Develop a Generic Test Wafer Generator Which Can Be Used to Characterize Standard Cells and Other Test Structures

Kartik Gupta.

ABV-Indian Institute of Information
Technology and Management, Gwalior
Morena Link Rd, IIITM Campus,
Gwalior, Madhya Pradesh 474015
guptakartik279@gmail.com

***Abstract*—This work presents the design procedure and the architecture of simple, efficient and easy-to-use test generator for characterizing and validating any set of Standard Cells. Characterization and verification are accomplished considering performance, functionality, process parameter variation for both timing and power dissipation values. The resulting structure is relatively useful for cell library verification at different levels.**

Keywords—Standard Cells, Logic Gates, Test Generation, Testability, Test Wafer.

I. INTRODUCTION

In Very Large Scale Integrated (VLSI) circuits we are facing tremendous testing problem, because test generation and fault simulation of VLSI packages are very costly and incompetent. Since the cost of test generation increases nonlinearly with respect to the size of the device, we have to test the device after every step of completion.

For standard cell based transistor test structures, it is important to retain the environment around DUT as close to that in novel standard cell as possible.

Cell Characterization is the process of simulating a standard cell with an analog simulator or an automated characterization tool to extract this data and convert into a format that other tools can use. Cell Characterization can be completed by analog simulation using HSPICE simulator, whose output can be used to generate the timing characterization data, we have to find the result in standard format like liberty file format.

Types of testing are following-

Design debug or verification testing.

Burn-in (stress test).

Wafer sort or probe test.

Functional test.

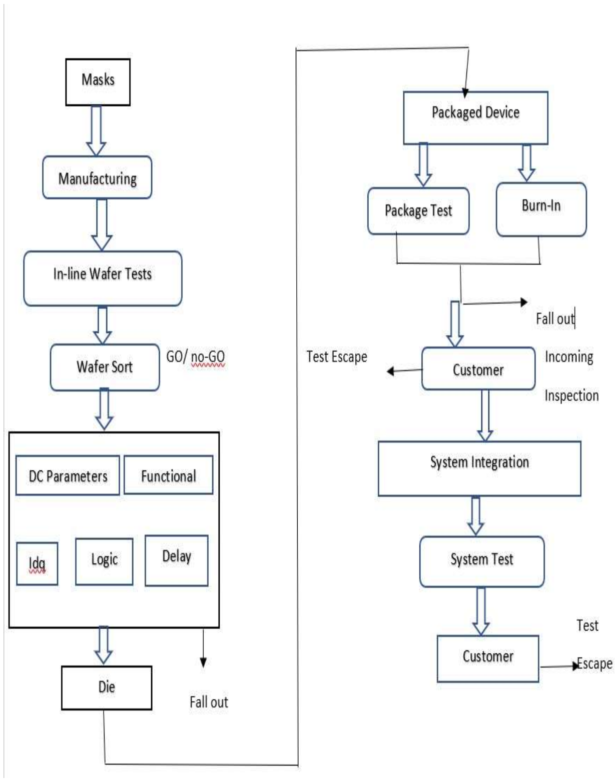
We have to use EDA tools to generate test wafer, which is used for testing the standard cells. So, we do some steps to create a test generator as follows-

Placement.

Routing.

Floor planning

Test Flow-



The test program, the digital test vectors and the analog test waveforms are required once the chip is mounted in the tester. CAD tools to automate the generation of the test programs.

To create cell library, each cell is a previously designed, fabricated, and tested logic cell. like, each type of logic cells is first designed using the CAD tool. The CAD tool then predicts the response of the designed logic cell, to one or more input signals and in particular shows the propagation delays of the output signal relative to the input signals.

This work presents an effective way for evaluating and validating Standard logic cell libraries. The major contributions include an advanced design methodology for such a kind of test vehicle, as well as a simple and flexible multi-operating mode circuit architecture. The resulting circuit is quite useful for cell library verification at different levels, in the EDA environment and on silicon prototyping. The proposed methodology can be applied for analysis taking into account the logic gate functionality, timing performance, power consumption.

Our main aim is to provide a solution with limited number of I/O pads and minimum use of Automatic Test Pattern Generation (ATPG) tools, reducing significantly the test engineering costs.

The applications of proposed method are-
Process Development, to verifying process on new machine, and for Porting a process to a smaller node etc.

Design Requirement

a) every cell under test has to be instantiated at least once to guarantee it is fully stimulated during the circuit operation.

b) every cell in the library is considered as distinct cell. So, we have to check the characteristics of cell.

Process Flow

The standard cell characterization flow begins with following basic input files. First, a netlist of the cell in SPICE format is required, It is generated by Popcorn[3]. Second is layout of the netlist. The layout tool, Magic[4] is used to generate the files in this work. Third, information on the device technology that is dependent on the specific technology and the process utilized is required (in our case we use osu180 technology). And finally, a file is required that consists of information on the operational temperature and voltage used on the cells. For this work we require open source PDK file. We require a virtual environment for the specific technology, this environment used to measure temperature, pressure. In our case we create this environment by following command.

Python3 -m venv my-librecell-env

the above command is used in linux platform.

Table I
Input files used for the test generator

Directory or File	Contains Information on
MODELS	Device technology
NETLIST	Transistor netlist in spice format
TEMPLATE	Slew, output load, cell names
.tcl	Files with tcl command
.bash	Files with bash command
.pl	Files with perl command

Table II
Standard Cells list of Test generator

Cells
INVX1, INVX2, INVX4, INVX8 AND2X1, AND2X2, OR2X1, OR2X2, NAND2X1, NAND3X1, NOR2X1, NOR3X1, XNOR2X1, XOR2X1, FAX1, HAX1, BUFX2, BUFX4, CLKBUF1, CLKBUF2, CLKBUF3 DFFNEGSQ, DFFPOSSQ, DFFSRSQ, LATCHSQ
Xn indicates drive strength, where n=1, 2, 3, 4, SQ indicates sequential circuit

The input files are summarized in Table I and list of cells used are illustrated in Table II. And output files are stored in the outlib folder, it contain all .gds and .lef files. Below, steps are introduced for creating standard cell characterization of an inverter in osu180 technology.
These steps can be repeated for any cells in any technology by using proper related files.

• Within the MODELS sub folder, the file with .sp extension has the accurate device technology file with the process corners used for nMOS, pMOS and CMOS transistors.

• Within the NETLIST sub folder, netlists contain the layout netlist in SPICE format.

• Within the TEMPLATE sub folder, the template osu.tcl file contains all the input signal slew and output capacitive loads, the lower and upper slew values, the name of the cell used.

• Within the char.tcl file, all the environmental variables like voltage, temperature are included. Some Tcl scripts provided by the CDS have been modified and reused to create the necessary results.

• Within TECH folder all the technology files are stored.

• A Makefile has been created to hide the complexity of calling different commands and making it easy to run the complete process.

We also require one shell script which constantly monitor the progress of the standard cell generation.

Open Source Tools

1) Popcorn.
2) Magic.
3) Qflow.[5]
4) Ngspice.[6]
5) Tcl.

Preferred programming language for development:-
Python, Perl, TCL, Bash, C, C++

References

[1] A. Ricci, I. De Munari and P. Ciampolini, “An evolutionary approach for standard-cell library reduction,” Proc. of ACM Great Lakes Symposium on VLSI (GLSVLSI), pp.305-10, 2007

[2] http://eceresearch.unm.edu/jimp/vlsi_test/slides/html/overview1.htm

[3] https://github.com/chipforge/StdCellLib

[4] http://opencircuitdesign.com/magic/

[5] http://opencircuitdesign.com/qflow/

[6] http://ngspice.sourceforge.net/

[7] https://github.com/libresilicon/PerlRiver