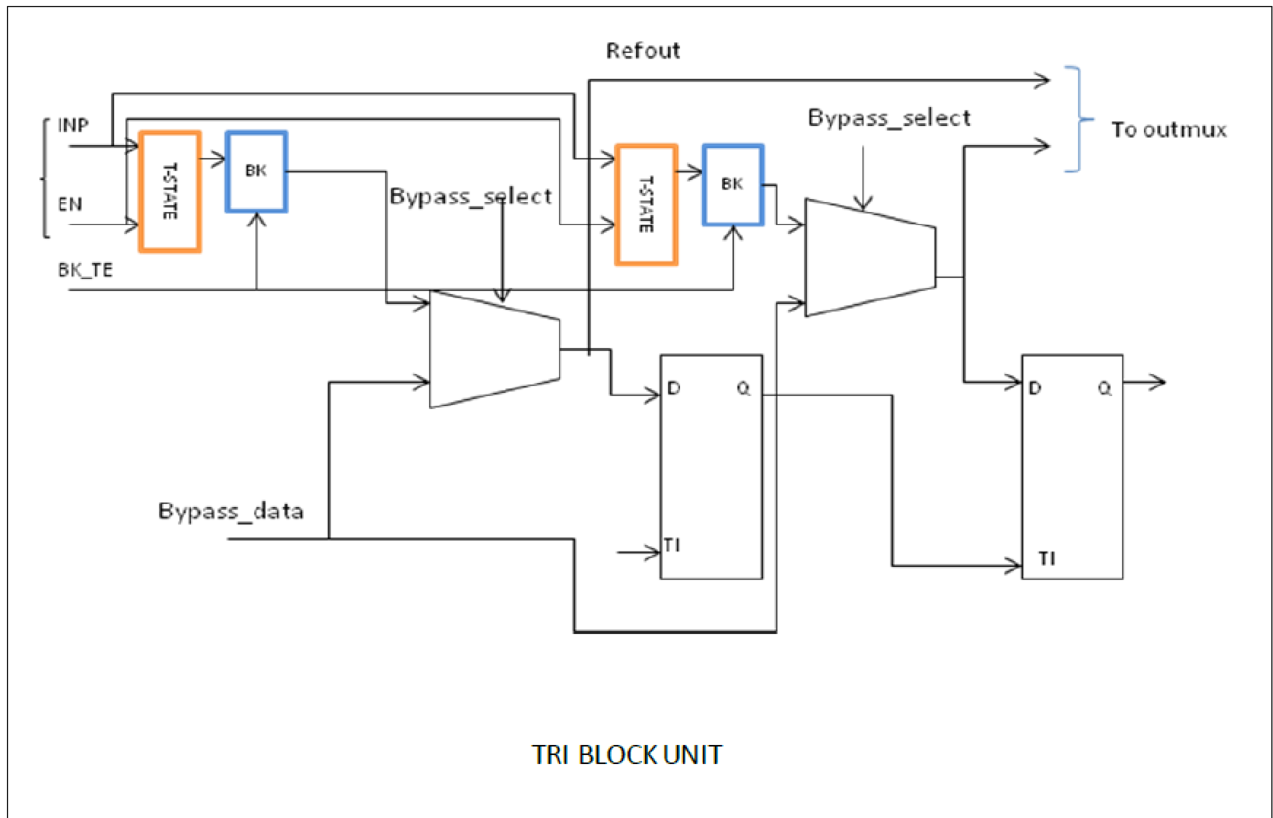

Figure 3.9: TRI

CBUF

The Clock gating cells are kept in CBUF unit of ALLCELL block. A basic Clock gating cell has 3 inputs, E (Enable), TE (test_enable), and CP (clock input).

E & TE do the same task of enabling the clock propagation through the cell. Both these inputs are required in the cell because clock gating enable has to be top-controlled in test mode besides being internally generated in functional mode. Both these inputs come from CUT_DATA2. CP input comes from CP [0] of interface unit.

In terms of connectivity of other components, CBUF is same as COMBO or SEQ.

3.3.3 FDD

Basic FDD structure contains 3 units:

- FDD_INTERFACE
- FDD_MUX
- FDS

FDD_INTERFACE

FDD interface, as the name suggests provides an interface b/w top and FDS block containing the cells.

Main functions of this block:

- It provides separate TI inputs to all the FDS cells.
- Provides Blockselect logic to select a particular cell of FDS.
- Provides Blockselect gated Clocks to FDS cells in normal mode, in test mode CP is mapped to all the cells.
- Provides the muxing bw Q & TQ outputs of the FDS cells.

FDD_MUX

FDD MUX is the muxing structure present inside FDD unit. It takes MUXCTRL INOUT_BLOCK MUXCTRL [13:8], as select inputs (same as the block sel inputs in FDD interface). Depending upon these bits, QOUT [35:0] from FDD interface is muxed. The single output MUXOUT then goes to the Ref cell of that particular FDD unit.

Thus the FDD unit containing the above three sub blocks is replicated no. of times (specified on the spec. sheet). This way, we have whole library instantiated no. of required times.

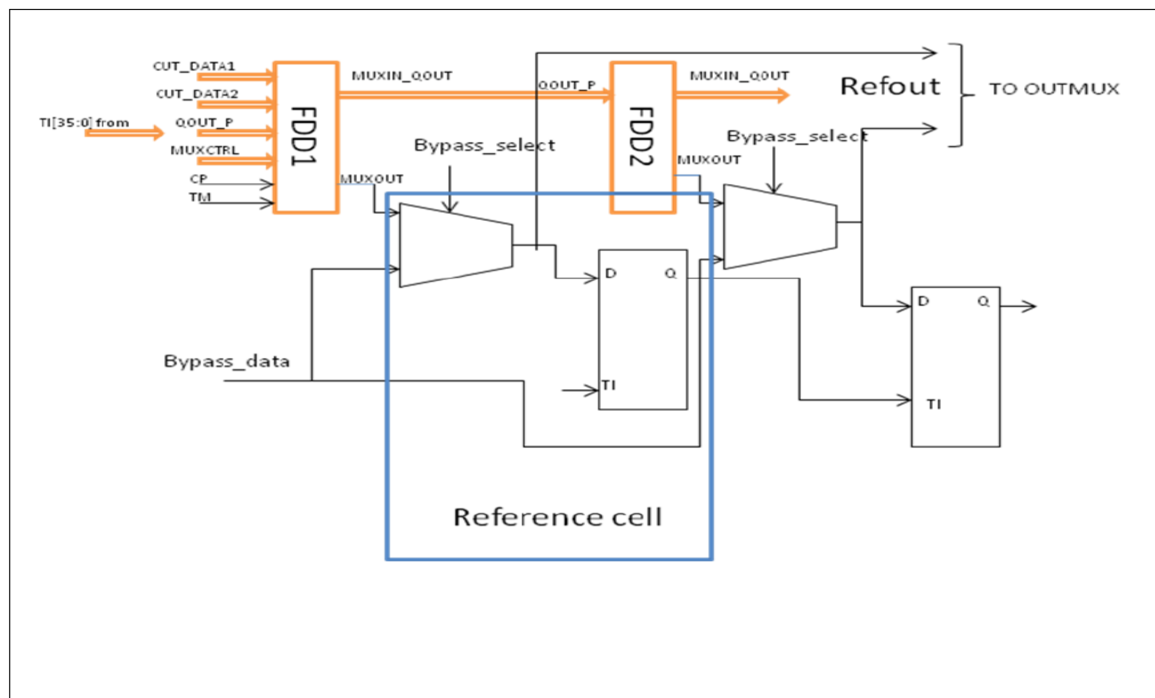

Figure 3.10: FDD MUX

FDS

FDS structure contains the whole library of Cells. It contains another buffer hierarchy of dummy block, one for each cell, interfacing bw the FDS hierarchy and cells inputs and

outputs. Separate TI inputs are provided from FDD Top hierarchy. Both Q & TQ outputs are taken out of each cell, and taken to the top of FDS hierarchy.

3.3.4 RETENTION BLOCK

Retention block is used to test Retention flops which retain their data. Architecture and connectivity wise this block is just same as FDD block. Separate block other than FDD is used for these flops because the SLEEP pins of these flops have to be routed through always on cells.

Chapter 4

Auto Test Pattern Generation Tool

4.1 Test Patterns of Standard Cell

Test Patterns require to access to testchip at time of testing. These are predefined cycle based patterns which are logic low or logic high input value depending on particular pattern. One test pattern is related to one LIB targeted with selecting the respective Standard cell library and their related Mode Selection. Test Patterns give strobing to the output so tester can expect ideal logic value comparing with actual logic value coming.

Test Patterns are written with cycle base language developed by STMicroelectronics. These patterns are delivering to the Testing team for silicon testing of Test chip. Test patterns of any function faults have been written with algorithm basics so that Tester at testing time need not larger size of memory whereas for ATE (Automatic Test Equipment) does require.

Test Pattern is input combination which can drive chip and can tested the functionality of the chip. Test patterns are given to the Fabrication Lab so that their testing engineer would be fired these test patterns and ensures that silicon chips have not any physical defects or faults. The good and bad chips are filtered out. Faulted TestChip are used for the faults and physical defect analysis. There are different test patterns algorithms are used for these testing like example Scan0, Scan1, Scan All BYPASS MUX, BYPASS SCAN, FUNC MUX and FUNC Scan.

Advantage of Test Pattern

- Test Vectors are not needed due to Usage of Test Pattern.
- Tester has lesser size of memory so Test Patterns overcome tester limitation.
- Reusable of cycles which part of Test Patterns.
- Ensures good testing quality.

- Testchip Test Patterns are in written in STMicroelectronics with help of the utile Language which is cycle based language. It requires the entire input chip pin has logic value. It ensures that any of TOP pin does not keep floating because tester does not 'X' as well as can ambiguous the output of the chip.
- Loops are used with given pin logic value with every time clock is given and toggles in cycles which easily debugged by the Test Engineer.
- Cycle is combination of driven input value. When main algorithm file use EXE "Cycle name", Cycle of "Cycle name" is executed and drives values to input pins. Different Cycle file have to been used for different modes.
- This utile language is converted into Verilog which used for the simulation and debug these test patterns at Simulation level.
- After testing the Chip, testing output would be forwarded to the design team and verify the CAD level chip and testing chip output.
- Testchip give better fabrication chip defects and fault models for latest technology development and gives low financial loss if technology is not reliable and stability.

Test Patterns supported files functionality

- First step to define the constant file which has constant that are used in algorithm at any stage. It also has clock cycle time and other time constant defined.
- Second step to define mapping pin information, in test chip standard cell pins mostly same but their pins are varied with top level mapping, so every time to make a pattern is very tedious job. To define compiler wise mapping and its algorithm is most practical and convenient way.
- Third step to define variables those are used later in the algorithms and give the defaults value to their.
- After variable definition writing the procedure files which are used to define value to TOP Pins.
- Next to define cycles depend on Mode where each every cycle has different pin values according to the functionality.
- Main algorithms is defined the pins of top and gives it value according to algorithm iteration or as per the functionality demands. Cycle passes these values to pins and clock is driven after these.
- This repetitive manner of algorithm executes until the proper functionality is not verified.

4.1.1 Flow of Test Patterns

In Test Pattern, above figure showed the way of pattern written, here detailed about how these patterns are worked with design.

- First the constant, variable, mapping, procedure files have to defined.
- Starting with algorithm, first Reset High and Reset Low cycle executes which give High and low logic at reset respectively.
- After this Block selection is mandatory. Block selection is done with data pins and some of the address pins.
- Next LIB selection is mandatory. LIB Selects based on the block indexing so same value (indexing value) have to pass for selection of LIB though from TOP side LIB Number is not as same as Index value.
- After selection of LIB, Providing the mode selection value with same methodology of data pins.
- Test chip is now accessible to standard cell or any other IP to perform the operation on it.

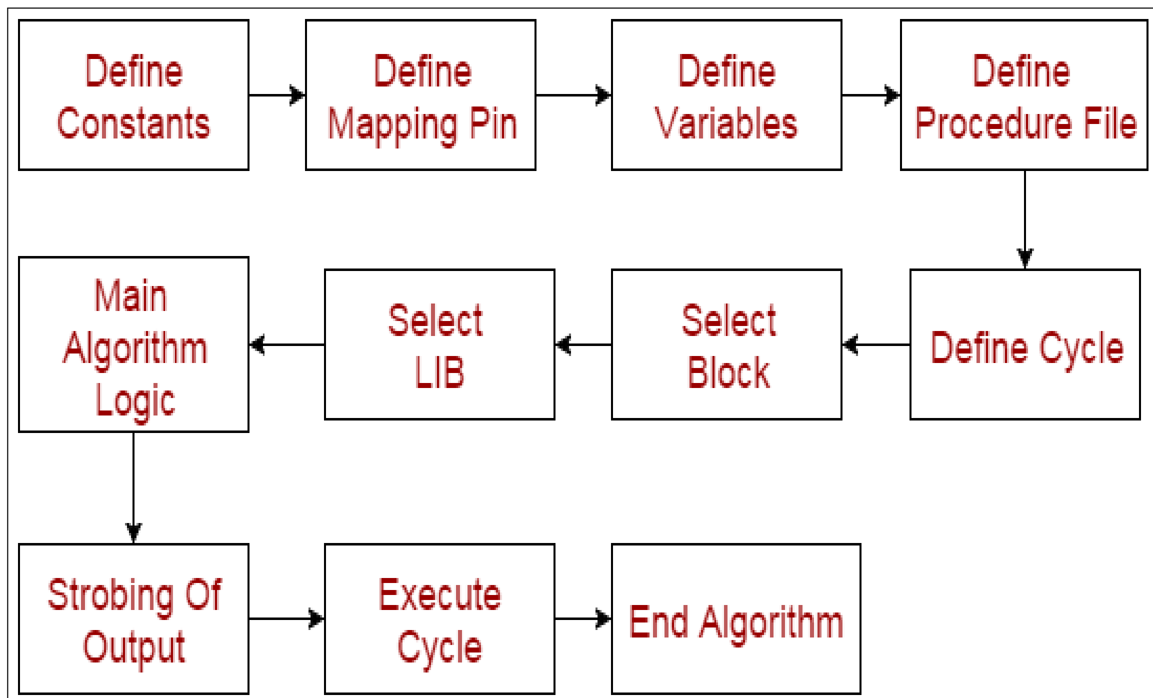

Figure 4.1: Tool flow

4.2 Need of Automation Tool

- Lesser time to generation of automation Pattern.
- Quality of Test Patterns are very much improved due to every single patterns are verified at silicon level.
- Similar Automated Test patterns(conversion to Verilog testbench) utilized in quick validation of RTL and netlsit of Design because this tool only need design document which deliver when RTL is ready.

4.2.1 Problem Statement and Its Solution

SOC integration starts after the test chip or parallel with testchip design. There is any delay in test chip development as SOC will have to be delayed. There is challenge to reduce time for test chip. It reduce time to market SOC or ASIC if it may reduce the time for test chip.

Any SOC or Test chip reduce time to market by how fast verification is performed so reduce time to verified the Design of Test Chip is very important.

Design of Test chip is almost reached at least time. So Challenge is be to verify Test Chip so that we can ensure that SOC might have been right functionality except any fabrication defects.

By Introducing AUTO TEST PATTERN GENERATION TOOL, It generates test patterns automatically by giving the Design mapping Document of IP and Design activation IP Documents. Design mapping Document has the mapping information that which top pins related to the IP pins at the low level hierarchy and their execution cycle file values of top pins. Design activation IP Document has its block select value which used for active particular block and its down hierarchy block called as LIB where actual IP and their supported driven blocks(standard Cell).

Tool makes the Patterns to not relate to Specific Test chip because the top of TEST CHIP always being changed, not Library. Design Mapping Document of Specific Test chip is provided by the Test Chip Developers which are used to generate the mapping files in the patterns and other files and algorithms are always being generic to Standard cell. Tool has the list of the patterns of algorithms and list of LIB information those are being likely to generate.

4.3 Tool

Tool is generated patterns automatically providing input of Design mapping Document and Design LIB Information document. Design mapping documents (CSV file) contains

information of top pin to IP pins and Design IP Information document has list of pattern to generate for any particular LIB and how many combo, SEQ, CBUF cell present in the LIB.

Tool is generated mapping file which is related to Cell of LIB. Pattern Algorithms Pins which are specific LIB pins. Mapping file used #define mapping pin TOP PIN (Processor directive) for test patterns.

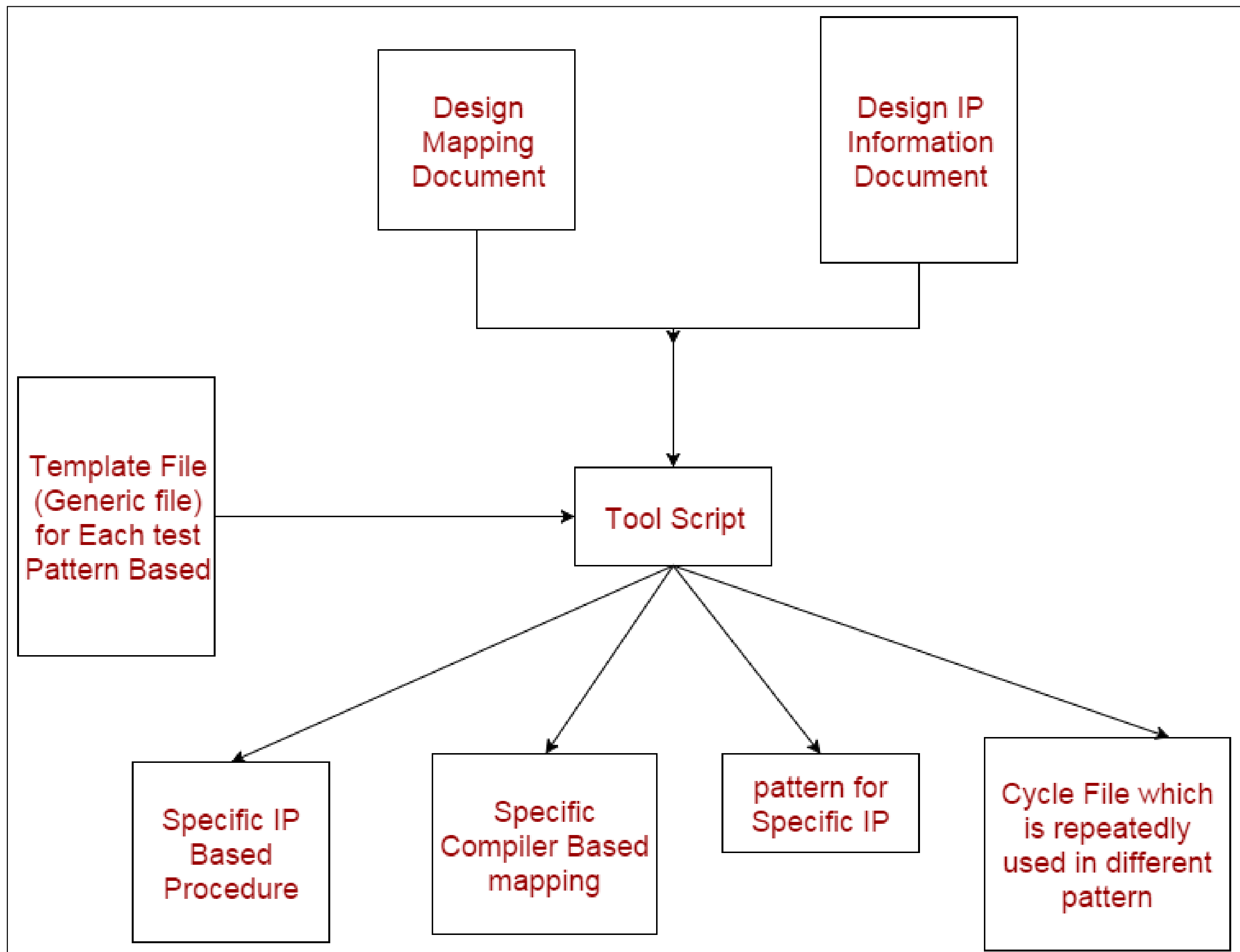

Figure 4.2: Tool structure

4.3.1 Design sheet information:-

- **TOP_PIN:** give information about the top pin(example : MIM_BOT MIM_RES SCANEN TDI TESTMODE TMS BS_CK BS_EN BURNIN COMP_TQ)
- **DIRECTION:** give direction for that pin (example : BIDIR BIDIR INPUT INPUT INPUT INPUT INPUT INPUT INPUT INPUT)
- **INPUT_PIN_MAP:** give input pin definition for which you want use in utile pattern. (example : N/A N/A N/A N/A TEST_MODE N/A N/A N/A OLT_MODE_EN N/A)

- **OUTPUT_PIN_MAP:** give output pin definition for which you want use in utile pattern. (example : N/A N/A N/A N/A N/A N/A N/A N/A N/A N/A)
- **LOAD_REGBANK:** give value o, 1, D, N/A or LOAD_REGBANK for which input pin you want to give as input. (example : 0 0 0 0 0 0 0 0 0 0)
- **STDCELL_EPOD_DUMMY_CYCLE:** give value o, 1, D, N/A for which input pin you want to give as input. (example : N/A N/A N/A N/A 0 N/A 0 0 N/A N/A)
- **CIRCUIT_RESET_LOW:** give value o, 1, D, N/A for which input pin you want to give as input. (example : 0 0 0 0 0 0 0 0 0 0)
- **STDCELL_SCAN_CYCLE:** give value o, 1, D, N/A or STDCELL_CLK for which input pin you want to give as input. (example : N/A N/A N/A N/A 0 N/A 0 0 N/A N/A)
- **STDCELL_FUNCTIONAL_CYCLE:** give value o, 1, D, N/A or STDCELL_CLK for which input pin you want to give as input. (example : N/A N/A N/A N/A 0 N/A 0 0 N/A N/A)
- **SEQ_CELL_FUNCTION_CYCLE:** give value o, 1, D, N/A or STDCELL_CLK for which input pin you want to give as input. (example : N/A N/A N/A N/A 0 N/A 0 0 N/A N/A)
- **CBUF_CELL_FUNCTIONAL_CYCLE:** give value o, 1, D, N/A or STDCELL_CLK for which input pin you want to give as input. (example : N/A N/A N/A N/A 0 N/A 0 0 N/A N/A)
- **TRI_CELL_FUNCTIONAL_CYCLE:** give value o, 1, D, N/A or LOAD_REGBANK for which input pin you want to give as input. (example : N/A N/A N/A N/A 0 N/A 0 0 N/A N/A)
- **SET_DEFAULTS:** give value o, 1, D, N/A or LOAD_REGBANK for which input pin you want to give as input. (example : 0 0 1 0 N/A 0 N/A N/A 0 0)

4.3.2 LIB information sheet:-

- **BLOCK_NAME :** give library name which you want to generate (example : REP1_8T_LIB1)
- **BLOCKSELECT :** give block select for lib (example : 0000_0000_0000_0000_0010_0000_0000_0000)
- **LIB_NUMBER:** give lib number (example: lib1 lib2) it is used for the change number of lib array in variable file.

- **LIB_COUNT:** give lib count which is change in you template file (example: LIB_COUNT 1) it is used for change the value of particular lib different scan pattern.
- **CELL_TYPE :** give name which type of cell you want to produce (example : ALL-CELL , FDD)
- **BASIC_CELL :** give information about which type of basic cell you want to generate (example : combinational, seq, cbuf ,tri)
- **CELL_SELECT_NO:** give how many pin are used for the cell select this used for produce the cell_select_procedure.
- **NUMBER_OF_OUTPUT_COMBO:** total number of output for combinational cell.
- **NUMBER_OF_OUTPUT_CBUF:** total number of output for cbuf cell.
- **NUMBER_OF_OUTPUT_SEQ:** total number of output for the sequential.
- **SCAN_NAME :** give information about how many scan you want to generate
- **CYCLE_NAME :** give information about cycle you want to generate
- **LIBRARY_COUNT:** LIBRARY_COUNT information which is change in template.

4.3.3 Feature of Tool

- Less Time consuming to generate test patterns. Design Mapping and IP information CSV are going to make in one hour.
- Tool takes time to generate test patterns are only 10 min.
- Tool with Sanity Checks, any undefined format of input is not acceptable, flash ERROR and Quit to process.
- Test Pattern template makes and once verify on silicon wafer then after assurance that these patterns have not any bug so QUALITY of TEST PATTERNS are improved very much.
- STMicroelectronics has tool which convert these test patterns to Verilog file which used for simulation and validation purpose so Validation of RTL and NETLIST is very quick and easy.

Chapter 5

RTL Generation Tool

There is generic architecture for Testing the standard cell. We can use that architecture and I can automate writing RTL for standard cell. I develop tool which is generating the RTL for the standard cell test chip. Basic Architecture is as shown earlier. its use same refcell for create scan chain and only cell are going replace with which we want to test. That is why automation is needed for generating RTL. Manually, RTL write take time of 3-4 days. This can be done by only 2 to 3 hours only. By time within 1 or 2 day test Spec change if we write the RTL manually it will take more 2 or 3 day which is not help to tackle with time to market product. Its require to done using automation

5.1 Need of Automation Tool

- Lesser time to generation of RTL.
- Quality of RTL are very much improved due to every cell is separated using the Design compiler and Prime Time.
- Its Generate RTL cum Netlist of the Test Chip for the BLOCK level so it don't require to synthesis. Its also save time for its.

5.1.1 Problem Statement and Its Solution

SOC integration starts after the test chip or parallel with testchip design. There is any delay in test chip development as SOC will have to be delayed. There is challenge to reduce time for test chip. It reduce time to market SOC or ASIC if it may reduce the time for test chip.

Any SOC or Test chip reduce time to market by how fast design is performed and if there is some fault in design we can take over on it. so reduce time to verify the Design of Test Chip is very important.

Design of Test chip is almost reached at least time. So Challenge is be to verify Test Chip so that we can ensure that SOC might have been right functionality except any fabrication defects.

By Introducing RTL GENERATION TOOL, It generates RTL of the standard cell library block automatically by giving the Design Document of IP and group of the library. Design Document has the library block level information that which library we want to create library and which library we want to combine and also contain the reference library we can use that library cell as some reference library. We also give the library file of the standard cell in two formate .lib and .db with there corner. We give some information about the path from where its take library and from where its generate output.

Tool makes the RTL to not relate to Specific Test chip because the top of TEST CHIP always being changed, not Library. Design Document of Specific Test chip is provided by the Test Chip Developers which are used to generate the RTL and Architecture is always being generic to Standard cell so its easy to create the RTL of the Library. Tool has the list of Library with the reference library.

5.2 Tool

Tool is generating RTL automatically providing input of Design Document and Library file contain the cell information. Design documents (CSV file) contains information of Library with there corner. Also contain the reference library with reference cell.

Write RTL for Standard cell testchip without the automation take too much time. Which leads to increase the time to market of any product. So test chip has very less time to validate IPs. So there is already one tool which use library of the standard cell and process on that file than segregate the cell in different category. Also define the pin of the cell for port map. It use XML language for segregate the cell information from the .lib file of the standard cell lib. As Standard cell team use different version of the Design compiler which result in change some of the information may change for standard cell. It difficult for segregating the cell in different category. Designer does not has knowledge on the xml language which leads to increase the difficulty in debug the tool.

So They require atleast one engineer for maintain the tool. I come with one solution. For that i use design compiler and prime time tool for identify the cell type category and write the script in tcl language for characterize cell. I use design compiler command for identify cell and its pin information. Designer has knowledge on the tcl language if tool misbehave in future than he can easily debug the problem and he can also add new feature if require in tool.

5.2.1 Feature of Tool

- Less Time consuming to generate RTL.

- Tool takes time to generate test patterns are only 10 min.
- Tool with Sanity Checks, any undefined format of input is not acceptable, flash ERROR and Quit to process.
- Easy to debug.

Chapter 6

Work Contribution and Result

- Developed Tool Pattern Generation Tool
- Making Test Pattern for the Test Chip
- Automation for Design Flow to reduce time for tape out used SHELL and PERL Script
- Modification of Verilog standard cell behavioral Code
- Validation of RTL and NETLIST design

6.1 Validation Flow

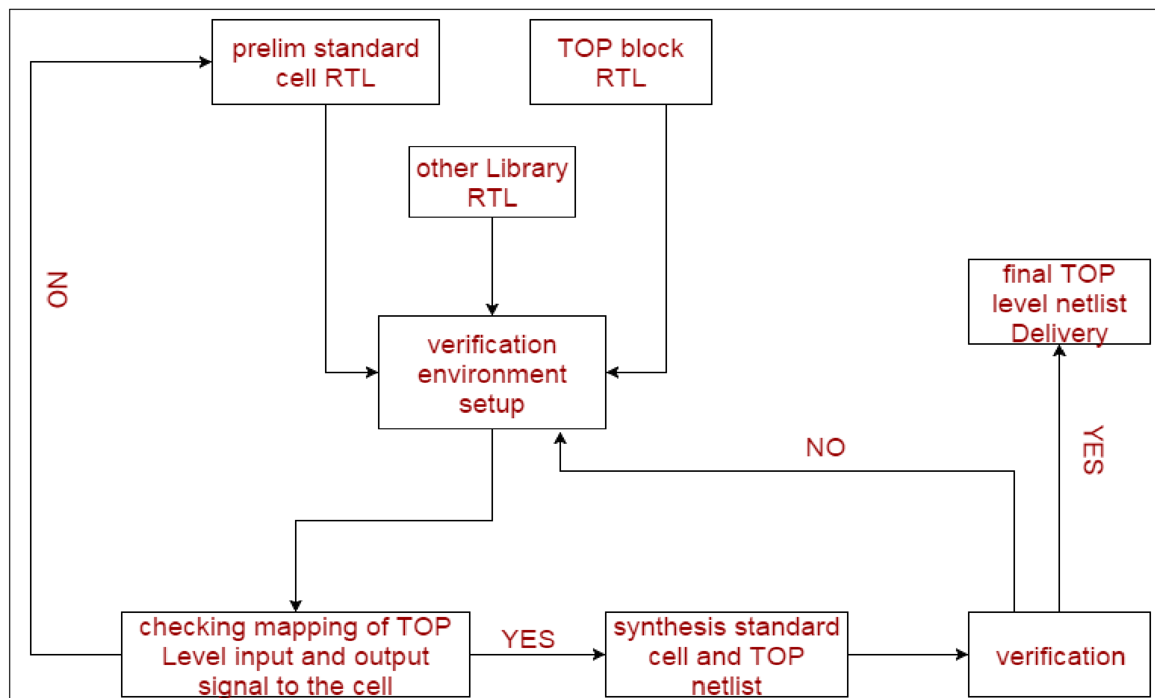

Figure 6.1: Validation Flow

6.2 Simulation Graphs

6.2.1 SCAN 0, SCAN 1 and SCAN ALL

In this test TI-Q path of Reference cell flop is targeted. TE of all the flops is set high, and TI inputs are loaded. During TE=1, the ref flops TI are mapped to their respective Qs and we have a scan chain with input TI and output SCAN OUT.

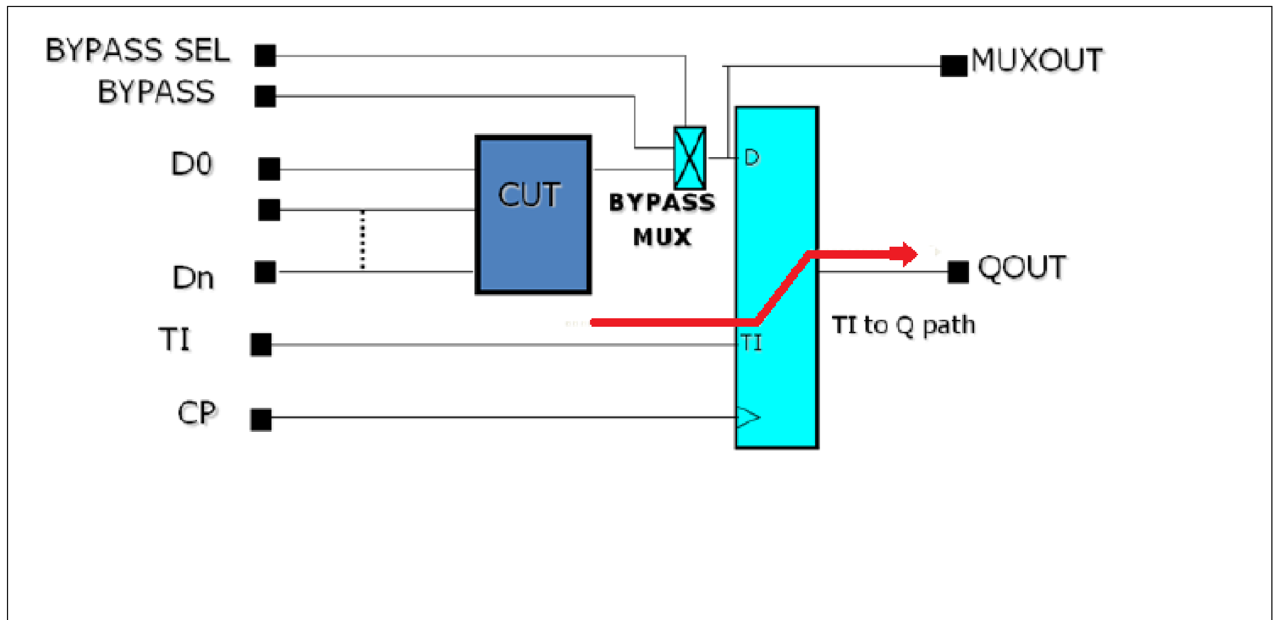

Figure 6.2: ref chain flop for scan0

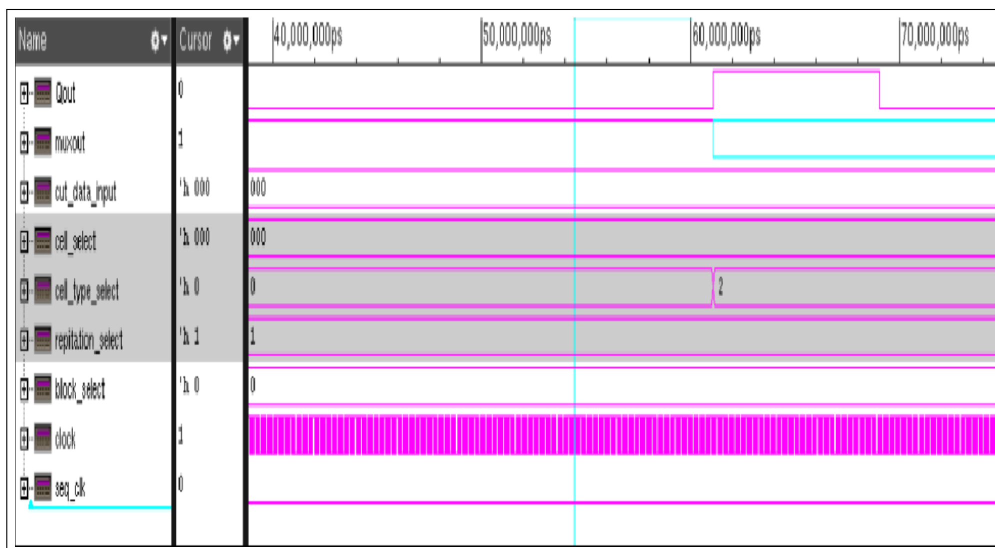

Figure 6.3: scan0

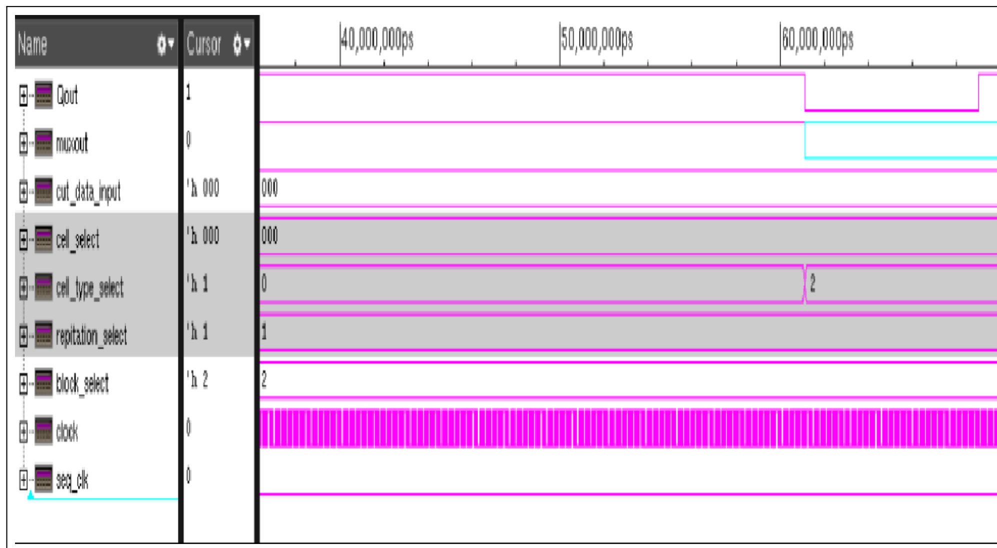

Figure 6.4: scan1

6.2.2 BYPASS MUX

In this test, the Bypass mux is tested. Targeted path is Bypass_data Muxout. For this, Bypass selection is turned on, and appropriate MUX selection is made. The output at MUXOUT should be equal to BYPASS_DATA.

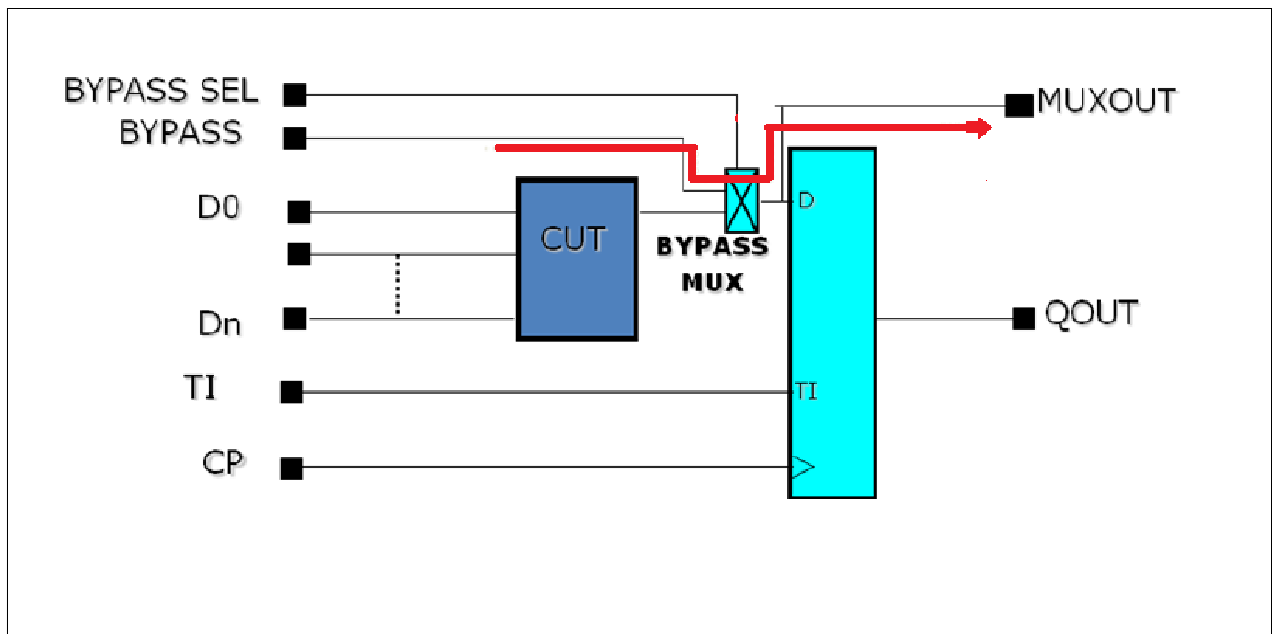

Figure 6.6: ref cahin bypass mux

Figure 6.5: scan all

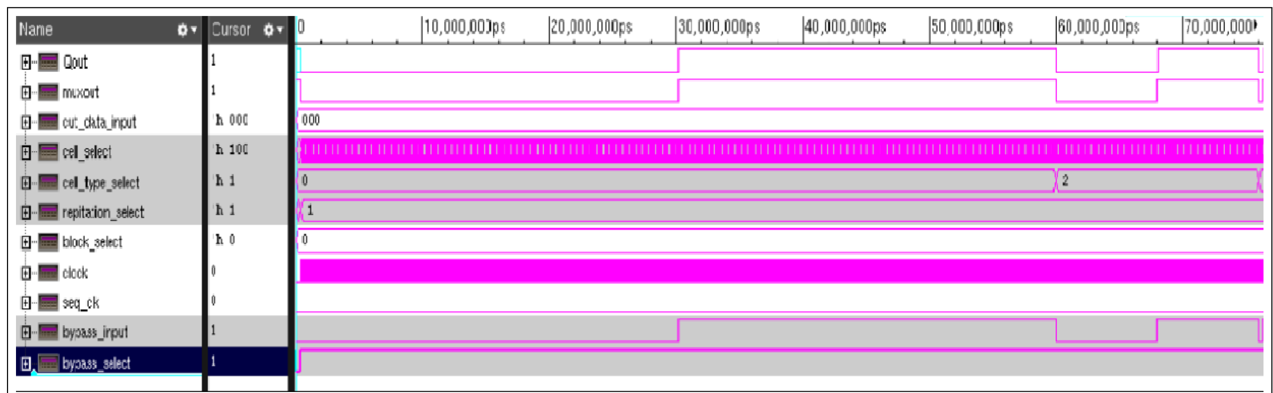

Figure 6.7: bypasss mux

6.2.3 BYPASS SCAN

After we have checked the Bypass Mux, D-TI path of Ref flop is targeted. In this test the Qout data is compared against the BYPASS_DATA.

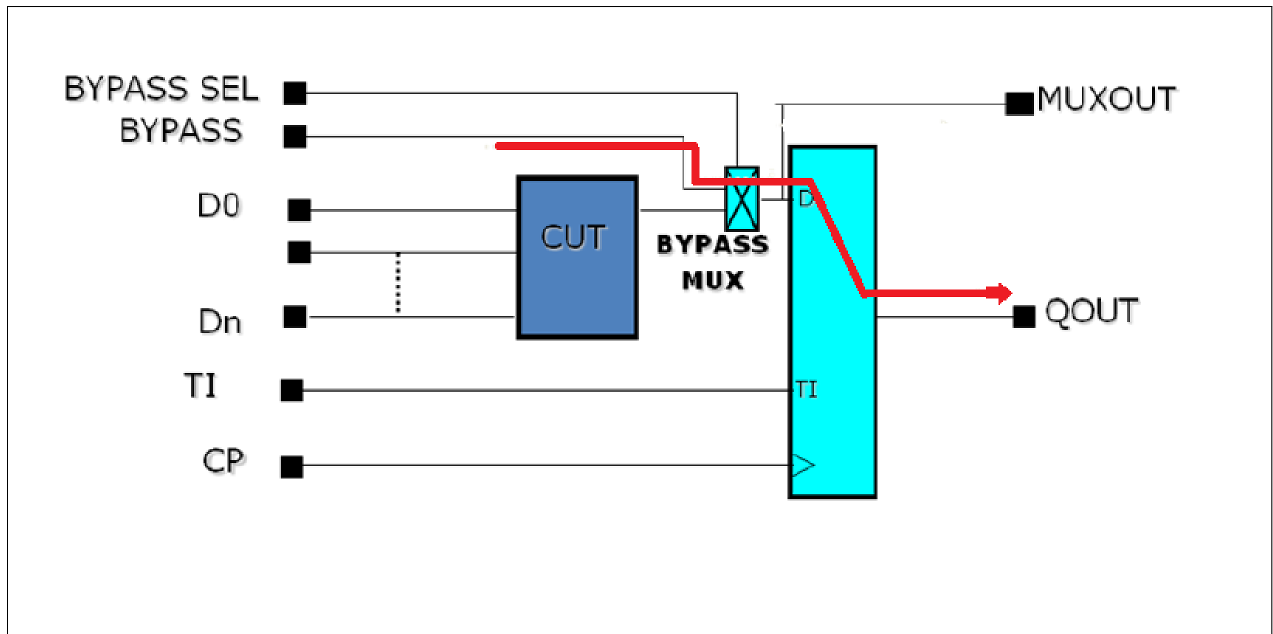

Figure 6.8: bypasss mux

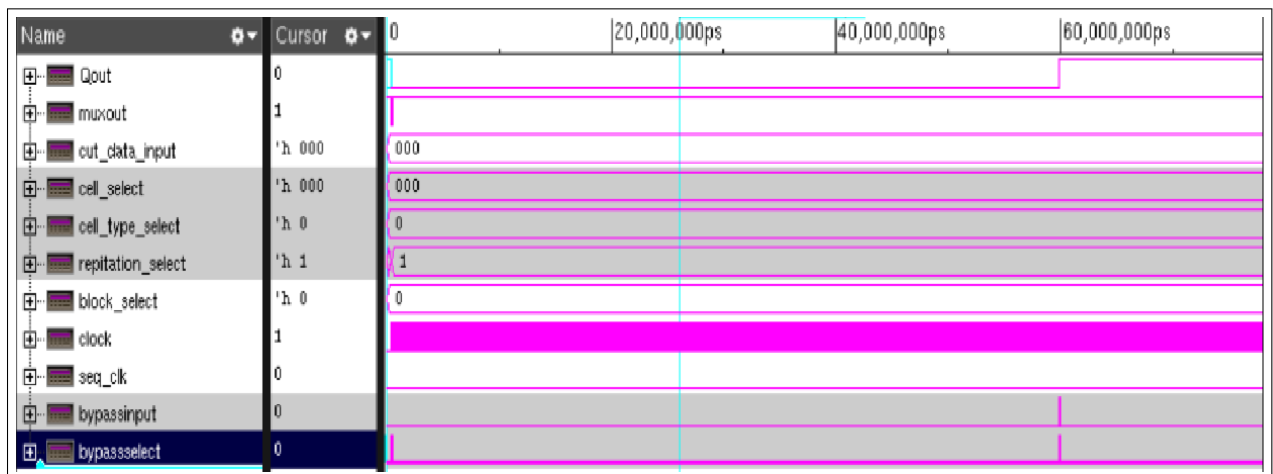

Figure 6.9: bypass scan

6.2.4 FUNC SCAN AND FUNC MUX

Thus after all the paths through which CUT data is received have been checked. we can move to checking CUT functionality. There are two possible paths through which CUT data can be observed which are CUT - MUXOUT and CUT QOUT.

First step towards checking the functionality is to compare the two outputs MUXOUT & QOUT. ATPG pattern cover all the paths in the ALLCELL block and remove the possibility of any uncovered path in the design.

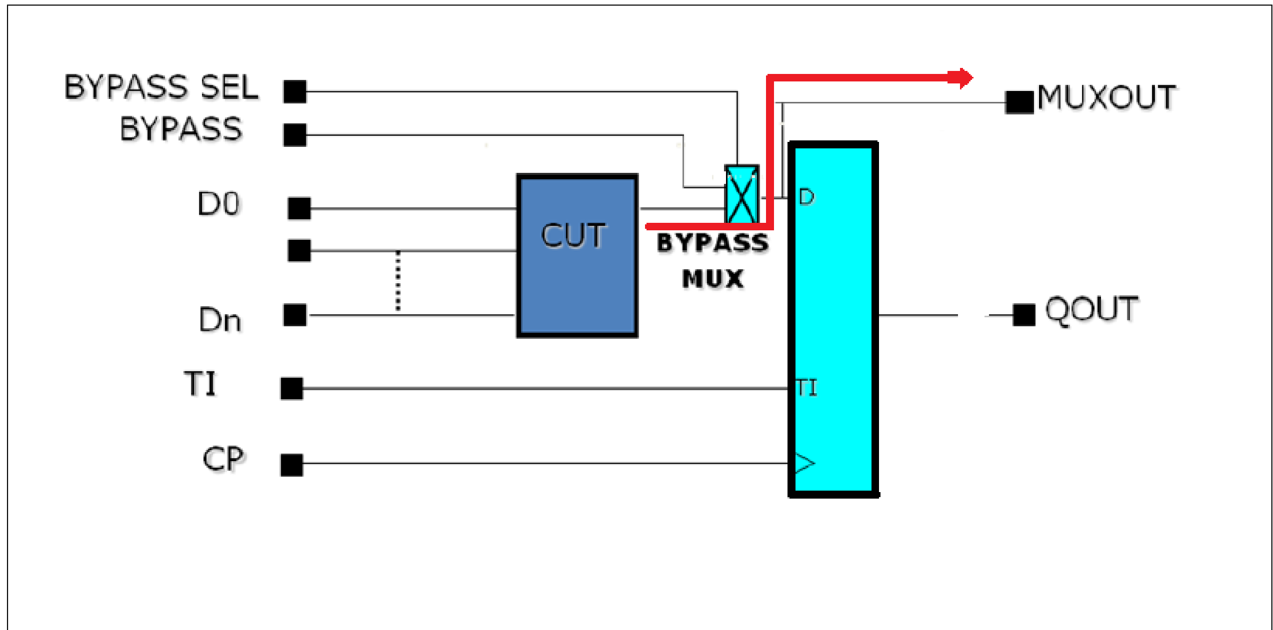

Figure 6.10: ref cahin func mux

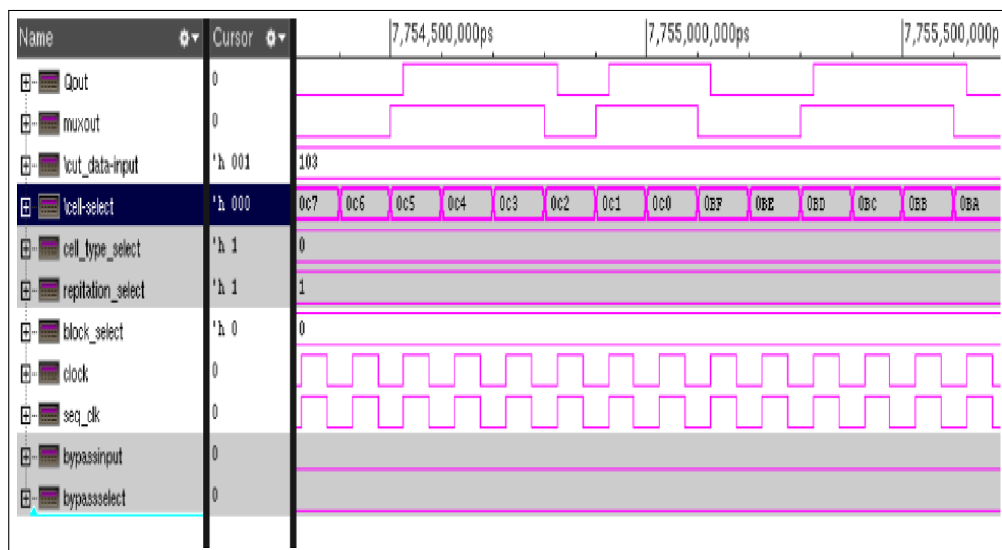

Figure 6.11: func mux

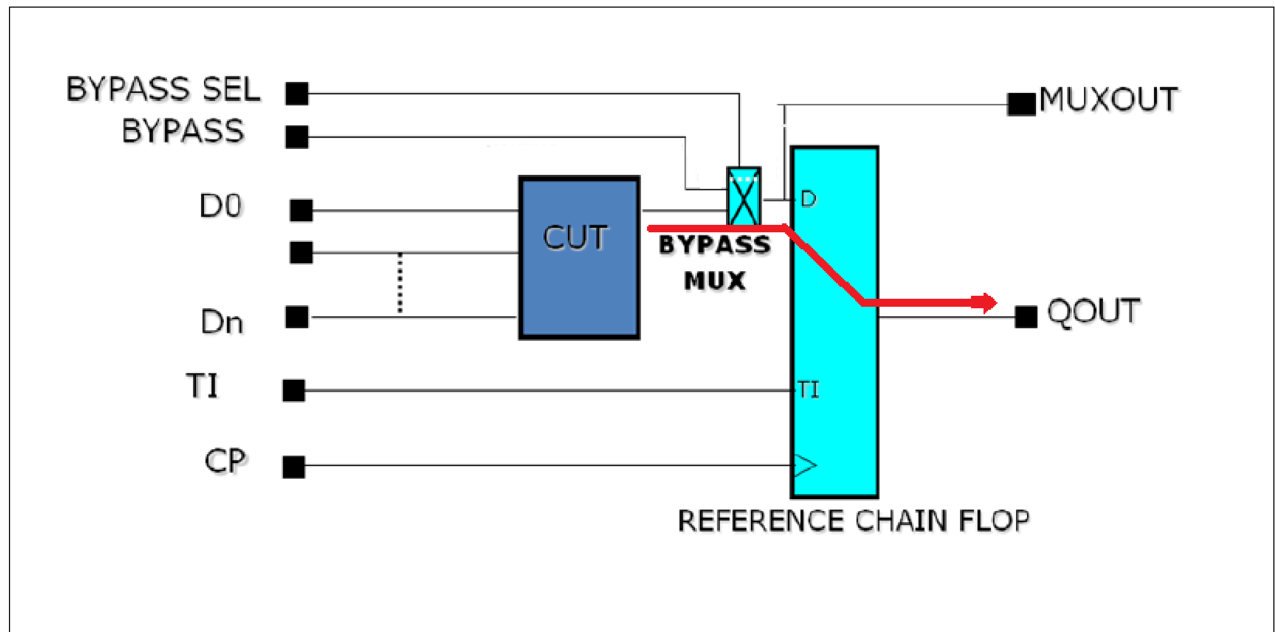

Figure 6.12: ref cahin func scan

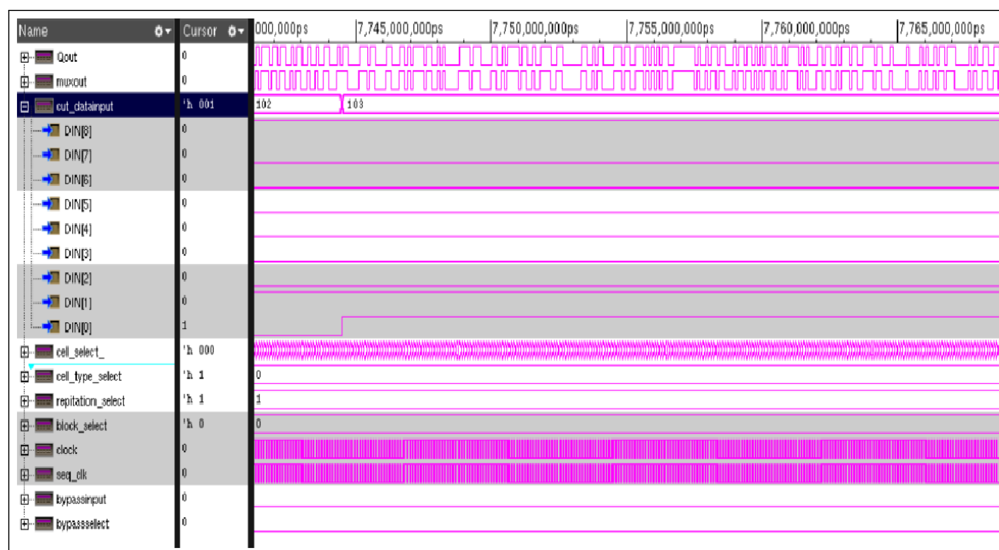

Figure 6.13: func scan

Chapter 7

Tools

- Incisive unified simulator (IUS)
- Shell Scripting

7.1 Incisive unified simulator

Incisive is a suite of tools from Cadence Design Systems related to the design and verification of ASICs, SOCs, and FPGAs. Incisive is commonly referred to by the name NCSIM in reference to the core simulation engine. Depending on the design requirements, NCSIM has many different bundling options of the following tools.

Tool	Command	Description
NC verilog	ncvlog	The NC-Verilog compiler performs syntactic and static semantic checking on the verilog HDL design units. If no errors are found, compilation produce an internal representation for each HDL design unit in the source files
NC VHDL	ncvhdl	The NC-VHDL compiler performs syntatic and static semantic checking on the input source files or VHDL design units. if no errors are found, compilation produces an internal representation for each HDL design unit in the source files
NC Elaborator	ncelab	Unified linker / elaborator for verilog. The elaborator constructs a design hierarchy based on the instantiation and configuration information in the design and compute initial values for all objects in the design.
NCSim	ncsim	Unified simulation engine for verilog. Load snapshot images generated by NC Elaborator. This tool can be run in GUi mod. In GUI mode, ncsim is similar to the debug features of Modelsim.

Table 7.1: NUC Command

7.2 Shell Scripting

A shell script is a computer program designed to be run by the UNIX shell, a command line interpreter. The various dialects of shell scripts are considered to be scripting languages. It uses for automation, text and file manipulation.

Conclusion

- Testchip provides necessity post-silicon validation data before any mass production of SOC and also ensure maturity any technology library.
- Test patterns are cycle based patterns which give advantage of less time for testing and implementation reusability of cycle.
- Test Patterns are similar algorithms except minor modification needed so automation tool is developed which have improved test patterns quality and less human interruption. Standard cell Test Chip use almost same architecture we can automate RTL generation for better quality and less human interruption.
- Tool , reduced human efforts , is developed test patterns and RTL which untimely time saving and used in generation of RTL and after that using Test pattern validate that RTL and Netlist in early days of Design.

Bibliography

[1] ST Internal Documents

[2] A. J. van de Goor “Testing semiconductor memories: theory and practice ”

[3] Jen-Chieh Yeh, Chi-Feng Wu, Kuo-Liang Cheng, Yung-Fa Chou, Chih-Tsun Huang, and Cheng-Wen Wu ”Flash Memory Built-In Self-Test Using March-Like Algorithms” , National Tsing Hua University

[4] Sung-Mo Kang & Yusuf Leblebici “Cmos Digital Integrated Circuits Analysis and Design”

[5] M.S. Abadir and J.K. Reghbati, LSI testing techniques.

[6] A Ring Oscillator Based Variation Test Chip by Joseph Sinohin Panganiban - Massachusetts Institute of Technology