Methodology For Silicon Qualification Of Standard Cells: Design, Verification And Silicon Debug

Major Project Report

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology

in

Electronics & Communication Engineering

(VLSI Design)

By

Ankita Agrawal

(15MECV02)

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

May 2017

Methodology For Silicon Qualification Of Standard Cells: Design, Verification And Silicon Debug

Major Project Report

Submitted in partial fulfillment of the requirements
for the degree of

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

By

Ankita Agrawal

(15MECV02)

Under the guidance of

External Project Guide:
Mr. Balwant Singh
Senior Section Head
ST Microelectronics Pvt Ltd,
Greater Noida

Internal Project Guide:
Dr. N.M.Devashrayee
Institute of Technology
Nirma University
Ahmedabad

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

May 2017

Certificate

This is to certify that the Major Project entitled **"Methodology For Silicon Qualification Of Standard Cells: Design, Verification And Silicon Debug "** submitted by **Ankita Agrawal (15MECV02)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in VLSI Design , NIRMA University, Ahmedabad is the record of work carried out by her under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.The results embodied in this 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.

Dr. N. M. Devashrayee
Internal Guide

Dr. N. M. Devashrayee
PG Coordinator (VLSI Design)

Dr D K Kothari
Head, EC Dept.

Dr. Alka Mahajan
Director, IT-NU

Date:

Place: Ahmedabad

Certificate

This is to certify that the Project entitled "**Methodology For Silicon Qualification Of Standard Cells: Design, Verification And Silicon Debug**" submitted by **Ankita Agrawal (15MECV02)**, towards the submission of the Project for requirements for the degree of Master of Technology in VLSI Design, NIRMA University, Ahmedabad is the record of work carried out by her under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.

Mr. Balwant Singh
Senior Section Head,
ST Microelectronics Pvt Ltd,
Greater Noida

Declaration

This is to certify that

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

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

-Ankita Agrawal

Acknowledgement

I would have never succeeded in my thesis without the cooperation, encouragement and help provided to me by various people. Firstly, my sincere thanks to my team for their help during this ongoing internship. Their wisdom, clarity of thought and support motivated me to bring this project to its present state.

I am deeply indebted to my thesis supervisors, **Mr. Balwant Singh**, Senior Section Head at ST Microelectronics Pvt Ltd his constant motivation regarding the project and also for his constant guidance, supervision, kind co-operation, and invaluable support in all aspects.

I would like to express my sincere gratitude to **Dr. Alka Mahajan** (Director of Institute of Technology, NIRMA University, Ahmedabad) and **Dr. D.K.Kothari** (Head of Electronics and Communication Department) for his continuous guidance and support. I Would like to take this opportunity to thank **Dr. N. M. Devashrayee** (Professor and Program Coordinator, M. Tech - EC (VLSI Design)), and my Internal Guide and all the faculties for their vision, support, and encouragement to provide me with the opportunity to carry out my project work in such a renowned and esteemed organization.

Last but not the least I wish to thank my friends for their delightful company which kept me in good humor throughout the year and thus helping me complete the degree program successfully.

- Ankita Agrawal

15MECV02

Abstract

A new technology before being used in products has to be validated on silicon to check for various parameters like power consumption, manufacturing feasibility, electrical characteristics and PVT variations etc. Once it has been silicon qualified, it can then be used for mass production. Test chips, as the name suggests, are used to test these new technologies and IP cores like standard cell library, memory library, PLL, PMBs, etc. on silicon. Here, at ST, the testing architecture used to test the functionality of each cell of standard cell libraries is called ALLCELL. ALLCELL consists of a cell whose output is taken to a scan flop, called reference flops. Each standard cells output is taken into a scan flop, called reference cell, and they are stitched together. Cell output is also directly taken to a mux. The two outputs are compared to validate the functionality.

In order to implement this methodology, complete RTL to GDS flow is employed. Design RTL is coded in Verilog and then using industry standard synthesis tools, the design is synthesized. The design is then functionally verified on CAD using simulation tools. These functional patterns are tested on silicon through the ATE. Post silicon results are analyzed through the debug setup environment.

Contents

List of Figures

Chapter 1

Introduction

1.1 Testchip Introduction

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.

1.2 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.

Test chip is single chip contains different type of IP blocks which includes for silicon qualification and functionality test.

Advantage

- It provides the best design accessibility 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.

1.3 Purpose Of Testchip

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 Figure 1.1 below:

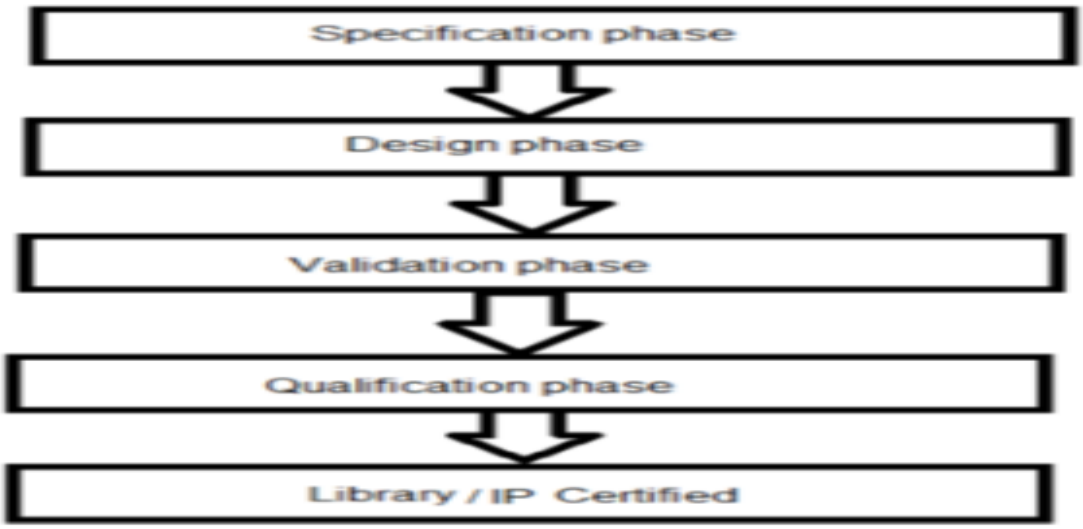

Figure 1.1: Maturity level of Library and IP

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.

1.4 Architecture Of Testchip

TestChip Architecture is shown as in Figure 1.2 above with TOP Level hierarchy. Their functionality are given below:

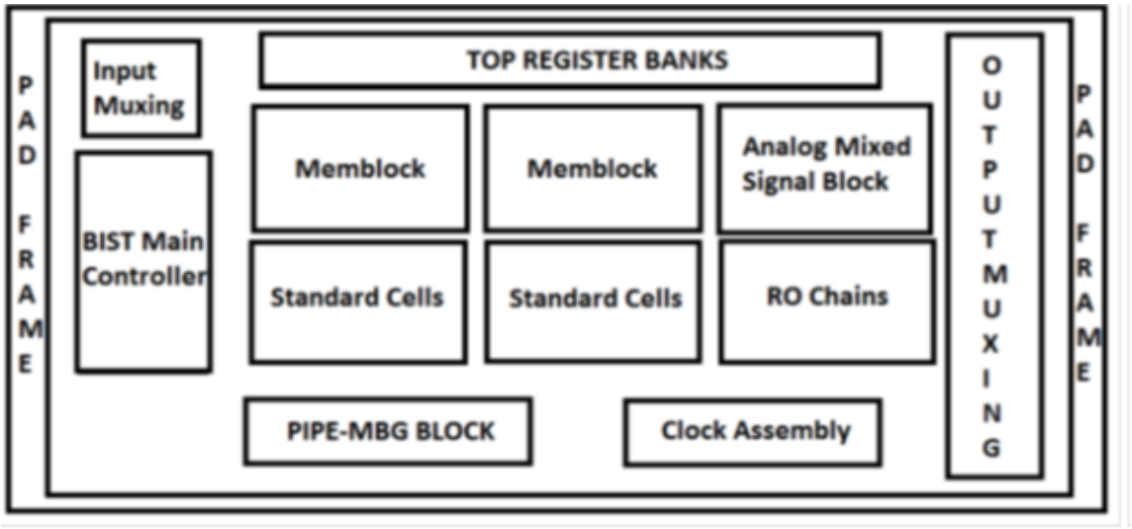

Figure 1.2: Testchip Architecture Block

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.

1.5 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 oating point unit (FPU). Due to the bug, the processor would return wrong outcomes for some computations utilized in math and science. The error was rarely encountered by users. Intel were criticized very heavily and ultimately recalled the defective processors.

1.6 Components added to Testchip

- Memory IP
- Standard Cell
- Analog Mixed IP

1.7 About Project

For Testing functionality and timings and delays of standard cell in silicon, we have a design named as ALLCELL block for the testchip. So, as part of front end flow, we have:

1. To generate the RTL of ALLCELL block with respective DUTs.
2. RTL of the block is to be synthesize.
3. Generate the test patterns to functionally validate each cell of the library.
4. to make a tool for silicon debug in which post silicon results are analyzed through the debug setup environment.

Since the architecture of ALLCELL block is standardized or largely, frozen, we can stitch together the various steps involved in the Front-end flow for the development

of ALLCELL block. This will help greatly in reducing design cycle time. Currently, it takes 4-6 weeks for the block development. So, Intent is to bring it down to 2 weeks- one week for development and validation, another for BA simulation. Also, errors due to manual intervention will be reduced owing to a standard flow procedure. Moreover, deployment of design and flow know-how to others will be simplified.

Chapter 2

ALLCELL Design

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 united information utilized as a part of planning a SOC (system on chip). It is an accumulation of low level rationale capacities, for example, AND, OR, INVERTER, ip-operations, latches and buffers. These cells are acknowledged as fixed height variable width full custom cells. The key angle with these libraries is that they are of a settled stature, which empowers them to be set in lines, facilitating the procedure of computerized advanced format. The cells are ordinarily upgraded full custom designs, 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 blocks present at the top. ALLCELL interface block which have certains buffers and then DUTs which have to test and finally two outputs, one is directly from the DUTs and other one is scanned output from REFRENCE cell. The lenght of Refcell depends on number of DUTs.

2.1.2 FDD Block

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

Figure 2.1: FDD Block Pin

The dual-edge triggered ip-op 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 ops, 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 figure 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.

2.1.3 Retention Block

Retention block is used to test Retention ops 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 ops because the SLEEP pins of these ops have to be routed through always on cells.

2.2 ALLCELL Design

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

There can be several blocks present at the top. ALLCELL interface block which have certains buffers and then DUTs which have to test and finally two outputs, one is directly from the DUTs and other one is scanned output from REFRENCE cell. The lenght of Refcell depends on number of DUTs.

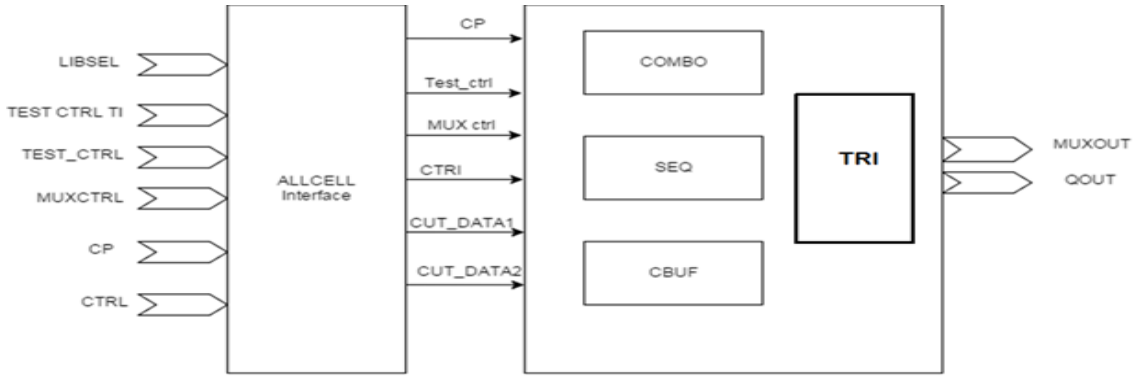

Figure 2.2: ALLCELL Top

2.2.1 ALLCELL Interface

The Input Block is an interface block which is made up of buffers as shown in figure 2.3.

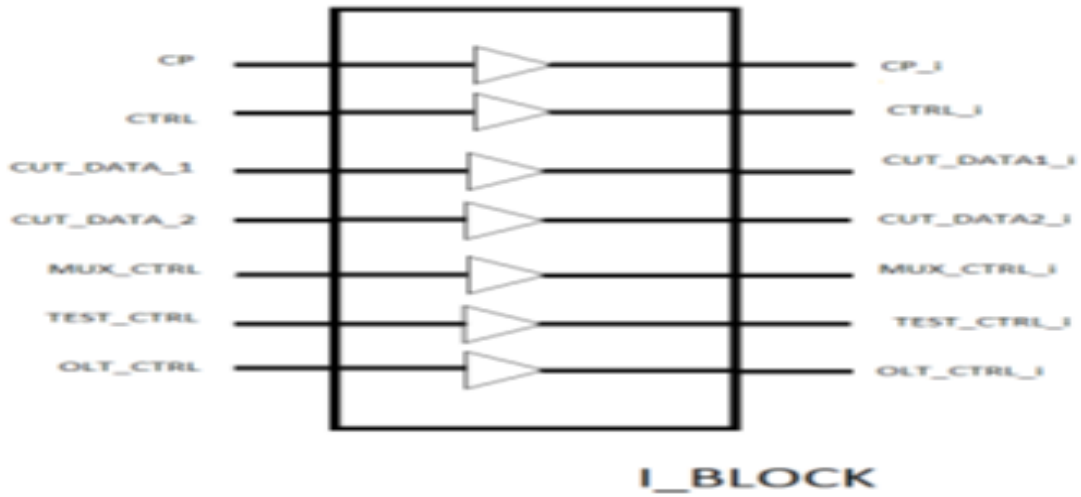

Figure 2.3: Input Block

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.

2.2.2 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 Figure 2.4.

Main components of this unit are:

- Cell Under Test [CUT]
- Reference Cell
- Outmux and Refout

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 op. Ref ops are connected in a scan chain such that, Q output of each op is connected to TI Input of next op. The no. of ref ops for each CUT depends upon type of block i.e. [Allcell, FDD].

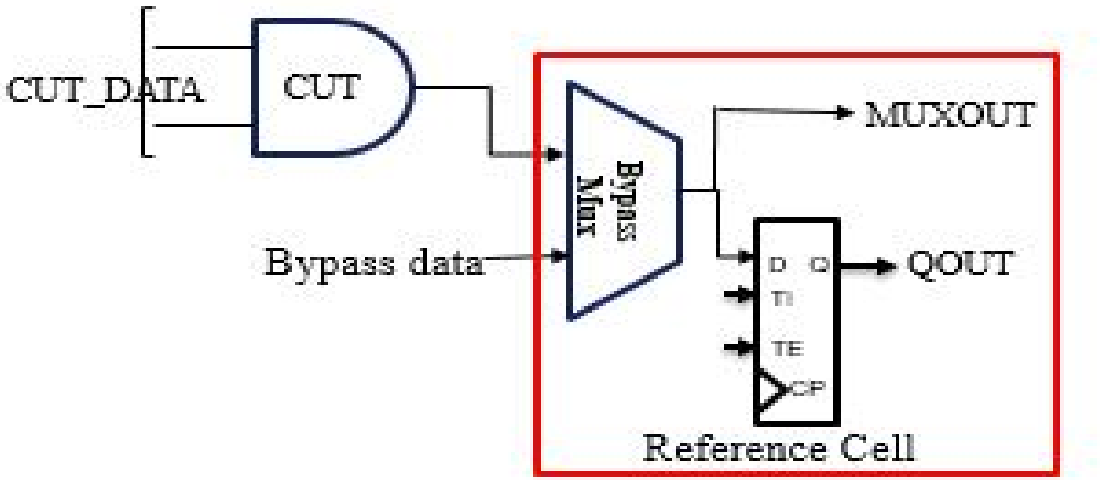

Figure 2.4: COMBO Block

2.2.3 SEQ

In SEQ the cells are ops. CP<0> from Interface unit is given to these ops as clock input. CP<1> from Interface unit is given to Reference ops 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.

2.2.4 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.

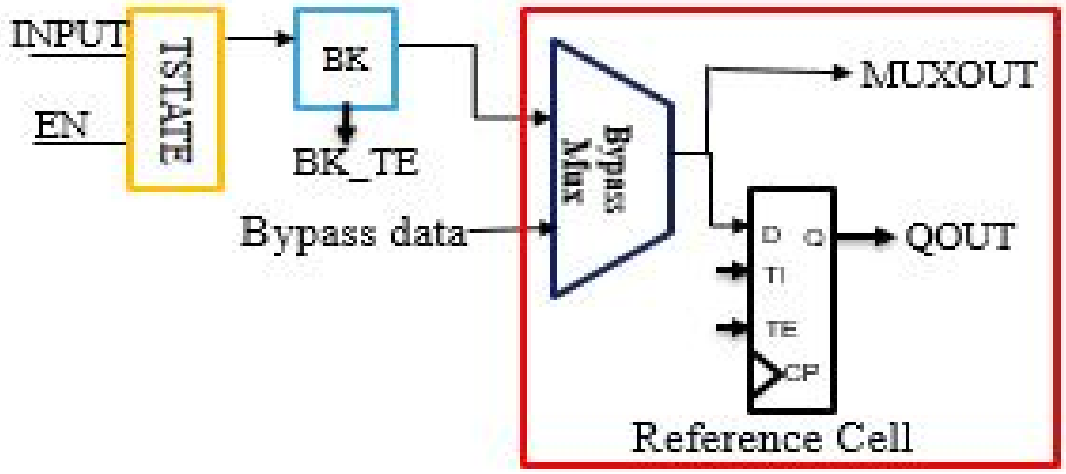

Figure 2.5: TRI Block

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.

2.2.5 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 and 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 CP0 of interface unit. In terms of connectivity of other components, CBUF is same as COMBO or SEQ.

2.3 ALLCELL High Speed Interface

With the realm of advancements in nanometer technologies, today's chip design complexities are increasing manifold with the shrinking feature size. Thus, with such increased complex designs, consisting of millions of gates, operating at GHz frequency range, at-speed test is crucial.

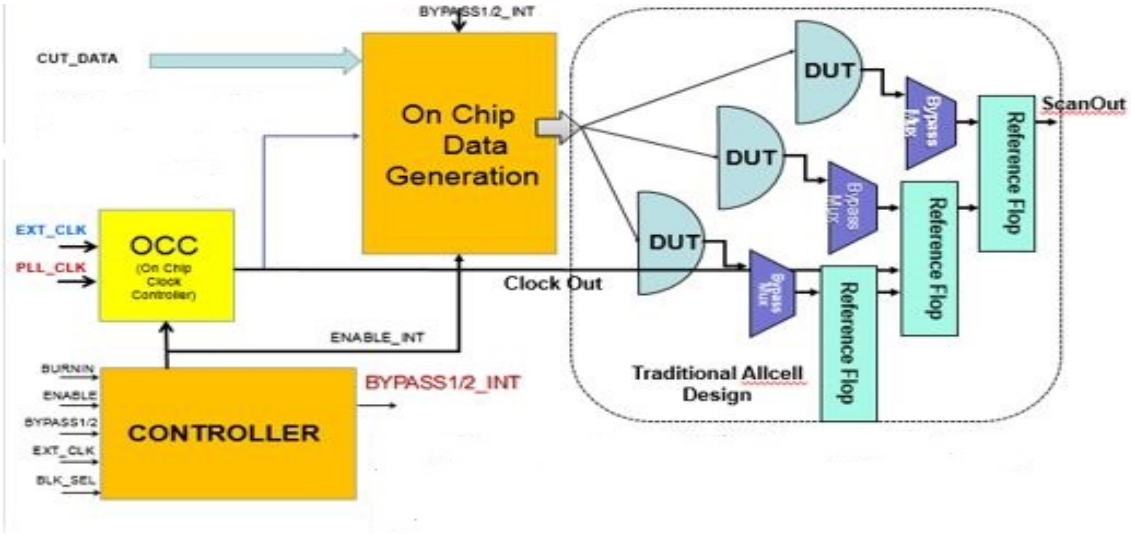

Figure 2.6: High speed functional test in allcell Design

The methodology presented here establishes a functional testing environment for standard cells by utilizing the conventional transition fault method used in At-Speed ATPG at super GHz frequencies of operation. This method is used to stress the standard cell DUTs at ultra high frequencies beyond being functionally tested.

2.3.1 Standard Cell Testing Structure

Any pattern generator intending to functionally test complete libraries of standard cell must generate vectors for all possible transitions. 1 to 0 or 0 to 1. At normal frequencies of operation, a launch flop fires the pattern on the DUT triggering the functionality and matching it with the expected pattern. The resultant pattern set is dumped in a scan chain of capture flops, shifting incrementally dumped patterns serially at normal clock.

2.3.2 At-Speed Fault Models

At present, path delay model and transition model are the two most widely considered At- speed fault models at industrial level, As compared to the stuck at fault model for static testing, testing the logic at-speed necessitates the requirement of test pattern with two parts. The first part launches a logic transition value along a path, and the second part captures the response at a specified time determined by the system clock speed. If the captured response indicates that the logic involved did not transition as expected during the cycle time, the path is said to have failed the test and is considered to be defective.

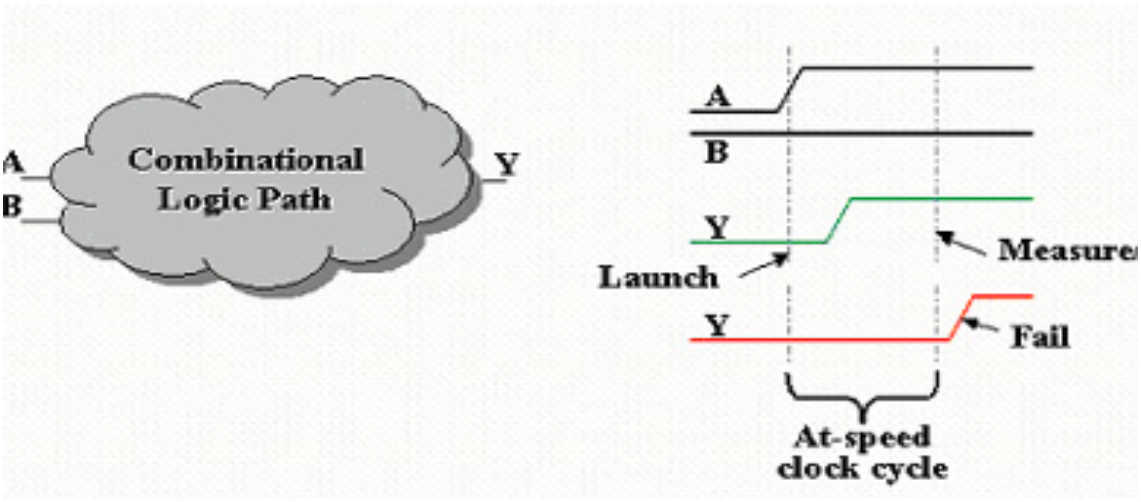

Figure 2.7: Pattern launch event propagates transition

For example, the pattern's launch event may propogate a 0 to 1 (rising edge) transition along a specific path while holding all other conditions constant, as shown in Figure 2.7. Then the capture event pulses a functional clock to latch in the path's response to the transition. If the capture point doesn't detect an expected "high" value in time, the path fails the test and is considered to have a "slow-to-rise" defect.

Transitions are launched from a path's start point and captured at the path's endpoint. The start and the endpoint are typically both scan cells. While paths can start or end at the device input or output pins, testing for these types of paths requires

high resolution clocking typically provided by the ATE. These capabilities are only available on high-end testers. Fortunately, some advanced ATPG tools utilize the internal PLLs to automatically provide these high-resolution clocks.

Transition delay fault model

Transition tests target the transition fault model, which models manufacturing defects that behave as gross delays on gate terminals (pins). It tests for for slow-to-rise or slow-to-fall transition behavior at each pin.

Two types of transition tests can be created: launch-off-shift and broadside. In the launch-off-shift approach, the last shift of the scan chain load also serves as the transition launch event. The time from that last shift (or launch) clock to the capture clock is known as the critical, as shown in Figure 2.8.

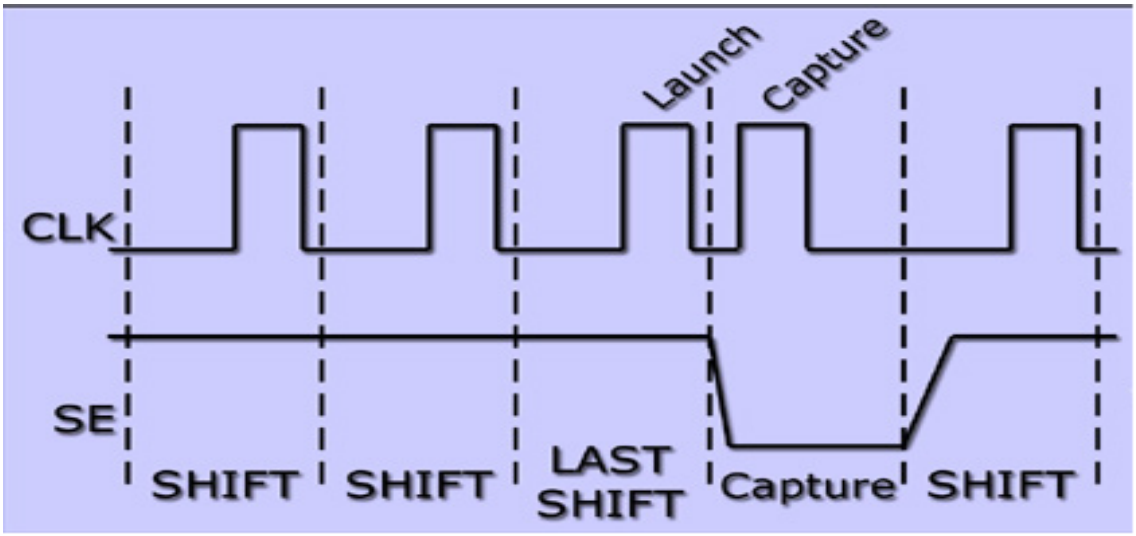

Figure 2.8: Launch-off-shift transition test timing

In the launch-off-shift approach, the scan enable signal (SE) must be able to turn off very quickly after the last shift clock and allow the logic settle before the occurrence of capture clock. For this reason it becomes imperative for the scan enable signal to be routed as a clock signal. Depending on the design frequency required for testing, the scan chains themselves might also be required to shift at system frequencies.

This can be a drawback as most scan chain shifting is done at lower frequencies. If the chains are shifted and tested at-speed, the outcome might be an unnecessary yield

loss. The main advantage of this launch-off-shift approach is that it only requires the ATPG tool to create combinational patterns, which are quicker and easier to generate.

The other transition testing technique is called broadside. In this technique, the entire scan data shifting can be done at slow speeds in test mode, and then two at-speed clocks are pulsed for launch and capture in functional mode. Once the values are captured, the data can be shifted out slowly in test mode. The timing for this technique is illustrated in Figure 2.9.

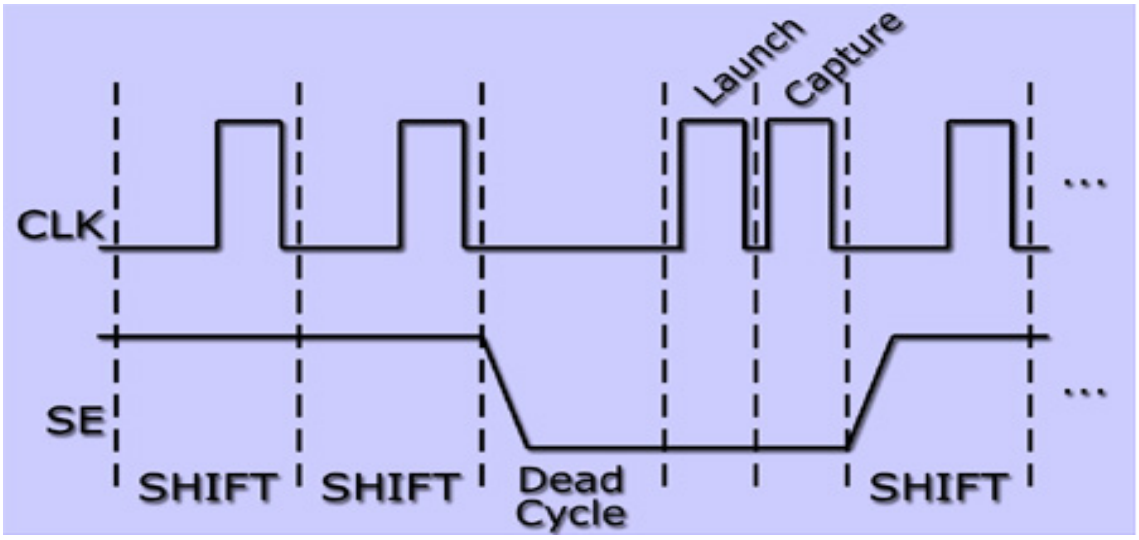

Figure 2.9: Broadside transition test timing

Major advantage of this approach is, it does not require scan chains to shift at-speed or the SE signal to perform as a high-speed clock. So, from the design side, implementation complexity reduces.

Additionally, broadside transition testing can also support LSSD-style scan designs. Because of these advantages, most companies have adopted this approach for creating their transition test patterns. The main disadvantage of broadside approach is that the ATPG problem is now a sequential one, which can increase the test pattern generation time and might result in a higher pattern count. It is also believed that launch-off-shift patterns provide higher test coverage than broadside testing. However, this is not necessarily true. The launch-off-shift approach provides additional detection of some non-functional faults that most companies do not want or need to

test. Including tests for these faults could even contribute to unnecessary yield loss. We are also using this approach.

2.3.3 At-Speed Clock

An at-speed test clock is required to deliver timing for at-speed tests. There are two main sources for the at-speed test clocks. One is the external ATE and the other is on-chip clocks. Traditionally, ATE has always supplied the test clocks. In some cases, the ATE is still a viable source for the at-speed testing clocks. However, the sophistication and cost of the tester increase as the clocking speeds and accuracy requirements rise.

The second source of clocks can come from inside the chip itself. More and more designs include a phase-locked loop (PLL) or other on-chip clock generating circuitry. Using these functional clocks for test purposes can provide several advantages over using the ATE clocks. First, test timing is more accurate when the test clocks exactly match the functional clocks. Secondly, the high-speed on-chip clocks reduce the ATE requirements, enabling use of a less sophisticated and, thus, less expensive tester. Figure 2.10 shows a design with a high-speed on-chip PLL.

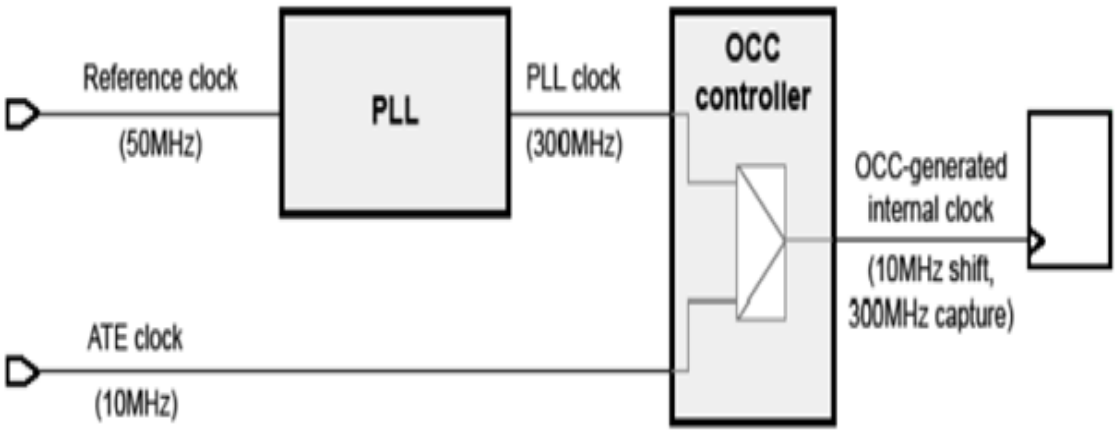

Figure 2.10: High-speed, on-chip PLL

In this situation, the design has both internal and external clocks and control signals. By using something called "named capture procedures," the user can specify the functionality and relationships between the internal and external signals. The ATPG tool then uses these relationships to create accurate at-speed test patterns driven by the on-chip clocks.

Control registers are typically used to program which clocks to pulse and when. These internal control registers can be loaded with specific values by the ATPG tool. The ATPG user simply needs to specify the desired register values by using "condition statements" in the named capture procedures. This method is much easier than loading values through a boundary scan test access port (TAP) controller, and it does not require extra external pins to feed in those register values.

2.3.4 OCC Functionality

When SE goes from 1 to 0, the SE signal is latched on slow clk and synchronized internally to the fast clk. Two pulses are generated for launch and capture flops.

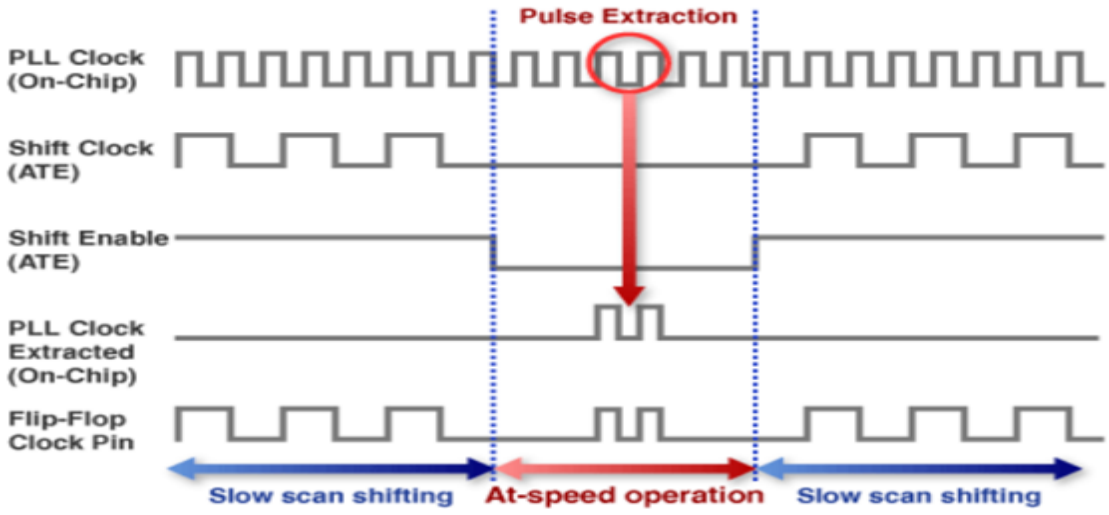

Figure 2.11: OCC Functionality

2.3.5 LFSR

Linear feedback shift registers are pseudorandom pattern generators which when loaded with a seed value generate a pseudorandom pattern of 1s and 0s when clocked. Clock is the only signal necessary to generate the test patterns.

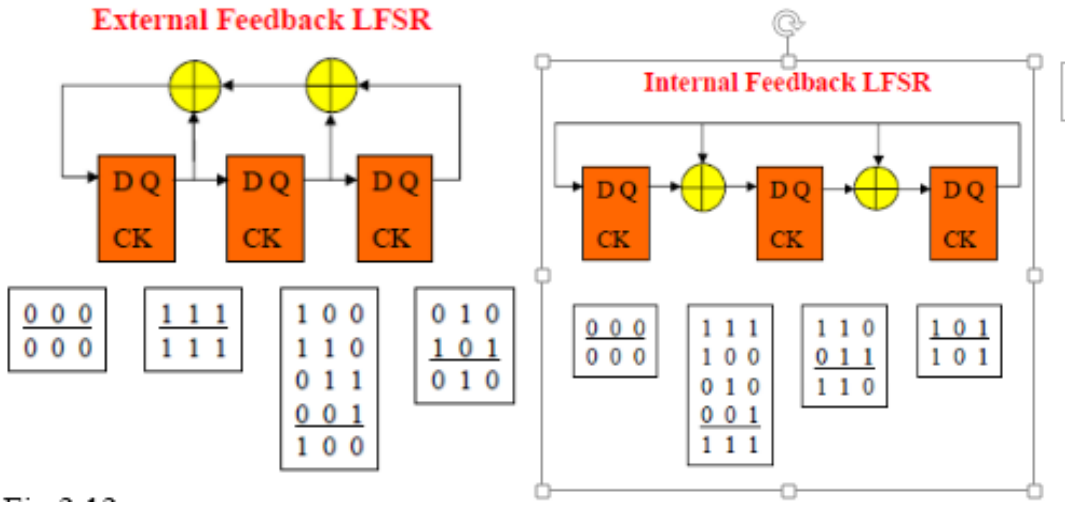

Figure 2.12: Ext. And Int. LFSR

Characteristic Polynomial in Allcell Design P(x) = x6 + x + 1

2.3.6 Data Generator

It includes LFSR and Signal controller block. In signal controller, we give a muxing logic in which if OLT with select 0 and OCC top ctrl is 0 then it takes input internally and if OLT will be high then interally all function would be on. So, the question arises why we test our design on OLT mode.

2.3.7 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 MUX-CTRL 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.

2.4 Summary

For testing the functionality of standard cell , a testing architecture called ALLCELL is employed. Each standard cells output is taken into a scan flop, called reference cell, and they are stitched together. Cell output is also directly taken to a mux. The two outputs are compared to validate the functionality. And also, to stress the standard cell DUTs at ultra high frequencies beyond being functionally tested, we add High speed interface to our design. For that, at speed testing has done by using the transition fault method. And use on chip clock controller for giving slow clock at shift mode and two fast clock, one for launch the data and one for capture the data. And also, for testing the operation life of Standard cells, we check our design in OLT mode also. So, In order to implement this methodology, we write RTL, design RTL is synthesized and tester pattern are generated.

Chapter 3

RTL generation and Synthesis

3.1 RTL Generation Tool

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 testchip. 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. 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. 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 debug by designer. So develop one tool which is using tcl language for generating Register Transfer Logic(RTL).

3.1.1 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. Its also save time.

3.1.2 Implementation of RTL

I again automate this tool to generate the RTL in 30-40 mins only. Functionality of this tool is as follows:

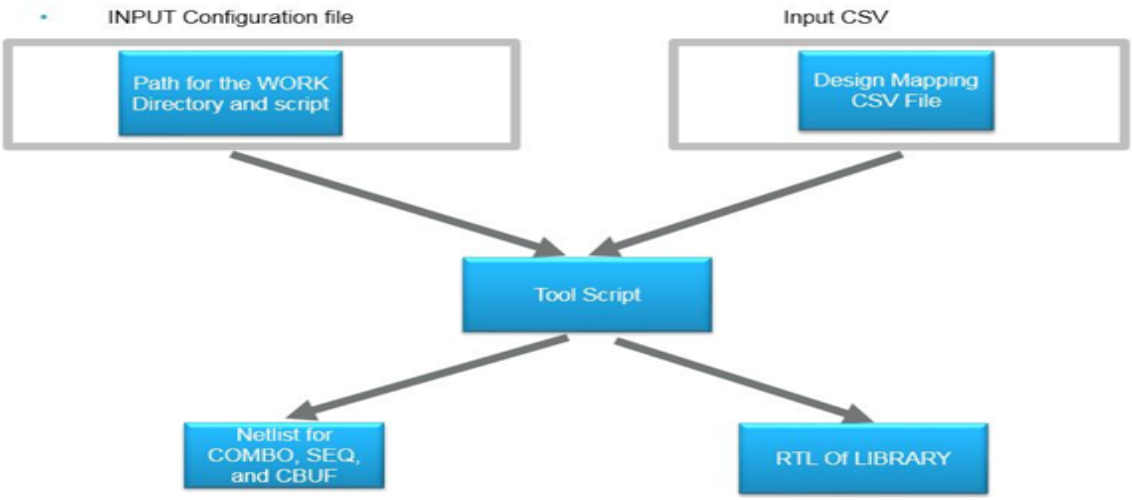

Figure 3.1: Block diagram of RTL Generation Tool

Input CSV:

standard cell specification xls are given by the standard cell team, with the help of that I prepare a CSV file that contains these information.

- **LIBRARY GROUP CELLS-** In this give the library group name. For example, LIB1,LIB2,etc.

- LIBRARY NAME- Give the standard cell library name with its PVT (Process Voltage Temperature) condition, without the .db or .lib extension.
- **FDD ALLCELL RETENTION ANY OTHER-** This column specifies the structure requirement for the paricular library. Fill with YES or NO accordingly in the respective columns.
- **NUMBER OF CELLS IN LIBRARY-** Indicates total number of cells present in library.
- **NUMBER OF COMBINATIONAL CELLS-** Indicates total number of combinational cells.
- **NUMBER OF SEQUENCIAL CELLS-** Indicates total number of sequencial cells.
- **NUMBER OF TRISTATE CELLS-** Indicates total number of tristate cells.
- **NUMBER OF CBUF CELLS-** Indicates total number of CBUF cells.
- **CELLS TO BE TESTED-** Indicates which cells to be tested. If all the cells are to be tested, fill the cell with ALL. If only certain type of cells to be tested, write N/A in the cell.
- **CELLS TO BE NOT TESTED-** Indicates which cells to be not tested. Write the string name from the cell name which is not to be tested. For example, LL,LS,P0,P4.
- **REFERENCE LIBRARY NAME COMBINATIONAL-** Give the combinational reference library name with its PVT condition, without the .db/.lib extension.
- **REFERENCE FLOP NAME-** Give the 'flop' name from reference library.
- **REFERENCE BUF NAME-** Give the 'buffer' name from reference library.

- **REFERENCE MUX NAME-** Give the 'mux' name from reference library.
- **REFERENCE CBUF NAME-** Give the 'mux' name from reference library.
- **REFERENCE OR NAME-** Give the 'or' name from reference library.
- **REFERENCE INV NAME-** Give the 'inverter' name from reference library.
- **REFERENCE AND NAME-** Give the 'and' name from reference library.
- **REFERENCE TRI NAME-** Give the 'tri' name from reference library.
- **REFERENCE BSK NAME-** Give the 'buskeeper' name from reference library.

Copy all the DUT libraries and reference libraries which we are using here while making INPUT.csv with its all ".db", ".lib".

Input config File

Then we make a configuration file in which we give the path of input csv as well as work directory, output directory.

Tool Script

Firstly source the config file that contains the csv and after than one by one scripts.

- **Envfile.csh:** set the environment that in this area all the outcomes would save and give a local name to all the tool script for more clarity.
- **inputcsvheadername.csh:** for checking any block in the csv has left blanked or not. if yes, it flags error and exit.
- **parsingcellfromlibrary.csh:** in this, all the libraries DUTs would be parsed one by one.
- **libraryinformation.csh:** this is the heart of the tool that segregates CELLS into COMBO, SEQUENCIAL, CBUF and TRI. and gives us the netlist of that particular library. And more would be used when we combine different library.

So we got the netlists as well as RTL of muxcell with the help of some rules and templates which designed earlier.

3.1.3 Feature Of Tool

- Less Time consuming to generate.
- Tool with Sanity Checks, any undefined format of input is not acceptable, flash ERROR and Quit to process.
- Easy to debug.

3.2 Synthesis

It is the process of converting design description, that is written in a hardware description language such as verilog or VHDL into an optimized gate level netlist, mapped to a specific technology library. Here, I use Design Compiler for logic synthesis. The steps of the synthesis process are as follows:

1. The input design files for DC are often written using a hardware description language (HDL) such as VHDL or Verilog.

2. Design Compiler uses synthetic, technology libraries or designware libraries and symbol libraries to implement synthesis and to display synthesis results graphically. During the synthesis process, Design compiler i.e. DC translates the HDL description to components extracted from the generic technology (GTECH) library and Designware library. The GTECH library consists of basic logic gates and flip flops. The Designware library contains more complex cells such as adder and comparators. DesignWare library that contains more complex cells, for example, adders and comparators. Both the GTECH and DesignWare libraries are innovation free or technology independent, that is, they are not mapped to a particular technology library. Design Compiler utilizes the symbol library to generate the design schematic.

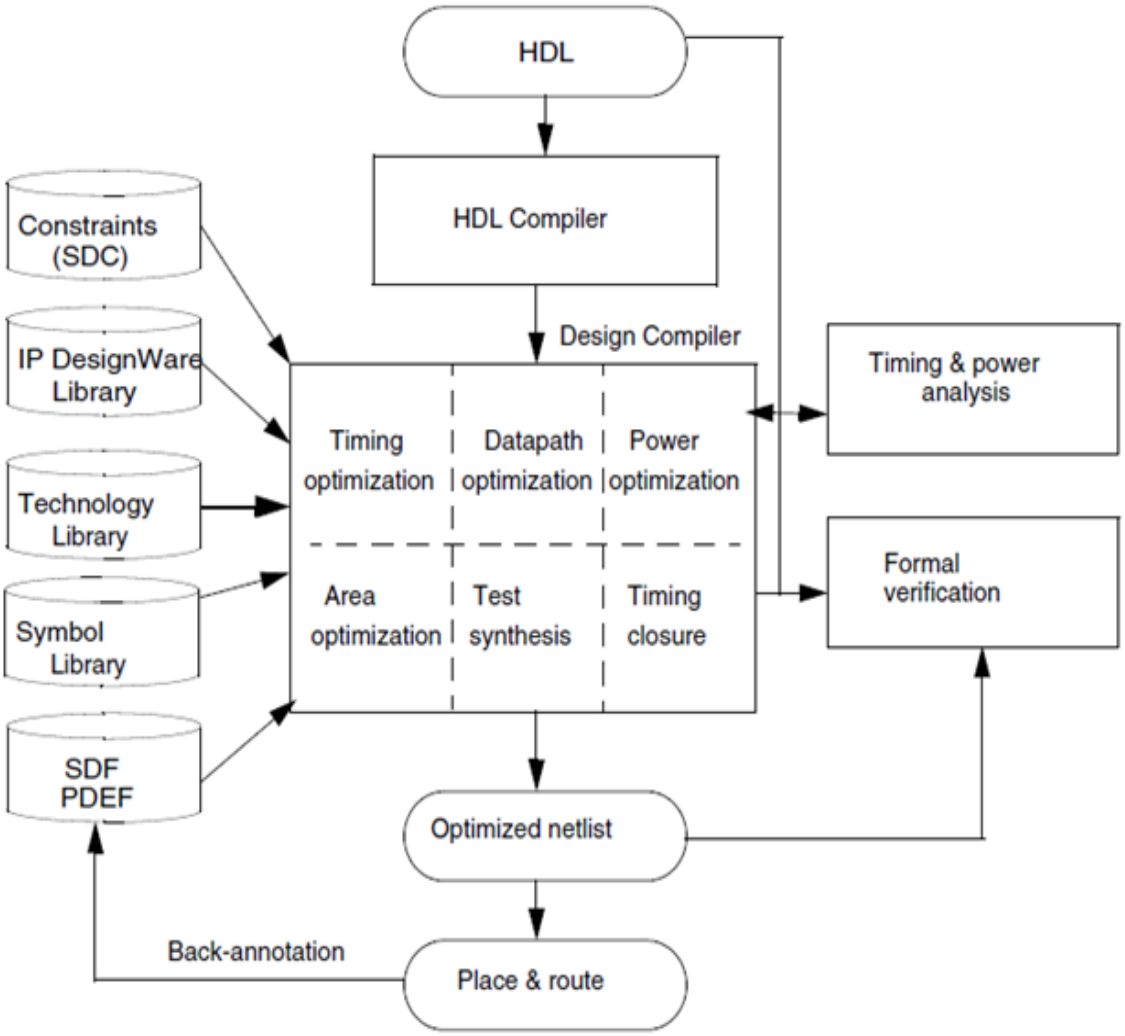

Figure 3.2: Design compiler and Design flow

3. After translating the HDL description to gates, Design Compiler upgrades that means optimizes and maps the design to a specific technology library, known as the target library. The procedure is constraint driven. Constraints are the planner's specifications of timing and environmental limitations under which synthesis is to be performed.

4. After the design is enhanced or optimized, it is prepared for test synthesis. Test synthesis is the procedure by which designers can coordinate or integrate test logic into a design during logic synthesis. Test synthesis enables planners or designer to

guarantee that an design is testable and resolve any test issues ahead of schedule in the design cycle.

Logic synthesis results process is an optimized gate-level netlist, which is basically list of circuit elements and their interconnections.

5. The design is ready for the place and route tools after test synthesis, which place the cells and interconnect cells in the design. after then on the basis of the physical routing, the designer may back-annotate the design with the actual interconnect delays. Design Compiler may then resynthesize the design for more accurate timing analysis.

3.2.1 steps during Synthesis

1. Develop HDL Files

The input files for Design Compiler are written using a hardware description language (HDL). I generate this from the RTL generation tool and write the RTL for High speed interface.

2. Specify Libraries

The link and the target libraries are technology libraries that explains the set of cells and related information, as a example cell names, cell pin names, delay, pin loading, design rules, and all operating conditions.

3. Read Design

Design Compiler can read RTL designs as well as gate-level netlists. Design Compiler uses hardware description language Compiler to read Verilog and VHDL RTL designs. It has a netlist reader for reading Verilog as well as VHDL gate-level netlists. The specialized netlist reader reads netlists faster and also uses less memory than HDL Compiler. Design Compilers uses these to read design files:

- The analyze as well as elaborate commands
- The read file command

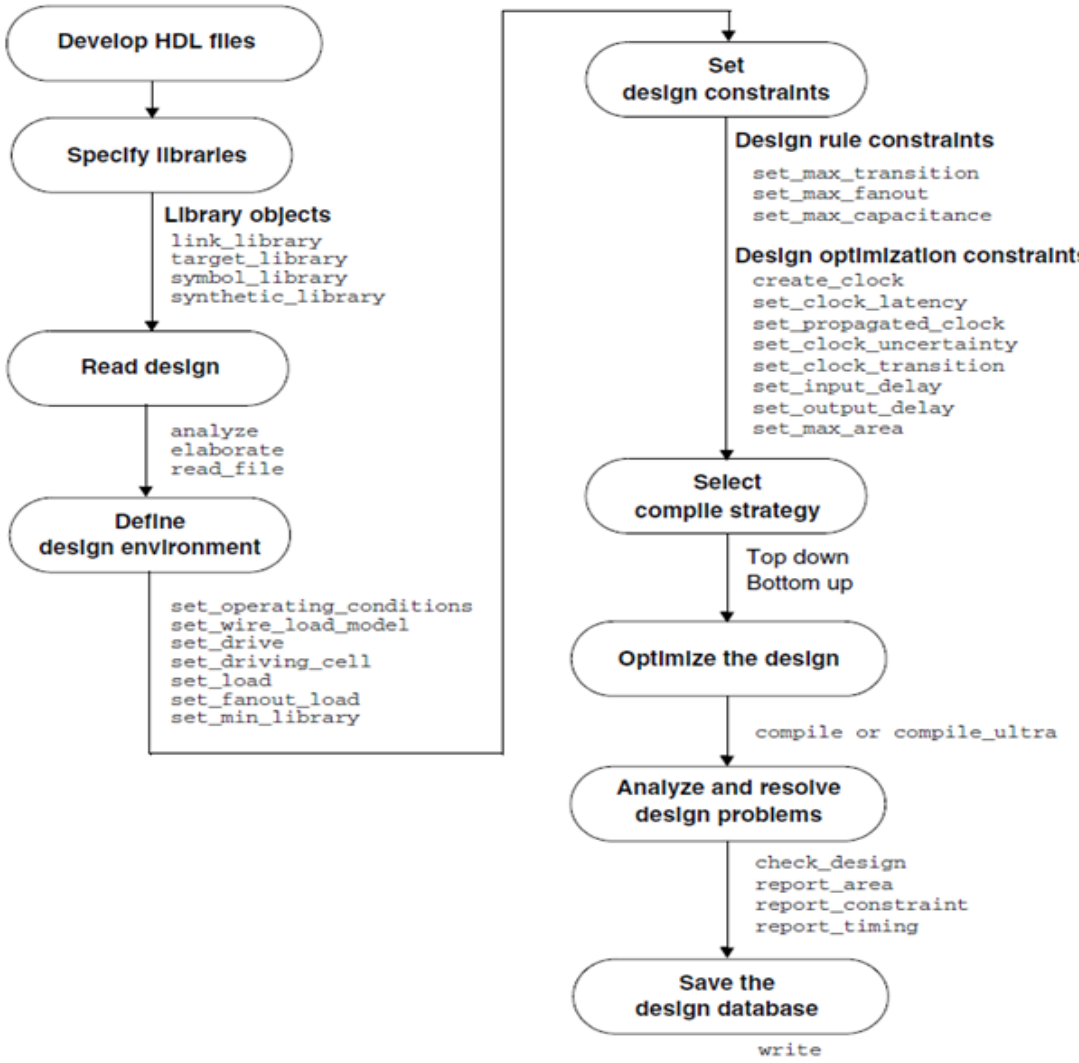

Figure 3.3: Steps During Synthesis

- The read VHDL and read verilog commands. These commands are taken from the read file -format VHDL as well as read file -format verilog commands.

4. Define Design Environment

Design Compiler requires that you show the environment of the design to be synthesized. This model involves the external operating conditions (fabricating procedure, temperature, and voltage), loads, drives, fanouts, and wire load models.

5. Set Design Constraints

Design Compiler utilizes design rules and optimization constraints to control the

synthesis of the design. Design rules are given in the vendor technology library to guarantee that the product meets specification and works in as expected. Typical design rules constrain transition times (set max transition), fanout loads (set max fanout), and capacitances (set max capacitance).

6. Select Compile Strategy

The two essential compiler methodologies that you can use to optimize hierarchical designs are top down and bottom up. In the top-down technique, the top-level design and all its sub designs are compiled together. All environment and constraints settings are characterized as for the top-level design. In the bottom up technique, individual subdesigns are constraints and compiled independently. After compilation, the designs are assigned the dont touch attribute to prevent additionally changes to them during compilation. Until the top-level design is synthesized, compilation process is continued up through the hierarchy. By this strategy, we compile large design because compiler donot need to load all the uncompiled subdesigns into memory.

7. Optimize the Design

We use the compile command or compile ultra command to invoke the Design Compiler synthesis as well as optimization processes.

8. Analyze and Resolve Design Problems

Design Compiler can generate various reports on the results of a design synthesis as well as optimization for example, set area, set constraint, and all timing reports. We use reports to analyze and resolve any type of design problems or you can say, to improve synthesis results. We can use the check-design command to check the synthesized design for consistency results.

9. Save the Design Database

We use the write command to save the synthesized designs. Because Design Compiler does not automatically save designs.

3.2.2 Implementation

For this synthesis, I prepared one csv that contains these information:

- **LIBRARY_GROUP_CELLS:** In this give the library group name. For example, LIB1,LIB2,etc.
- **LINK_LIBRARY:** Give the standard cell core as well as clock (which were required for that particular library) libraries name with its PVT (Process Voltage Temperature) condition, with the .db extension.
- **TARGET_LIBRARY:** Give the standard cell clock and core library (which were required for that particular library) name with its PVT (Process Voltage Temperature) condition, with the .db extension.
- **DONT_USE_DRIVE:** Give the lower limit upper limit of data as well as clock cell number from which we want to optimize.
- **DONT_USE_PB:** Give the PB number which we don't want to optimize.
- **OCC_SLOW_CLOCK_PERIOD:** Give the OCC slow clock period.
- **OCC_FAST_CLOCK_PERIOD:** Give the OCC fast clock period.
- **INPUT_DELAY:** Give the input delay for timing.
- **OUTPUT_DELAY:** Give the output delay.
- **INPUT_TRAN:** Give the input transition width.
- **MAX_DELAY:** Give the maximum delay.
- **LOAD:** Give the output load.
- **MAX_CAP:** Give the maximum capacitance.
- **MAX_TRAN:** Give the maximum transition.

- **CLOCK_UNCERTAINTY_HOLD:** Give the uncertainty hold time period.
- **CLOCK_UNCERTAINTY_SETUP:** Give the uncertainty setup time period.

And, made scripts for synthesize the high speed interface, DFT insertion, dumped a list for dont touch and cuts which are used by back end team not to change their functionality as well as that cells.

3.3 Summary

Here , I synthesized the ALLCELL design including high speed interface and also include the OLT mode in all high speed block RTL by generating scripts for all the block with required constraints and steps which I described earlier.

Chapter 4

Test Pattern Generation

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, FUNC Scan, Auto mode, Auto mode top control and OLT mode.

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.

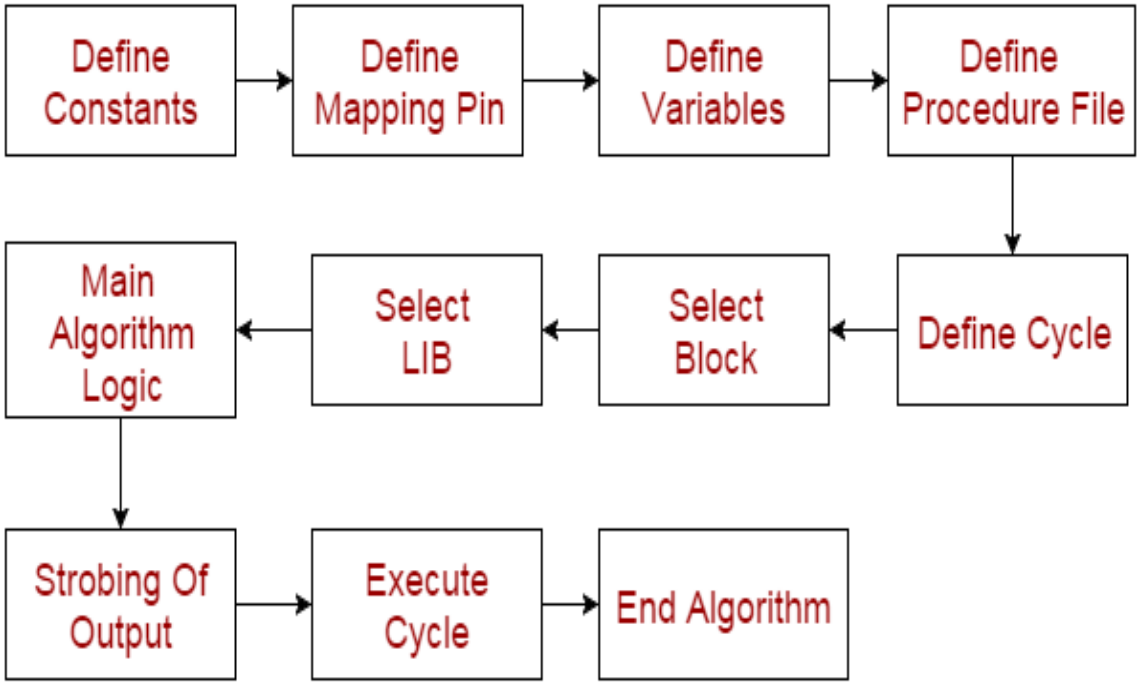

Figure 4.1: Tool flow

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.

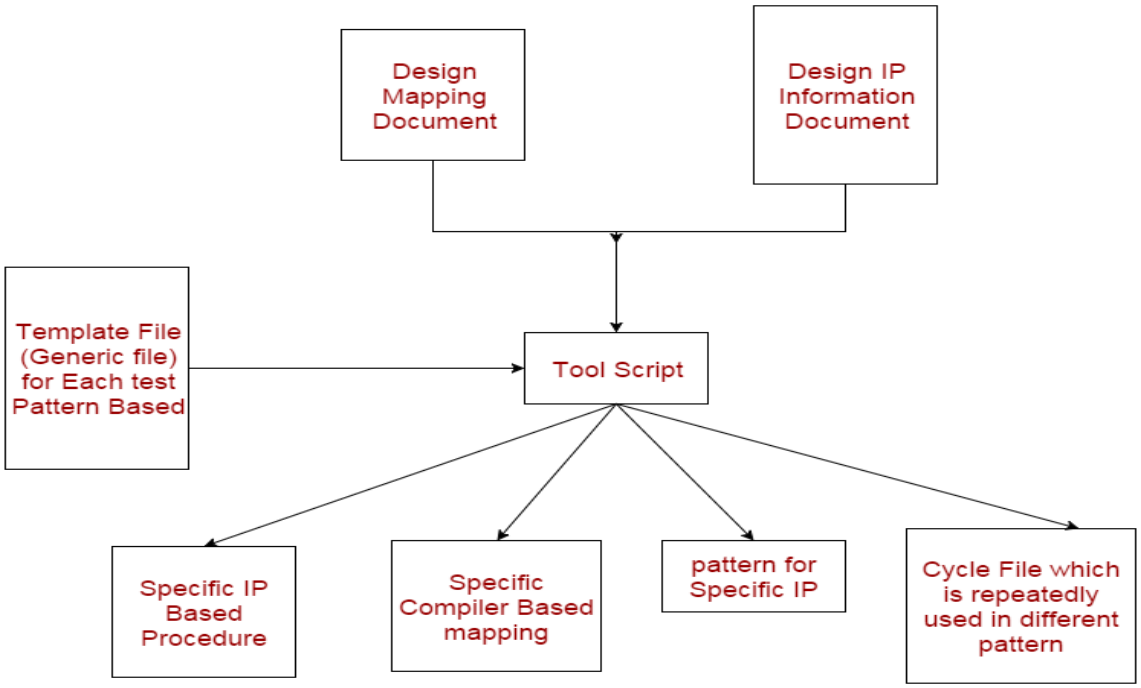

Figure 4.2: Tool flow

4.3.1 Design sheet information:-

- **TOP_PIN:** give information about the top pin(example : 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 STD-CELL_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)
- **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 : ALLCELL , 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.
- We can easily convert this pattern in Verilog Testbench for functional simulation as well as tester pattern for Silicon testing.
- 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.

Chapter 5

Silicon Debug Setup

5.1 ALLCELL Debug Setup

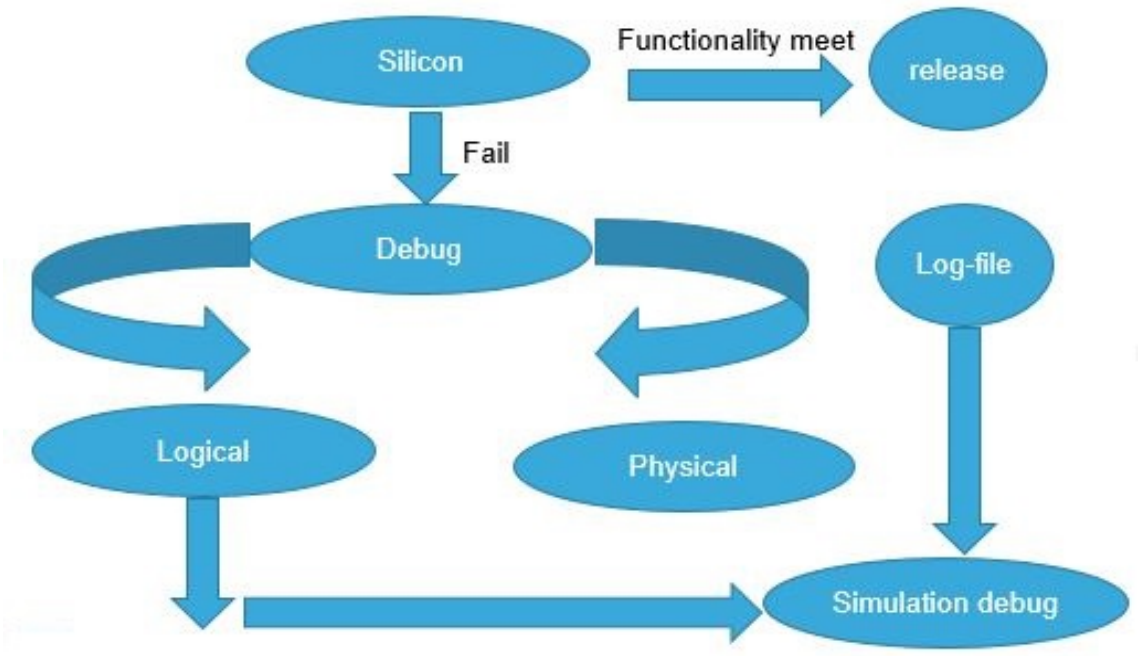

Figure 5.1: Tool flow

When the design is fabricated on silicon and the die goes for wafer or package testing, only the input and output ports are accessible from the external environment, no internal nodes are controllable or observable. Hence, upon functionality failure, if

the intended output is not obtained at the output port, it is very difficult to debug and diagnose the failure.

Hence, to mitigate this issue, we have for this design a "debug setup" methodology which facilitates in silicon failure diagnosis.

In the debug setup, there are two ways of getting to the failing node:

i. Logical approach

ii. Physical approach

Logical Approach:

By design, we are aware of the input connections to each cell and the output of each cell goes into which reference cell. As a result, from the design, the cell name, its input connections, the connecting reference cell, corresponding mux select line are tabulated into a csv file. While simulating the design, we dump a log file containing the input stimuli, status of output signals and also cycle number (timestamp). When these patterns are run on the tester, if there is any failure, a failure log is generated containing the cycle number and failing output port. Now, cycle number is the common parameter between CAD log and tester log. With the help of this failing cycle number, we can co-relate the two logs and design csv file to get the final information : cycle number, which DUT cell is selected in that cycle, the inputs to that cell, the output of the cell and mux select line. Hence, we are able to identify the failing DUT cell with the logical approach, for example:

Instance Name	Cell Name	Clock Cycle	input Pin	output Pin
C1_COMBO/cell1	cell1	0	A	Z
C1_SEQ/cell2	cell2	1	A	Z
C1_CBUF/cell3	cell3	2	A	Z
C1_TRI/cell4	cell4	3	A	Z

Physical Approach:

We also have the information of physical coordinates of every cell from the design GDS. This information can be used later for on-silicon debug purpose.

Chapter 6

Work Contribution and Result

- Automation for RTL generation tool used SHELL and PERL Script.
- Validation of RTL by making the script generic and time consuming.
- Validation of NETLIST design adding high speed interface and OLT mode.
- Stitching the RTL generation tool by synthesis scripts and it takes input from user that they want to run the tool or not, if yes, then it proceed.
- Developed Tool Pattern Generation Tool
- Developed the debug setup for analysing the post silicon failures.
- Generation of this complete flow (RTL GENERATION, SYNTHESIS, FUNCTIONAL VERIFICATION) for different Standard cell projects of 28nm & 40nm technology.

6.1 Simulation Graphs

6.1.1 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.

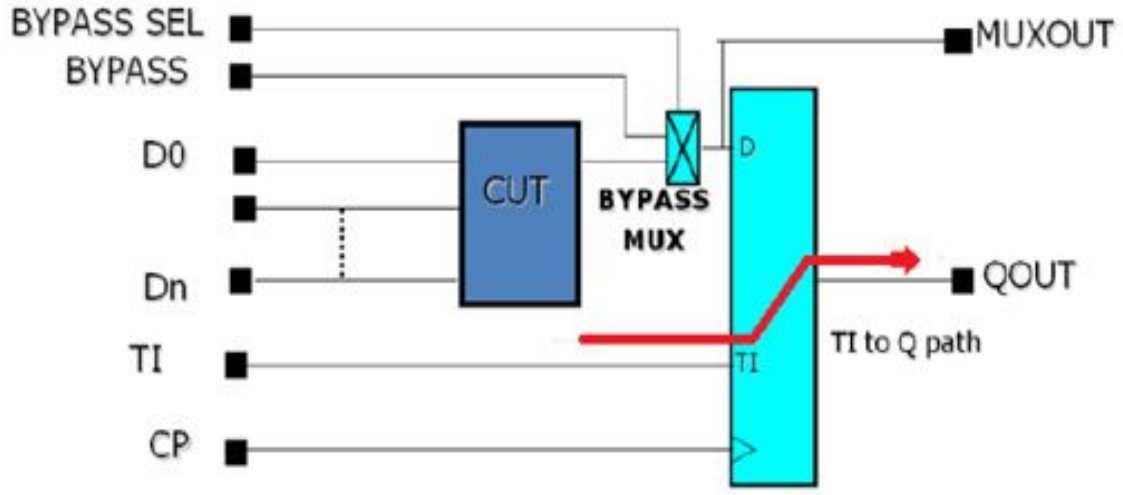

Figure 6.1: ref chain flop for scan0,1,ALL

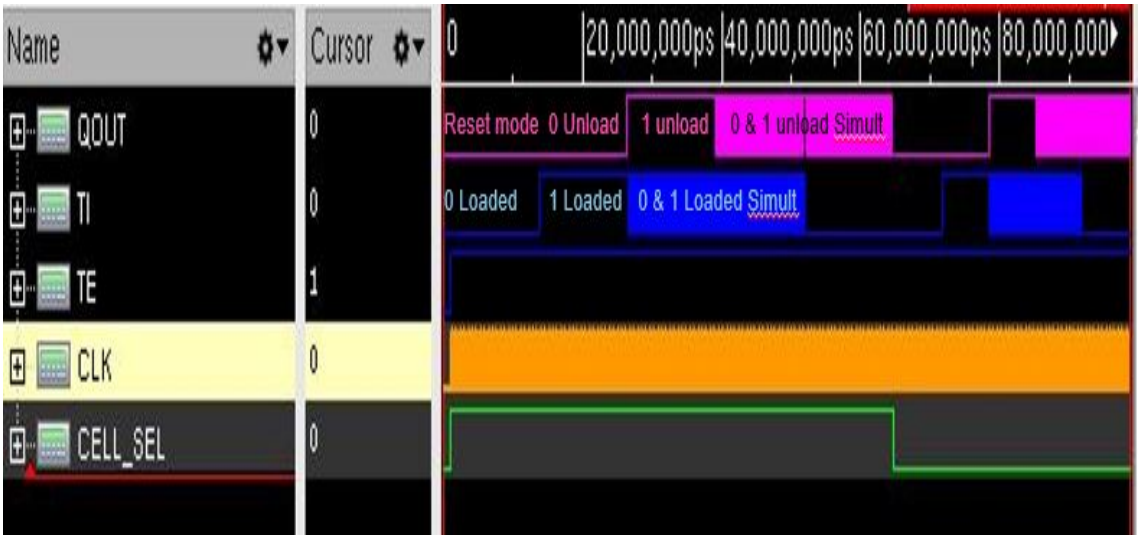

Figure 6.2: Scan All

6.1.2 BYPASS MUX

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

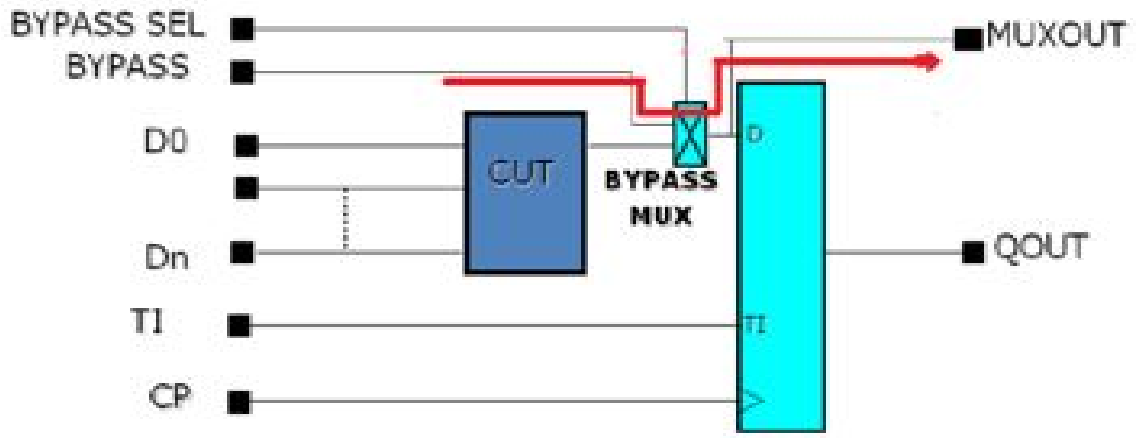

Figure 6.3: ref cahin bypass

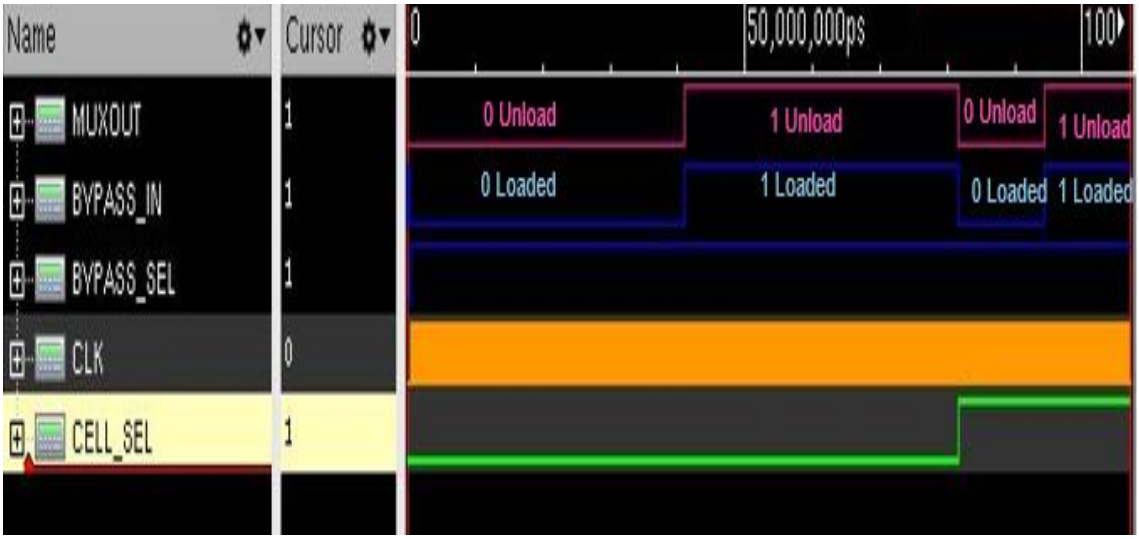

Figure 6.4: Bypass mux

6.1.3 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.

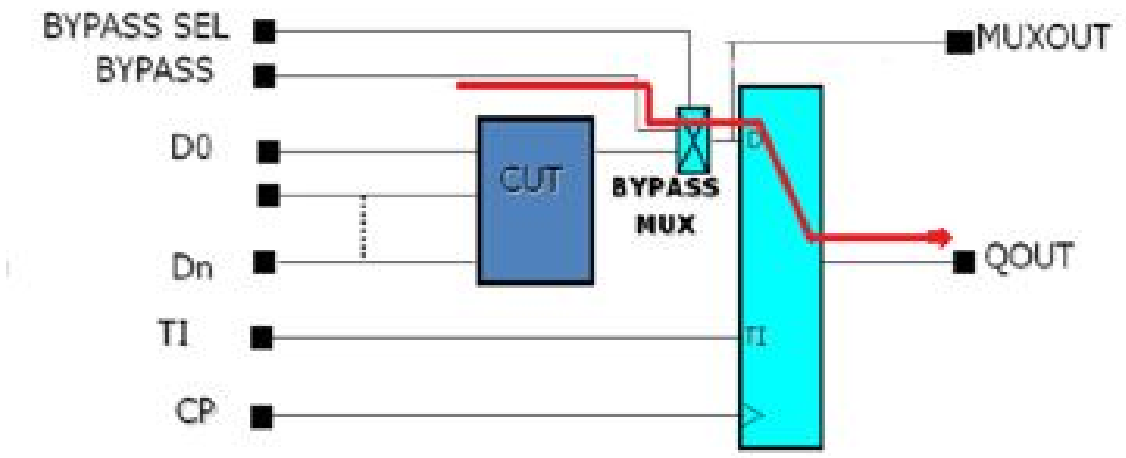

Figure 6.5: Bypass scan

Figure 6.6: Bypass scan

6.1.4 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 âĂŞ 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.

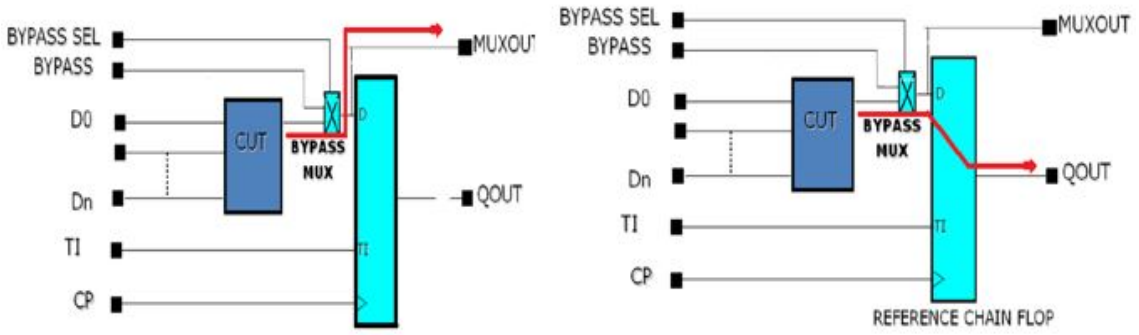

Figure 6.7: Functional Mux and Functional Scan

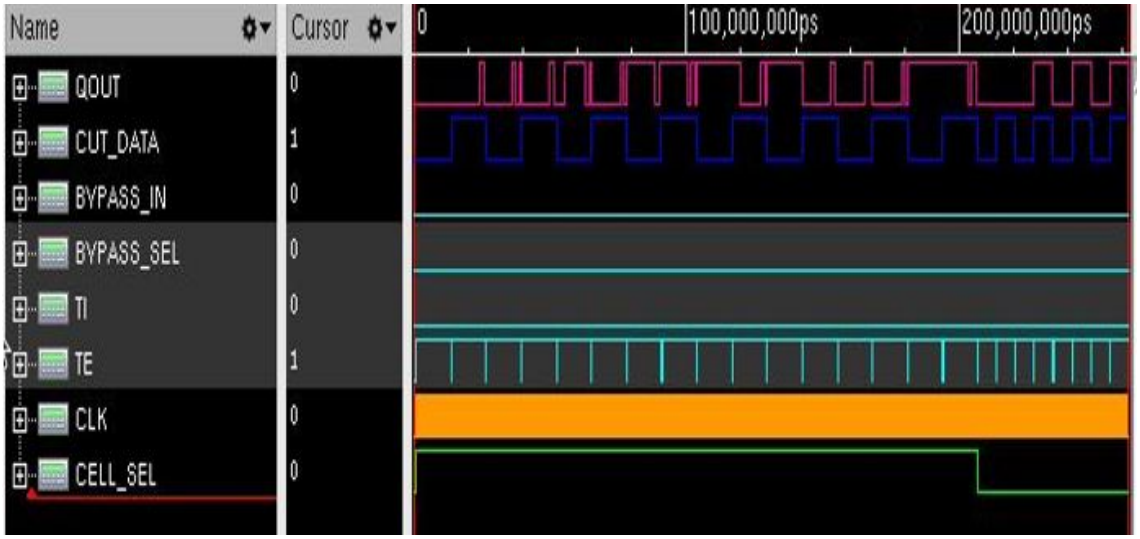

Figure 6.8: Refchain func mux & scan

Chapter 7

Conclusion

7.1 Conclusion

- The proposed test circuitry is designed in such a way that guarantees complete functional excitation. The design not only addresses combinational, sequential and clock gating cells, but it also presents a unique structure for testing tristate cells, which cannot be tested by standard ATPG tools.

- Moreover, every CUT output is observed via two paths multiplexer output and reference flop chain output that ensures design testability and enhancing debug ability in case of scan failure. Also, the HSI block provides a solution to at-speed testing and stress testing with limited number of I/O pads and minimum use of the ATE tools, thus reducing test engineering costs significantly.

- Besides, the automation of the design generation flow and failure diagnosis setup avoids loss of coverage and honours time to market window. Different repetitions and orientations of the structure on a die address the statistical yield aspect.

- Thus, this project provides a complete and full-circle solution to silicon qualification of standard cells.

References

[1] ST Internal Documents

[2] Synopsis, design compiler user guide, version D-2010.03-SP2, JUNE 2010

[3] http://ece-research.unm.edu/jimp/pubs/vts2005 atspeed.pdf

[4] R. P. Ribas, S. Bavaresco and M. Lubaszewski: ĂIJEfficient Test Circuit to Qualify Logic CellsĂİ, IEEE International Symposium on Circuits and Systems Âů June 2009.

[5] Jiang Jianhua, Liang Man, Wang Lei and Zhou Yumei: ĂIJAn effective timing characterization method for an accuracy-proved VLSI standard cell libraryĂİ, Journal of Semiconductors, Vol. 35, No. 2. February 2014.

[6] Jaafar K. Al-Frajat, Wameedh Nazar Flayyih, Roslina Binti Mohd Sidek, Khairulmizam Samsudin and Fakhrul Zaman Rokhani: ĂIJArea efficient test circuit for library standard cell qualificationĂİ, IEEE 5th International Conference on Energy Aware Computing Systems Applications,10.1109/ICEAC.2015.7352210,2015

[7] Renato P. Ribas, Vinicius Callegaro, Marcelo Lubaszewski, Andre lvanov and Andre I. Reis: âĂIJCircuit Design For Testing Standard Cell LibrariesâĂİ.

[8] Jagannath Keshava, Nagib Hakim and Chinna Prudvi: ĂIJPost-silicon validation challenges: How EDA and academia can helpĂİ, IEEE Design Automation Conference, 2010.

Methodology for silicon qualification of standard cells

ORIGINALITY REPORT

%5	%	%5	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	"Introduction to Hardware Security and Trust", Springer Nature, 2012 Publication	%2
2	Lecture Notes in Electrical Engineering, 2013. Publication	%1
3	Fahmy Hafriz bin Mohamed Sultan, Zuraini binti Dahari, Yien Yien Koh, Neil Da Cunha, Jia Tian Ng. "Chapter 4 Development of At-speed Interconnect Test to Capture Marginal Open Defect on FPGA", Springer Nature, 2017 Publication	<%1
4	Khuong Tuyen Huynh. "Effect of hypercapnia on intracellular pH regulation in a rainbow trout hepatoma cell line, RTH 149", Journal of Comparative Physiology B, 05/03/2011 Publication	<%1
5	"Test Pattern Generation", Frontiers in Electronic Testing, 2002 Publication	<%1
6	MacNamee, C., and I. Indino. "DFT: Scan	

testing issues and current research", 25th IET Irish Signals & Systems Conference 2014 and 2014 China-Ireland International Conference on Information and Communities Technologies (ISSC 2014/CIICT 2014), 2014.
Publication

<%1

7 Lin, Wei, and Wen Long Shi. "An On-Chip Clock Controller for Testing Fault in System on Chip", Applied Mechanics and Materials, 2013.
Publication

<%1

8 Shi, Youhua, Nozomu Togawa, Masao Yanagisawa, and Tatsuo Ohtsuki. "Improved Launch for Higher TDF Coverage With Fewer Test Patterns", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2010.
Publication

<%1

9 R. P. Ribas. "Efficient test circuit to qualify logic cells", 2009 IEEE International Symposium on Circuits and Systems, 05/2009
Publication

<%1

10 Natarajan, Sudhakar, and Rajasekar Natarajan. "Effective Suppression of Conducted Electro Magnetic Interference in DC-DC Boost Converter Using Field Programmable Gate Array Based Chaotic Pulse Width Modulation Switching", Electric

<%1

Power Components and Systems, 2014.
Publication

11	Saiah Michèle. "Abstracts", Canadian Journal of Anaesthesia, 05/1992 Publication	<%1
12	A, Sreejithlal, Ajith Jose, A Shooja, and B Manoj Kumar. "IEEE 1451.2 based smart sensor system using ADuc847", 2015 International Conference on Communication Information & Computing Technology (ICCICT), 2015. Publication	<%1
13	Kevin Whitton, X. Hu, Cedric Yi, Danny Chen. "An FPGA Solution for Radiation Dose Calculation", 2006 14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, 2006 Publication	<%1
14	H. Balachandran. "K longest paths per gate (KLPG) test generation for scan-based sequential circuits", 2004 International Conference on Test TEST-04, 2004 Publication	<%1

EXCLUDE QUOTES	ON	EXCLUDE MATCHES	OFF
EXCLUDE BIBLIOGRAPHY	ON		