A Practical Approach to Obtain Defect Matrix for Integrated Circuit Testing

LARISSA SOARES
Federal University of Paraíba
Department of Electrical Engineering
Cidade Universitária, n/n João Pessoa
BRAZIL
larissa.soares@cear.ufpb.br

CLEONILSON SOUZA
Federal University of Paraíba
Department of Electrical Engineering
Cidade Universitária, n/n João Pessoa
BRAZIL
protasio@cear.ufpb.br

Abstract: Testing of integrated circuits (ICs) is always a challenge because the continuous miniaturization process and consequently increase of transistor density in microelectronic industry. Nowadays, the industry has to handle with defects that traditional testing approaches can not detect. The result of this imprecise testing process is the increase number of defective ICs that reach the end consumer. To improve the quality of IC testing, a new approach of fault modeling is being adopted which is not based on transistor or logic gate level, but in the IC layout perspective itself. This paper describes the meaning of testing based on layout perspective, particularly, Cell-Aware Testing (CAT) methodology, and a practical approach to obtain the matrix of defects, in which is the set of test patterns to each modelled fault coming from CAT, and that is the CAT's main result. Experimental simulation results show the matrix of defects obtained for a specific standard cell that can be immediately used by an ATPG.

Keywords: Defect, Fault-model, Layout-perspective, Defect insertion

1 Introduction

Throughout the fabrication process of integrated circuits (ICs), physical defects like opens, shorts and bridges can occur. To identify and detect such defects, fault models have been proposed since it is very difficult to detect all kind and variations of actual physical defects. Fault models like stuck-at [1], bridging [2] and transition delay [3] are the most well-known fault models and have been used as classical models of IC testing mainly because their effectiveness and their technology independence. However, because the continuous miniaturization process and consequently increase of transistor density in microelectronic industry, testing of current ICs is turn to be more challenge and the industry has to handle with defects that traditional testing approaches can not detect. In this way, defectives ICs are passing unnoticed in tests and the result of this imprecise testing process is an increase of the number of defective ICs that reach the end consumer.

To face such a problem a new fault modeling approach based on the IC layout perspective [4] has been studied which is not based on IC gate-level or transistor-level, as traditional ones, but on the actual layout of the IC under test [5, 6].

Take in consideration digital IC testing and, particularly, test of **standard cell library** based ICs, which are based on predefined layout cells in order to ease the process of automated digital IC development, the traditional gate-level based testing tools are not capable to detect all kind of cell-internal defects since they focus on external responses out of the specific cell, specifically, on gate interconnections. Additional, traditional fault modeling approaches do not consider analog simulations based on SPICE netlist and parasitic components [7]. Furthermore, some approaches model physical defects on transistor level by means of SPICE simulations but without considering parasitic objects [8], [9] and are ineffective when dealing with millions of transistors [10, 11]. In general, traditional testing tools do not show how low-level defect information can be used to achieve better and high efficient fault models [7].

In other hand, it is clear that if the fault model considers the IC layout rather than only gate or transistor circuit diagram more accurate must be the testing process. For example, in Fig. 1 there are three defects (circles) that could cause some fault, but only one causes a actual fault: the one that connects two nets causing a short [12]. Probably, this fault-causing defect must be detected by traditional gate-level based testing tools. However, the two non-fault-causing defects shown in Fig. 1, even though do not cause a short, can lead to defective circuit since they can change the capacitances between the respective nets causing dynamic errors.

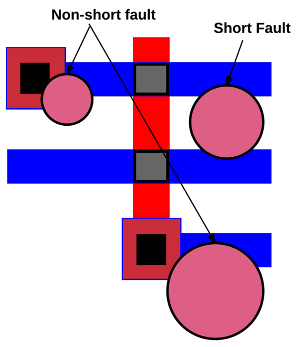


Figure 1: Defect and fault in a layout.

A state-of-the-art methodology that considers IC layout perspective is the **Cell-Aware Test** (CAT) that deals with IC post-layout transistor-level netlist including its parasitic components [7] which claims to be a defect-based methodology that targets layouts with multimillion transistors. The CAT methodology takes advantages of the transistors and parasitic components from the IC netlist and from IC layout to obtain a more comprehensive fault models [7].

In this work, take in consideration a given standard cell layout, a proposed and simple fault injection circuit (emulating fault-causing defects) is used to obtain the defect matrix (*i.e.*, the mapping between defects and the possible test patterns that detect each of them) to be used in CAT-based defect simulation.

Another contribution of this paper is that all simulations were doing on a tool different from the tool that the original CAT methodology was based [4, 5, 6, 7, 13], that is, Mentor Graphics. Cadence was used instead.

In this paper, Section 2 describes the CAT methodology. In Section 3, it is proposed a way in order to obtain the defect matrix and the complete IC defect matrix in shown in Section 4. Section 5 concludes the paper.

2 Cell-Aware Test

This section described a new defect-oriented methodology, called Cell-Aware Test (CAT) [4]. As shown in Fig. 2, the CAT flowchart starts with the layout extraction step, which reads the cell layout data file (called F1) of an individual library cell and creates a SPICE transistor netlist including parasitic elements like resistors and capacitors [7]. These SPICE parasitic netlist information is stored in a file called F2.

Next, an analog simulation step takes place in the CAT flow. In this step, from the SPICE netlist file F2, it is chosen some prone-to-fault parasitic components to be considered defects and these considered defects are stored in file F3. Additionally, transistor faults like stuck-open and stuck-on are also stored in file F3. Each defect in F3 is single simulated [7].

As an example, Fig. 3 shows a layout of a multiplexer (MUX 31X4) where it can been seen its inputs ($D0, D1$ and $D2$), its selection inputs ($S0$ and $S1$), and its output (Z). Take this MUX in consideration, a list of possible considered defects as stored in F3 can seen in Fig. 4 where 48 defects have been considered and d_i is a particular defect of the MUX.

From the list of MUX potential defects stored in F3, it was derived 48 additional netlists, each one from a considered defect, and an additional one from the golden netlist [4]. Knowing this, each one from these 49 netlists is simulated applying all possible inputs to the circuit (namely, 2^5 inputs). The output responses are processed and stored in the defect matrix file (called F4), as shown in Fig. 2. At the end, the output responses in F4 are used to compute the fault coverage and the Cell-Aware Model (CAM) in file F5. For each

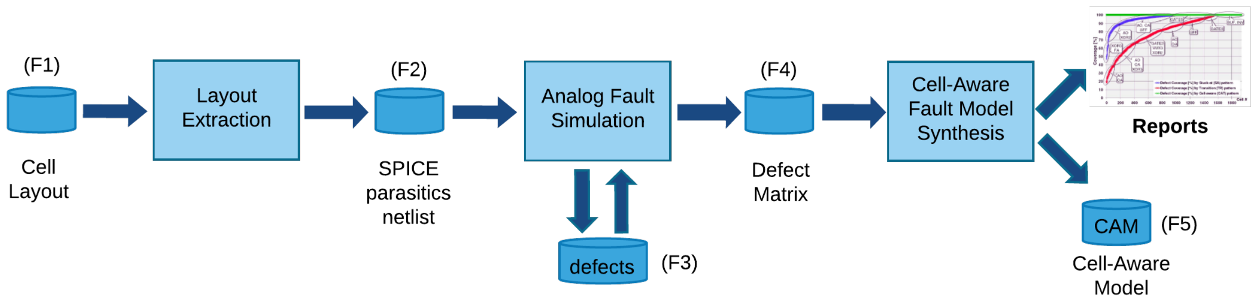


Figure 2: Cell-Aware methodology flow.

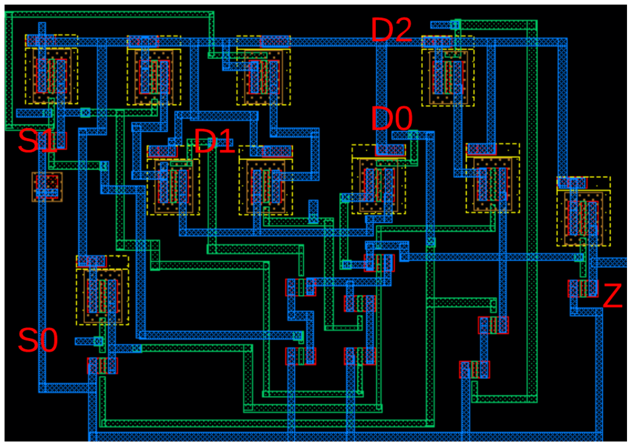


Figure 3: Layout of a MUX31X4.

defect, file F5 contains the obtained **defect matrix** that store all input conditions for detecting the corresponding defect [4].

d1 = S0N, gnd	d17 = D0, gnd	d33 = net38, D1
d2 = S1N, gnd	d18 = vdd, gnd	d34 = net81, D0
d3 = net65, gnd	d19 = Z, net65	d35 = net38, D0
d4 = net57, gnd	d20 = S1, S0N	d36 = S1, S0
d5 = net19, gnd	d21 = S1N, S1	d37 = D2, S1
d6 = net81, gnd	d22 = net65, S1	d38 = S0, D1
d7 = net38, gnd	d23 = S0N, S0	d39 = vdd, S0N
d8 = net85, gnd	d24 = net81, S1	d40 = vdd, S1N
d9 = net35, gnd	d25 = S1, net38	d41 = vdd, net65
d10 = net31, gnd	d26 = D2, S1N	d42 = D0, S0
d11 = net69, gnd	d27 = net81, D0	d43 = net38, vdd
d12 = Z, gnd	d28 = net65, D2	d44 = vdd, Z
d13 = S1, gnd	d29 = net38, S0	d45 = vdd, S1
d14 = S0, gnd	d30 = S0N, D1	d46 = vdd, S0
d15 = D2, gnd	d31 = net81, D1	d47 = vdd, D2
d16 = D1, gnd	d32 = S0N, D0	d48 = vdd, D1

Figure 4: Defect list extracted from multiplexer CAT layout.

Input	**d1**	**d2**	**d3**	**d4**	**...**	**d41**	**d42**	**d43**	**d44**	**d45**	**d46**	**d47**	**d48**
00000	-	-	-	-		-	-	-	D	-	-	-	-
00001	D	D	-	-		D	D	D	-	D	D	-	-
00010	-	-	-	-		-	-	-	D	-	D	-	-
00011	-	D	-	-		D	-	D	-	D	-	-	-
00100	-	-	-	-		-	-	-	D	D	-	-	-
00101	D	-	-	-		D	D	D	-	-	D	-	-
00110	-	-	-	-		-	-	-	D	D	D	-	-
00111	-	-	-	-		D	-	D	-	-	-	-	-
01000	-	-	-	-		-	-	-	D	-	-	-	D
...													
11111	-	-	-	-		D	-	-	-	-	-	-	-

Figure 5: Defect matrix example.

As shown in Fig. 5, the first column of the resulted defect matrix shows the circuit inputs ($D0, D1$, and $D2$, $S0$ and $S1$) and its other columns show the result of each considered defect. The rows mean that when a given circuit input generates a resulting output that is different from the golden response, the column-labeled respective defect is detected and 'D' is shown on the respective entry. For example, only input $'00000'$ detects the defect $d44$.

3 Proposal of a Defect Insertion Circuit

All schematics, layouts, simulations and results have been created and obtained in Cadence. One of the reasons of those steps is to simulate our proposal on defect insertion using a simple scheme of switching, in which is used to insert defects in the simulated circuit and turn it into failed one. The switching circuit works injects defects in circuit schematic, one defect per time, in order to see how this defect injection affects the circuit response.

For example, from the defect list shown in Fig. 4, the defect $d48$ can be injected using the proposed defect injection circuit shown in Fig. 6. In this example, a short between input $D1$ is shorted to VCC that correspond to the injection of defect $d48$.

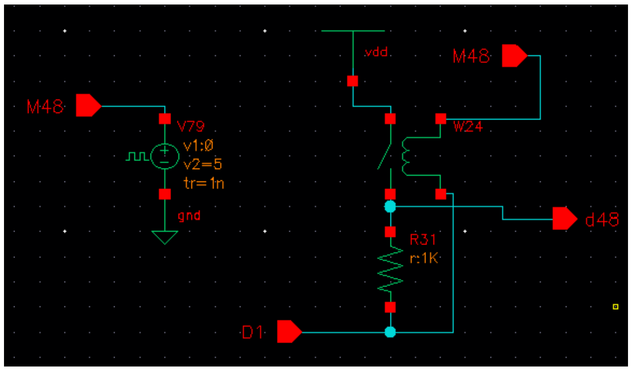


Figure 6: Defect insertion with keys.

In Fig. 7, it is shown all the 48 switching circuit, each representing the 48 possible defects, given from CAT, in the circuit. Such switching circuit proposal has the advantage of simulating all faults in the same time and obtain the responses in order to compare with the gold one.

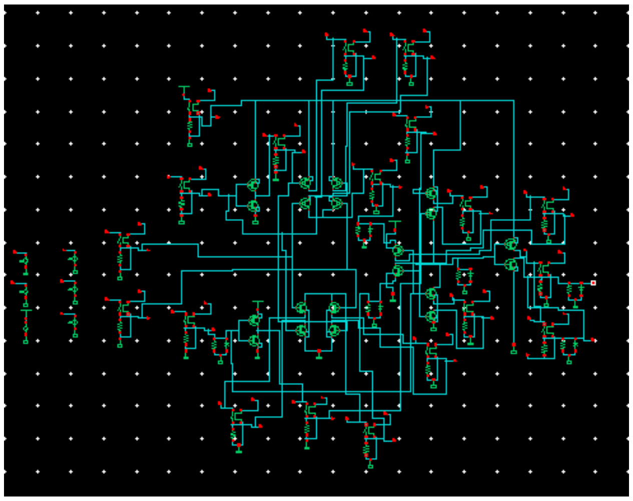


Figure 7: MUX schematic with Complete Fault Injection.

4 Simulation Results

4.1 Computation of the Defect Matrix

In order to obtain the Defect Matrix of a given digital cell, it was simulated a multiplexer with 3 inputs $(D0, D1, D2)$, 2 selectors $(S0, S1, S2)$ and the output Z, the same considered in [7] and its 48 defects. The schematic is made of only transistors (pfets and nfets), as can be seen in Fig.8.

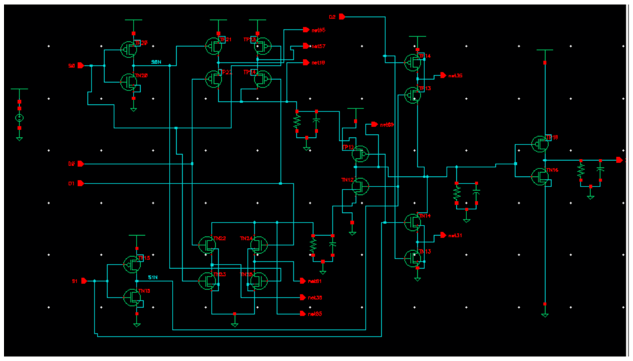


Figure 8: Schematic of a multiplexer.

4.2 Defect-free circuit

The first step of the simulation was to insert all the 32 possible inputs, $(00000 - 11111)$ into the defect-free circuit to obtain the golden output.

Fig. 9 shows the waveform of the inputs and selection signal. The first waveform, $S0$ has a period of $10\mu s$, the second waveform, $S1$ has a period of $20\mu s$, the third waveform, $D0$ has a period of $40\mu s$, the fourth waveform, $D1$ has a period of $80\mu s$ and the fifth one, $D2$ has a period of $160\mu s$. It can be observed that in every $5\mu s$ an input is applied. For example, in the first $5\mu s$ of all waveforms there is a logic level $'0'$, so the input is $'00000'$. In the second input, we have $'00001'$ and so on until $'11111'$.

After a $160\mu s$ simulation, it is obtained the first output sequence, which is the golden output, as can seen in Fig. 10. In the simulation, the high level is $Vdd = 5V$ and the low level is $Vss = 0V$.

4.3 Results with Defective circuit

In order to simulate defects, each defect from the defect list shown in Fig. 4 is modelled as a short

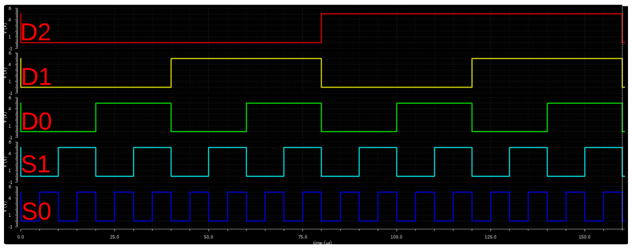


Figure 9: Multiplexer input.

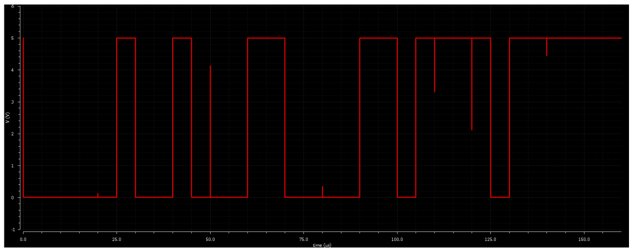


Figure 10: Multiplexer golden output.

or precisely a low-valued resistive bridge using the swithing circuit.

In this way, it was simulated 48 short defects using the proposed defect injection circuit shown in Fig. 6.

It was need $160\mu s$ to insert all 32 possible inputs to each injected defect. At the first $160\mu s$ the defect $d1$ was injected. From $160\mu s$ to $320\mu s$, the $d2$ was inserted, and so on until the last one, which was the defect $d48$. In total, it was achieved $160\mu s \times 48\,(defects) = 7680\mu s$ as simulation time consumption.

As an example, the output response of the injection of defect $d48$ is shown in Fig. 11.

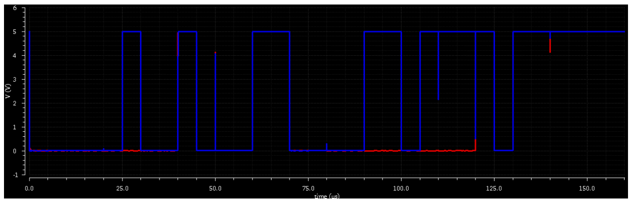


Figure 11: Output with d48 defect.

Finally, with all the inputs and defects inserted, the complete defect matrix can be obtained.

The complete defect matrix for the multiplexer is the one with all the possible inputs in the column, all the defects that are extracted from the layout in the rows. When an input into the circuit produces an output response that is different from the golden one, this input detected such a defect. When it occurs, a **'D'** is placed in the defect matrix.

In Fig. 12, it is shown the complete matrix obtained with the simulations from Cadence. It was considered a threshold value higher than 50% of VDD to be considered undefined. For example, if the response expected was $5V$ (high logical level), and the response was $2.5V$, we considered the output undefined **'U'**.

These results was compared with the defect matrix obtained from CAT [4] and the obtained defect matrix shows the same results with the same inputs.

5 Conclusion

In this paper, it was described the new approach of integrated circuit fault modeling that is based on the its layout perspective. It was addressed the Cell-Aware Testing methodology and the description of a proposed simple switching circuit used to inject defects from CAT. Experimental simulation results show the applicability of the proposed circuit.

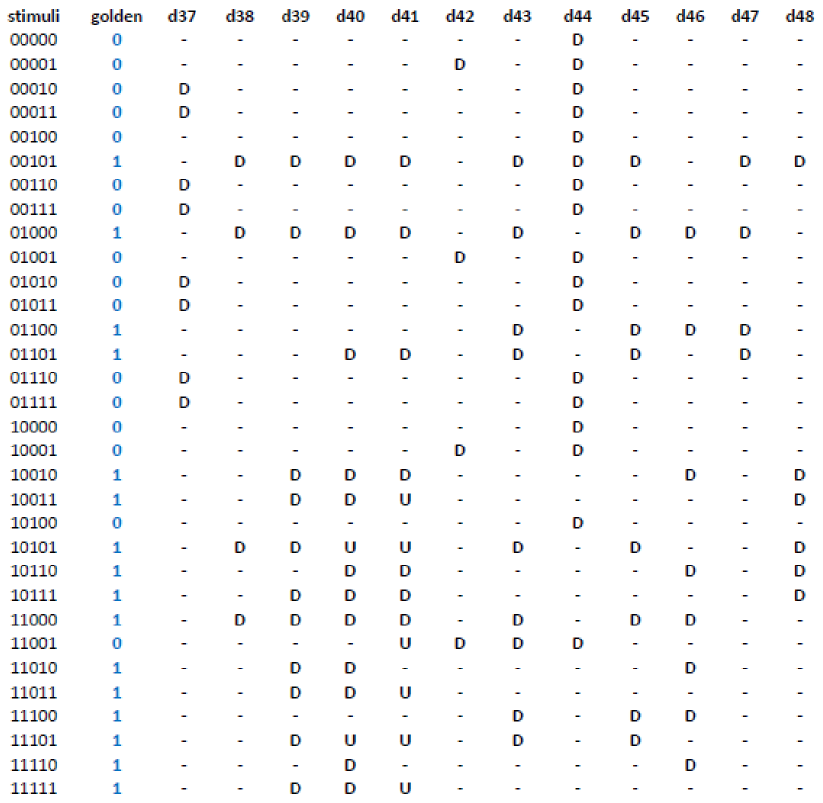
References:

[1] K. Y. Mei, Bridging and Stuck-At Faults *IEEE Transactions on Computers*, Vol. C-23, No 7, 1974.

[2] F. J. Ferguson, T. Larrabee, Test Pattern Generation for Realistic Bridge Faults in CMOS ICs *IEEE Design and Diagnostics of Electronic Circuits and Systems*, 2007.

[3] J. A. Waicukausi, E. Lindbloom, B. K. Rosen, V. S. Iyengar, Transition Fault Simulation *IEEE International Test Conference*, 1994.

[4] F. Hapke, R. Krenz-Baath, A. Glowatz, J. Schloeffel, H. Hashempour, S. Eichenberger, C. Hora, D. Adolfsson, Defect-Oriented Cell-Aware and Fault Simulation for Industrial Cell Libraries and Designs *IEEE*, pp. 1–10 2009.



stimuli	golden	d37	d38	d39	d40	d41	d42	d43	d44	d45	d46	d47	d48
00000	0	-	-	-	-	-	-	-	D	-	-	-	-
00001	0	-	-	-	-	-	D	-	D	-	-	-	-
00010	0	D	-	-	-	-	-	-	D	-	-	-	-
00011	0	D	-	-	-	-	-	-	D	-	-	-	-
00100	0	-	-	-	-	-	-	-	D	-	-	-	-
00101	1	-	D	D	D	D	-	D	D	D	-	D	D
00110	0	D	-	-	-	-	-	-	D	-	-	-	-
00111	0	D	-	-	-	-	-	-	D	-	-	-	-
01000	1	-	D	D	D	D	-	D	-	D	D	D	-
01001	0	-	-	-	-	-	D	-	D	-	-	-	-
01010	0	D	-	-	-	-	-	-	D	-	-	-	-
01011	0	D	-	-	-	-	-	-	D	-	-	-	-
01100	1	-	-	-	-	-	-	D	-	D	D	D	-
01101	1	-	-	-	D	D	-	D	-	D	-	D	-
01110	0	D	-	-	-	-	-	-	D	-	-	-	-
01111	0	D	-	-	-	-	-	-	D	-	-	-	-
10000	0	-	-	-	-	-	-	-	D	-	-	-	-
10001	0	-	-	-	-	-	D	-	D	-	-	-	-
10010	1	-	-	D	D	D	-	-	-	-	D	-	D
10011	1	-	-	D	D	U	-	-	-	-	-	-	D
10100	0	-	-	-	-	-	-	-	D	-	-	-	-
10101	1	-	D	D	U	U	-	D	-	D	-	-	D
10110	1	-	-	-	D	D	-	-	-	-	D	-	D
10111	1	-	-	D	D	D	-	-	-	-	-	-	D
11000	1	-	D	D	D	D	-	D	-	D	D	-	-
11001	0	-	-	-	-	U	D	D	D	-	-	-	-
11010	1	-	-	D	D	-	-	-	-	-	D	-	-
11011	1	-	-	D	D	U	-	-	-	-	-	-	-
11100	1	-	-	-	-	-	-	D	-	D	D	-	-
11101	1	-	-	D	U	U	-	D	-	D	-	-	-
11110	1	-	-	-	D	-	-	-	-	-	D	-	-
11111	1	-	-	D	D	U	-	-	-	-	-	-	-

Figure 12: Defect matrix.

[5] F. Hapke, W. Redemund, A. Glowatz, J. Rajski, R. Krenz-Baath, J. Schloeffel, M. Wittke, H. Hashempour, S. Eichenberger, Defect-Oriented Cell-Aware and Fault Simulation for Industrial Cell Libraries and Designs *IEEE Test Conference (ITC)*, pp. 1–10, 2010.

[6] F. Hapke, J. Schloeffel, Introduction to the defect-oriented cell-aware test methodology for significant reduction of DPPM rates *IEEE*, pp. 1–6, 2012.

[7] F. Hapke, W. Redemund, A. Glowatz, J. Rajski, M. Reese, M. Hustava, M. Keim, J. Schloeffel, A. Fast, Cell-Aware Test *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1396–1409, 2014.

[8] P. Dahlgren, P. Liden, A fault model for switch-level simulation of gate-to-drain shorts *IEEE Comput. Soc. Press*, pp. 414–421, 1996.

[9] R. C. Aitken, Finding defects with fault models *Int. Test Conference*, pp. 498–505, 1995.

[10] M.K. Reddy, S.M. Reddy, P. Agrawal, Transistor Level Test Generation for MOS Circuits *IEEE*, pp. 825–828, 1985

[11] Abraham, Saab, CRIS: A test cultivation program for sequential VLSI circuits *IEEE Comput. Soc. Press*, pp. 216–219, 1992.

[12] G. A. Allan, J. P. Elliott, A. J. Walton, A Layout-Driven Yield Predictor and Fault Generator for VLSI *IEEE Transactions on Semiconductor Manufacturing, Vol. 6, No 1*, 1993.

[13] F. Hapke, J. Schloeffel, H. Hashempour, S. Eichenberger, Gate-Exhaustive and Cell-Aware pattern sets for industrial designs *IEEE*, pp. 1–4, 2011.

[14] I. Bubel, W. Maly, T. Waas, P.K. Nag, H. Hartmann, D. Schmitt-Landsiedel, S. Griep,

, AFFCCA: A tool for critical area analysis with circular defects and lithography deformed layout *IEEE Comput. Soc. Press*, pp. 10–18, 1995

[15] G. A. Allan, A. J. Walton, Efficient critical area estimation for arbitrary defect shapes *IEEE Comput. Soc*, pp. 20–28, 1997.

[16] G. A. Allan, J. P. Elliott, A. J. Walton, Critical Area Extraction for Soft Fault Estimation *IEEE*, 1998.

[17] G. A. Allan, A. J. Walton, Efficient extra material critical area algorithms *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1480–1486, 1997

[18] M. Gkatziani, R. Kapur, Q. Su, B. Mathew, R. Mattiuzzo, L. Tarantini, C. Hay, S. Talluto, T. W. Williams, Accurately Determining Bridging Defects from Layout *IEEE Design and Diagnostics of Electronic Circuits and Systems*, Vol. 11, No 1, 2007.