Defect-Oriented Mixed-Level Fault Simulation of Digital Systems-on-a-Chip Using HDL

M. B. Santos and J. P. Teixeira
IST-UTL / INESC
jct@zeppelin.inesc.pt

Abstract

The validation of high-quality tests requires Defect-Oriented (DO) fault simulation. The purpose of this paper is to propose a methodology for mixed-level DO fault simulation, using HDL. A novel tool, `veriDOFS`*, is introduced. Structural zooming is performed only for the system module in which the faults are injected. Verilog models for bridging and line open defects are proposed for intra-gate and inter-gate faults. Design hierarchy is exploited, by pre-computing a test view of each cell in a library. The good trade-off accuracy / tractability, as well as the computational efficiency of the new tool are demonstrated by means of structural benchmarks up to 100,000 transistors and 300,000 realistic faults.*

1. Introduction

Complex Systems-on-a-Chip (SOC) use predefined macros, or cores, extensively. As SOC complexity increases, so do product quality requirements. Defect Levels (or escape rates) in the order of 50 ppm (parts per million) or less are now common, in safety-critical (or application-critical) markets. This puts a heavy burden on the computational efficiency of the test preparation process.

Product complexity drives the design activity towards higher levels of abstraction, and to HDL. Product quality drives the test activity towards lower levels of abstraction, and to more accurate fault models. As design & test is more and more an integrated activity, HDL-based test preparation becomes mandatory.

Proposed HDL-based fault simulation methodologies and tools typically use high level fault models [1-9]. However, the correlation between high-level faults and structural faults is difficult to prove. Multi-level fault simulators have been proposed for LSA fault models [10] and others [6] up to RTL with stuck-on and stuck-open fault models. The authors have previously demonstrated the feasibility of DO fault modeling using VHDL [4,5]. Nevertheless, the computational efficiency thus obtained is limited.

Quality requirements quest for Defect-Oriented Test (DOT) [11]. However, this is only feasible if complexity of the fault simulation process is adequately dealt with. Mixed-level (behavioural and structural) simulation alleviates the problem [10], as the single-fault approach allows us to restrict the structural simulation to the module in which the fault is injected. Judicious fault sampling [13] also reduces the fault simulation effort dramatically. However, efficient fault models for physical defects are still required.

It is a well known fact that 100% Line Stuck-At (LSA) fault coverage may lead to significantly lower *Defects Coverage (DC)*, namely bridging (BRI) and Line-Open (LOP) faults coverage. In out test environment, DO fault list generation is performed by an IFA-based extraction tool, `lobs` [13]. With large BRI and LOP fault lists, how can we validate a test pattern to be applied to a large IC?

The purpose of this paper is to present a set of DO fault models for CMOS realistic BRI and LOP faults for efficient fault simulation using a Hardware Description Language (HDL), like Verilog or VHDL. Moreover, a new mixed-level fault simulator, `veriDOFS`, is presented, implementing the proposed fault models with Verilog.

The paper is organised as follows. Section 2 summarises the main features of the fault simulation methodology. In section 3 and 4, realistic fault models are proposed for BRI and LOP faults. Section 5 introduces the DO fault simulator. In section 6, simulation results are presented. Finally, in section 7 the conclusions are presented.

2. Fault Simulation Methodology

The proposed methodology uses the following underlying assumptions:

- Test quality assessment of a SOC typically requires the quality assessment for modules up to 50,000 gates
- A standard cell layout cell is used
- DO routing faults are extracted by an IFA-based tool, like `lobs[13]`. Cell faults are pre-extracted and characterized in a *test view* of the cell library. Due to their low incidence, adjacency faults are not considered

- High-level design data is used to retain the behavioural description for all the SOC, except for the module where fault injection occurs
- Whenever the module complexity justifies it, a stratified fault sampling procedure [13], also implemented in lobs, is used to generate a manageable fault list
- In order to demonstrate the feasibility of the proposed methodology, a commercial HDL simulator is used as kernel simulator engine
- Test quality evaluation is performed by the Weighted Fault Coverage (WFC) metric [4], as the extracted DO faults have a probability of occurrence
- The accuracy of the simulation results is guaranteed, by using the concept of Damaged Sub-Circuit (DSC) [4], as the smallest sub-circuit which restores digital values at its output port, in the presence of the fault.

3. Bridging Faults Models

Several types of realistic BRI faults must be taken in to account for digital CMOS fault simulation:

- BRI between logical nodes;
- BRI between logical nodes and power lines (V_{DD} or V_{SS});
- BRI between electrical nodes and power lines (V_{DD} or V_{SS});
- BRI between logical and electrical nodes
- BRI between electrical nodes.

All BRI faults are assumed to be hard shorts (R=0), based on data showing that the majority of likely BRIs exhibit resistance values lower than 2KΩ [14].

As referred, in a standard cell type layout, faults are classified according to their layout topology, as: routing, cell faults and adjacency faults. **Routing faults** are extracted from the layout and their circuit behaviour modelled taking into account the drive and load parameters of the cells involved. **Cells faults** are pre-extracted from the cell library layout, modelled, and their model parameters saved for subsequent instantiation. The corresponding fault models, translating circuit-level into logic-level data, are referred as the **test view** of the cell. Other authors have proposed fault models for BRI faults between logical nodes [12,15-18]. Here, an extension of the biased voting model [12] is proposed.

Biased Voting

As an evolution of the *voting* model [15], the *biased voting (bv)* was proposed [12] for solving most of the **BRI conflicts between logical nodes.** The simulator must read a table of normalised conductance (relative to the N transistor) as a function of the analogue voltages (rows) and the pull-up/down topologies (columns).

Extended Biased Voting

For BRIs involving electrical nodes, feedback networks may be formed. As the biased voting model cannot model these topologies, an *extended biased voting* model is proposed. The extended biased voting model deals with these feedback topologies, by adding columns to the table of relative strengths. The added topologies are shown in Fig. 1. The corresponding column values are obtained by means of SPICE electrical simulations for a target technology and cell library. In this figure, all unconnected transistor gates are considered at V_{DD} or V_{SS} for the N and P networks, respectively.

However, when, in an active BRI fault, a PU/PD path includes both types of transistors (referred as a complex path), the extended *bv* table does not contain the corresponding behaviour. Moreover, such a path can not be added to the table because P and N transistors dimensions are not correlated. The table needed for the *bv* model, for the specific transistors dimensions of that BRI, can be stored in the test view of the cell in question.

Cells BRIs fault model, depending on the BRI activating condition, can include:

1. Cell logical nodes names with external PU/PD.
2. Strength of the internal PU/PD with a topology in the extended *bv* model table.
3. A table with the conductance values for the *bv* model computed for an internal PU/PD topology.
4. A table with the output voltage as a function of the BRI nodes voltage.
5. A constant voltage for the cell output.
6. A LSA0 or LSA1 information.

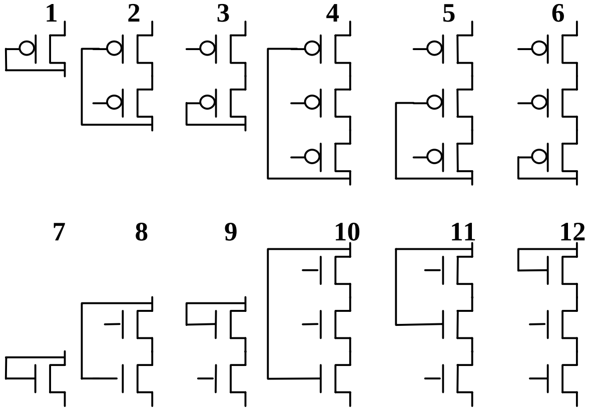

Figure 1: Added network topologies to the biased voting model.

All *routing BRIs* are modelled using the biased voting (*bv*) model. In order to use the *bv* model, some cell electrical parameters must be known. For each cell a *test view* is created where the stored relevant information is: (1) cell output pull-up and pull-down strength information: W/L and *n* (column of the *bv* table), for each possible input combination, and (2) logic threshold voltage for each input.

BRI Models Examples

In Fig. 2 a cell example is presented that exhibits three electrical nodes W, V and Z. This cell is used as a test vehicle to illustrate the modelling procedure for all possible BRIs between logical and electrical cell nodes. BZ bridging exemplifies all **gate-to-drain like BRIs** (AW, AV, AZ, BV, BZ and CZ). For this type of BRI, there are two different activating conditions that must be modelled separately.

If B=1 and D=0 there will be an active path between VDD and GND through PU, T_4 and T_6 (eventually T_5 and T_7, depending on A and C). In this situation there are two logical nodes, B and Y with analogue voltages. Node B voltage computation using the biased voting table is not feasible due to the existence of complementary transistors in the pull-down path, as such transistors dimensions are not correlated. Node Y voltage computation using the biased voting is not feasible as the pull-up of Y contains transistors of different cells (whose dimensions are not correlated). The solution, to use the *bv* model, is to pre-compute and store, in the test view of the cell, node B pull-down conductance table (model information item 3) for each activating condition. Additionally, in order to take PU in consideration in node B voltage computation, node B must figure as model information item 1. If A=0, T_1 is also a pull-up path for node B and thus model information item 2 must be added.

The second activating condition occurs for ABCD=000X. Here, the conducting path is T_1, T_2, T_3 and PD. In this case the **extended biased voting model** can be used considering the pull-up network marked in Fig. 1 as item 5. After node B voltage is known, node Y voltage is obtained using the table with model information item 4 that must also be pre-computed and stored in the test view of this cell.

Let us consider now **the gate-to-source like bridgings**: DZ, DV, DW, CV, CW and BW. For BW, if ABCD=00XX, the active path is T_1 and PD. The *bv* model is applicable and model information item 1, item 2 and item 4 are required.

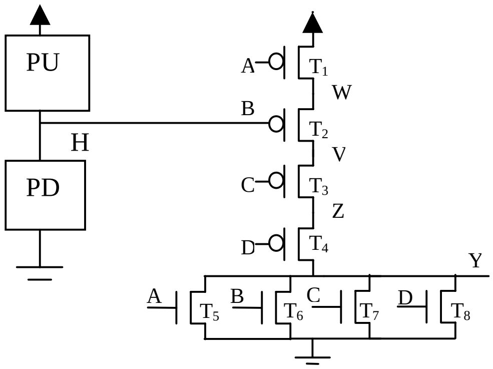

Figure 2: Cell for bridging between electrical and logical node exemplification.

4. LOP Faults Models

LOPs in power lines or in a pull-up/down transistor path can be modelled as conditional LSA faults. The conditions for these faults activation can be a cell inputs logical combination and/or a previous logical value in a cell node.

LOPs in logical nodes routing or LOPs causing floating gates are assumed to be LSA0 or LSA1. The approximation is based on the assumption that floating gate MOS transistors voltage drift (due to g-d/g-s capacitive coupling) is slow, as compared with the test clock period. These faults are considered detected if both LSA are detected. In floating gate LOPs one of the corresponding LSA fault can lead to a analogue voltage at the cell output. In such case, the analogue voltage can be pre-computed and saved in the corresponding cell test view. The proposed model, due to the lower LOP fault incidence in CMOS circuits [19-21], deliberately does not take into account charge sharing effects and transient paths to V_{DD} or V_{SS} effects [22].

LOP Models Examples

In Fig.3a example, the LOP is detected if AB_{t-1}=00, AB_t=10 and Y LSA1 is observable. These fault detection conditions can be saved in a *test view* of the cell.

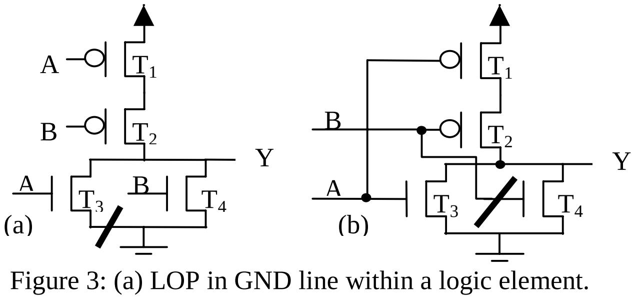

Figure 3: (a) LOP in GND line within a logic element.
(b) Floating gate line-open example.

Fig. 3b shows an example where, if the floating-gate is simulated in logic 0, T_4 is OFF and the fault is detected if Y_{t-1}=1, AB=0X and Y LSA1 is observable. If the floating-gate is simulated in logic 1, T_4 is ON and the analog voltage in Y can be computed if AB=00.

5. Fault Simulation

The name veriDOFS stands for "**veri**log **D**efect **O**riented **F**ault **S**imulator". The major features of veriDOFS are:

1. Verilog structural and behavioural circuits description;
2. Efficient fault simulation of large digital modules (combinational and sequential);
3. Support of LSA, and realistic LOP and BRI fault models with weighted fault lists;
4. "Biased voting" precision simulation of BRI and LOP faults;
5. Automatic fault list generation using a cell library test view ;
6. Voltage *hard* and *potential* detection and I_{DDQ} fault coverage computation.

Items 1 and 2 are supported by the use of a commercial mixed-level Verilog circuit simulator (verilog-XL from Cadence), as a background tool. The fault simulator was developed using the PLI interface of this simulator. Fault injection is performed exclusively on structural parts of the circuit.

Single faults in digital ICs only produce changes on the logical state of a small number of nets in the circuit. This allows the implementation of an efficient fault simulation technique. In fact, as the fault simulator is event driven, the Verilog primitive "force" (used to inject the fault in the required nodes) will trigger a reduced number of events to simulate. For fault detection decision, not only the primary outputs are monitored, but also the nodes which state is determinant for fault model application are checked. This is necessary to detect oscillation conditions in feedback BRI faults [16]. If the logic difference introduced by the fault is observable but oscillation may be present, the fault is considered potentially detected. Afterwards the forced nets are "released" and the circuit will return to the "good circuit" logic values, being ready for the next fault to be injected.

For sequential circuits fault simulation, it is necessary to force the "state variables" of the circuit for each test vector during fault injection and simulation. Additionally, not only the primary outputs, but also the signals that produce the "next state" are monitored. The described injection technique allows **multiple nets forcing** for each fault. Additionally, veriDOFS **fault injection is transparent** for the rest of the simulator routines, (primary inputs assignments, good circuit output save, circuit simulation, output analysis, etc) which allows the easy and modular integration of additional fault models.

A characteristic of the commercial Verilog simulator used as kernel simulator is that a forced and released node will then be slower simulated. To overcame this inconvenient, veriDOFS is capable of status saving and resuming, so that, after the simulation of a large number of faults (and correspondent slow down of many forced nodes), the simulation state can be saved and resumed with all the nodes in accelerated simulation state. Moreover, this feature is very useful for pattern simulation partitioning, in order to control the memory required for the simulation.

6. Results

At present, the test view of a testable cell library, IdLib10 [23] has been characterised. In Table 1, fault simulation results and simulator performance data is shown for several ISCAS benchmark [24,25] circuits. For the larger (sequential) circuit, s38417, the complete extracted fault universe (BRIs) contained about 300,000 faults. Hence, fault sampling [13] was also used. After fault simulation with the sampled faults, a confidence interval of [93.7%; 95.8%] (for a confidence level of 90%) was obtained for the weighted fault coverage (WFC) of the complete set. All vectors were randomly generated and full-scan capability was assumed for sequential circuits. Simulation was carried out in a SunSpark station 10, with 160 MB RAM. Fault simulation of s38641 circuit illustrates the simulator capability to deal with real size modules (100,000 transistors) and large fault lists. In Table 2, WFC and fault simulation performance are presented for the different classes of faults. Y represents a cell output node, A and B represent cell inputs, x is electrical node and VDD is a generic power or ground node. Table 2 shows that BRI faults between cell inputs (BRI A-B) are difficult to detect and WFC is smaller than the LSA FC.

Circuit	#faults	#vec.	WFC [%]	Time[s]
c432	789	1000	99.5	30.6
c1908	6571	10000	98.1	1660
s641	2586	5000	90.8	1427
S38417	298351	600	93.1	42756
s38417	3035	10000	95.4	13774

Table 1: Fault simulator performance.

Results of multi-level FS, so far, indicate that the majority of effort is associated to DO fault injection and simulation in *structural* parts; hence, the gains in retaining a behavioural description for the larger portion of the chip will be ascertained in larger examples.

circuit	c432	c432	c1908	s526
pattern	32 det.	1000 rnd	1700 rnd	12577 rnd
LSA FC	98.9	98.6	97.2	99
BRI Y-x	48.4	48.4	77.9	98.7
BRI A-Y	82.3	98	94.7	87.8
BRI A-B	75	75.2	33.8	62.4
BRI Y-Y	93.6	97.2	96.2	99.1
LOP Y-A	97.3	96.9	93.4	98.2
LOP VDD-Y	100	100	94.9	100
LOP VDD-x	82.3	82.3	90.8	81.25
Real. LSA	100	100	98.8	99.3
Total WFC	94.6	95.1	80.7	88.4
# Real. Faults	1794	1794	9039	2047
Sim. Time [s]	116	454	4112	15241

Table 2: Fault coverage for different fault classes.

7. Conclusions

A mixed-level DO fault simulation for SOCs, using HDL, was presented. Verilog models for realistic BRIs and LOP faults have been proposed. All the suggested models are easy to integrate in a simulation tool and have affordable computational costs.

The Verilog fault simulator tool, `veriDOFS`, is capable of dealing with large size circuits, fault lists and/or test patterns and illustrates the practicability of the suggested fault models for BRI and LOP realistic faults. The fault extraction costs are reduced and fault models are simplified by means of a cell library test view. When high quality test is required, `veriDOFS` can thus accurately compute the WFC and is used in a virtual DO test environment.

References

[1] J.R. Armstrong, F. Lam, P.C. Ward, “Test Generation and Fault Simulation of Behavioral Models”, Joel M. Schoen (Ed.), Performance and Fault Modeling with VHDL, pp.240-303, Prentice-Hall, 1992.

[2] S. Gosh, T.J. Chakraborty, “On Behavior Fault Modeling for Digital Systems”, Journal of Electronic Testing: Theory and Applications, n.2, pp.135-151, 1991.

[3] T. Riesgo, J. Uceda, “A Fault Model for VHDL Descriptions at the Register Transfer Level”, Euro DAC with EURO-VHDL ´96, pp.462-467, September 1996.

[4] F. Celeiro, L. Dias, J. Ferreira, M.B. Santos, J.P. Teixeira, “Defect-Oriented IC Test and Diagnosis Using VHDL Fault Simulation”, Proc. ITC, pp. 620-628, 1996.

[5] M. Santos, N. Vicente, M.B. Santos, J.P. Teixeira, “VHDL Modeling of BRI Defects in CMOS Integrated Circuits”, VHDL User´s Forum in Europe, pp.93-103, April 1997.

[6] Wolfgang Meyer, Raul Camposano, “Fast Hierarchical Multi-Level Fault Simulation of Sequential Circuits with Switch-Level Accuracy”, Design Automation Conf. (DAC), pp.515-519, 1993.

[7] P. Camurati, F. Corno, M. Meo, P. Prinetto, “A New Functional Fault Model for System-Level Descriptions”, Proc. IEEE VLSI Test Symposium, 1994.

[8] J.L Barreda, I. Hidalgo, V. Fernandez, P. Sánchez, E. Villar, “Fault Modeling in Vital”, Workshop on Libraries, Component Modeling, and Quality Assurance, IRESTE – IHT, Nantes, France, 1995.

[9] Weiwei Mao, Ravi K. Gulati, “Improving Gate Level Fault Coverage by RTL Fault Grading”, Proc. Int. Test Conf. (ITC), pp.150-159, 1996.

[10] M.S. Hsiao and J.H. Patel, “A new architecture-level fault simulation using propagation prediction of grouped fault-effects”, Proc. Int. Conf. On Computer Design (ICCD), pp. 628-635, 1995.

[11] L.C. Wang, R. Mercer, T.W. Williams, “On the Decline of Testing Efficiency as Fault Coverage Approaches 100%”, Proc. IEEE VLSI Test Symp. (VTS)., pp. 74-83, 1995.

[12] P.C. Maxwell and R.C. Aitken, “Biased Voting: A Method for Simulating CMOS Bridging Faults in the Presence of Variable Gate Logic Thresholds”, Proc. Int. Test Conf. (ITC), pp. 63 – 72, 1993.

[13] F.M. Gonçalves, J.P. Teixeira, “Sampling Techniques of Non-Equally Probable Faults in VLSI Systems”, Proc. IEEE VLSI Test Symp. (VTS), pp. 283-288 , 1998.

[14] R. Rodrígues-Montañés, E.M.J.G. Bruls, J. Figueras, “Bridging Defects Resistance Measurements in a CMOS Process”, Proc. Int. Test Conf. (ITC), pp. 892-899, 1992.

[15] J.M. Acken, S.D. Millman, “Accurate Modeling and Simulation of Bridging Faults”, Proc. Custom Integrated Circuits Conference, pp 17.4.1 – 17.4.4, 1991.

[16] B. Chess and T. Larrabee, “Bridge Fault Simulation Strategies for CMOS Integrated Circuits”, Proc. Design Autom. Conf. (DAC), pp. 458 – 462, 1993.

[17] U. Mahlstedt, J. Alt, “Simulation of non-classical Faults on the Gate Level – The Fault Simulator COMSIM“, Proc Int. Test Conf. (ITC), pp. 883 – 892, 1993.

[18] B. Chess, A. Freitas, F. Joel Ferguson, T. Larrabee, “Testing CMOS Logic Gates for Realistic Shorts”, Proc. Int. Test Conf. (ITC), pp. 395 – 402, 1994.

[19] J.P. Shen, W. Maly and F.J. Ferguson, “Inductive Fault Analysis of MOS Integrated Circuits”, IEEE Design and Test of Computers, n.2, pp. 13 – 26, December 1985.

[20] F.J. Ferguson and J.P. Shen, “A CMOS Fault Extractor for Inductive Fault Analysis”, IEEE Trans. on Computer-Aided Design, pp.1181-1194, Nov., 1988.

[21] W. Maly, M.E. Thomas, J.D. Chinn and D.M. Campbell, “Double-Bridge Test Structure for the Evaluation of Type, Size and Density of Spot Defects”, Tech. Report CMUCAD-87-2, Carnegie Mellon University, Feb., 1987.

[22] H. Konuk, F. J. Ferguson, T. Larrabee, “Charge-Based Fault Simulation for CMOS Network Breaks”, IEEE Trans. on Computer Aided Design”, pp. 1555 – 1567, Dec., 1996.

[23] M. Saraiva, M.B. Santos, A.P. Casimiro, I.M. Teixeira, J.P. Teixeira, “On the Design of a Highly Testable Cell Library”, Microprocessing and Microprogramming, vol. 35, pp.383-389, 1992.

[24] F. Brglez, H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortran", Proc. Int. Symp. on Circuits and Systems (ISCAS), pp. 662-698, 1985.

[25] F. Brglez, D. Bryan, K. Kominski, "Combinational Profiles of Sequential Benchmark Circuits", Proc. Int. Symp. on Circuits and Systems (ISCAS), pp. 1229-34, 1989.