EDA Solutions to New-Defect Detection in Advanced Process Technologies

Erik Jan Marinissen[1] Gilbert Vandling[2] Sandeep Kumar Goel[3] Friedrich Hapke[4]
Jason Rivers [5] Nikolaus Mittermaier[6] Swapnil Bahl[7]

[1] IMEC, Kapeldreef 75, Leuven, Belgium, erik.jan.marinissen@imec.be
[2] Cadence, 1701 North Street, Endicott, NY, USA, vandligc@cadence.com
[3] TSMC, 2585 Junction Ave., San Jose, CA, USA, skgoel@tsmc.com
[4] Mentor Graphics, Tempowerkring 1B, Hamburg, Germany, friedrich_hapke@mentor.com
[5] AMD, 5900 E. Ben White Blvd., Austin, TX, USA, jason.rivers@amd.com
[6] Synopsys, Hammerschmidt-Str. 34, Dornach, Germany, nikolaus.mittermair@synopsys.com
[7] ST Microelectronics, Via C. Olivetti 2, Agrate Brianza, Italy, swapnil.bahl@st.com

Abstract

For decades, EDA test generation tools for digital logic have relied on the Stuck-At fault model, despite the fact that process technologies moved forward from TTL (for which the Stuck-At fault model was originally developed) to nanometer-scale CMOS. Under pressure from their customers, especially in quality-sensitive application domains such as automotive, in recent years EDA tools have made great progress in improving their detection capabilities for new defects in advanced process technologies. For this Hot-Topic Session, we invited the three major EDA vendors to present their recent greatest innovations in hiqh-quality automatic test pattern generation, as well as their lead customers to testify of actual production results.

1 A Case Study in Faster-Than-At-Speed Testing

Gilbert Vandling – Cadence Design Systems; Sandeep Kumar Goel – TSMC

1.1 Introduction

Advances in semiconductor process technology are continuing to fuel the design of very complex and large integrated circuits. Integrated circuits manufactured in advanced processes can suffer from both manufacturing and process-related defects such as resistive opens/bridges, process variation, power supply noise, and crosstalk. Most of these defects introduce small additional delays in the circuit [1, 2] and are commonly referred to as small-delay defects (SDDs). Testing of small-delay defects has become increasingly important in today's smaller technologies. SDDs can cause immediate failure of a circuit if introduced on critical paths, whereas they cause major quality concerns if they occur on non-critical paths [3]. For very high-quality products (0-100 Defective Parts Per Million, DPPM), testing of SDDs is a must [4, 5]. We present a case study in SDD testing in a very advanced process node. A faster-than-at-speed methodology (FTAS) was used to produce SDD test patterns. The maxim passing speeds of these tests were compared to the speeds predicted by an on-chip ring oscillator. The effectiveness of the SDD tests was also compared with that of traditional transition delay fault (TDF) tests.

1.2 What is FTAS Testing

To illustrate faster-than-at-speed testing, consider the example circuitry shown in Figure 1.1. The numbers shown in different logic gates represent the lumped propagation delay through the gate. For simplicity we ignore interconnect delays.

Let us consider that the second fan-out branch of Node A has a delay defect of size 2ns as shown in Figure 1.1. The longest path for this fault is p1 with a length of 4ns. The cycle time for the circuit is 7ns. Since the timing-aware test is applied at cycle time (7ns), this defect is not detected since the defective path length is 6ns. This is also shown in Figure 1.1(a). However, when the same test is applied at faster-than-at-speed (6ns), then the defect is detected through flip flop FF2 (shown in Figure 1.1(b)). This example clearly indicates that for defects on the short paths, faster-than-at-speed is the only effective test method.

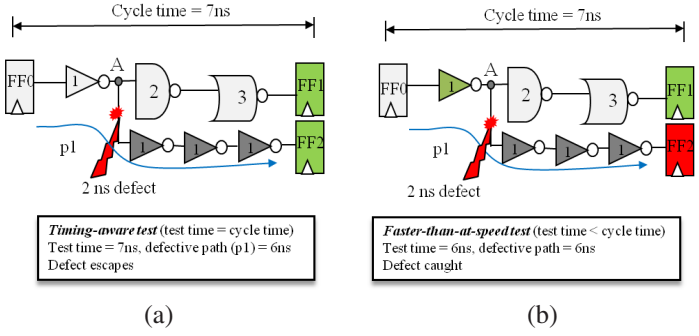

Figure 1.1: Example circuit for SDD.

978-3-9810801-8-6/DATE12/©2012 EDAA

Figure 1.2 shows the slack distribution for the paths in a hypothetical design. As mentioned in [6], this distribution is not atypical. Depending on the path slack profile, one would need to go X% faster-than-at-speed to cover possible SDDs on a pre-defined number of paths. For designs with tightly grouped path lengths (similar to Figure 1.2), faster-than-at-speed testing can readily be used to aggressively test a large number of paths.

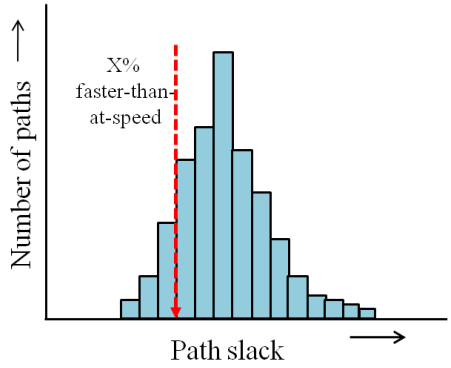

Figure 1.2: Typical slack distribution for paths in a design.

1.3 Pattern Application Results

SDD test patterns were created for a large, industrial design in a very advanced process node. As the design's slack distribution was similar to that shown in Figure 1.2, a single operating point (24% faster than nominal) was used. For comparison, a set of traditional TDF test patterns was also produced.

The SDD tests were characterized on a large number of devices. The performance of each device was predicted using measurements of the design's ring oscillator. This was com-pared to the maximum passing speed of the SDD test patterns.

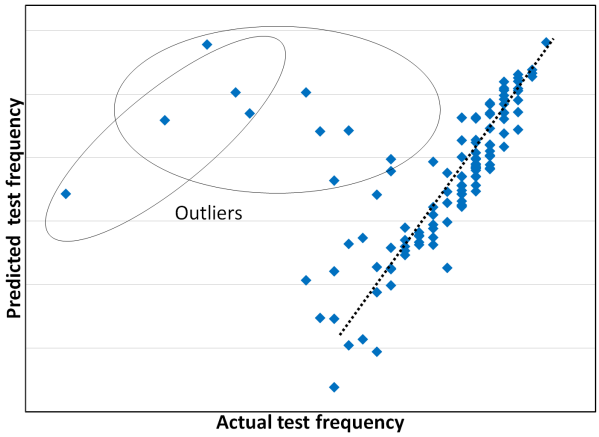

Figure 1.3: Pattern characterization results for the selected design.

Figure 1.3 shows a scatter plot of the predicted versus actual device speeds for the design tested. It can be seen that there is a good correlation between predicted and measured speeds for most of the devices tested. Few, if any, devices tested faster than was predicted, while a handful of devices tested markedly slower than predicted (labeled as outliers in the plot). A number of devices tested marginally slower than predicted, suggesting that the delay defect sizes were biased towards smaller delays.

Figure 1.4 shows the comparison between the SDD and TDF test patterns. Every device that failed the TDF patterns also failed the SDD patterns. It can be seen that the SDD patterns detected about four times as many (believed) defective devices as the TDF patterns. It should be noted that part of the higher SDD fallout may be attributable to the pass/fail condition having been correlated to the ring oscillator speed rather than testing at a fixed frequency, as was the case with the TDF test patterns.

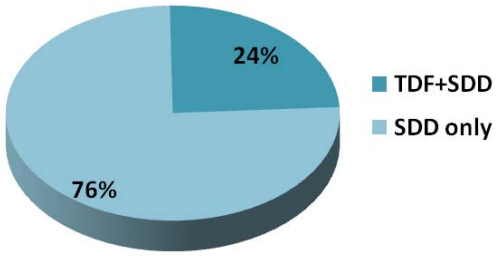

Figure 1.4: Pass/Fail comparison between SDD and TDF.

Diagnostics were run on the outlier devices and, in all cases, the callouts were unique. Analysis of both the tester fail data and the diagnostics results showed that the fails were consistent with random delay defects. Two of the outlier devices were selected for further characterization with path delay tests (both through and around the suspected defect). The path delay test results were consistent with the diagnostics callouts and the devices were submitted to physical failure analysis (PFA). In both cases, PFA was successful in confirming the defects.

1.4 Conclusion

A faster-than-at-speed test methodology was used on a large, industrial design in a very advanced process node. The tester results showed a number of devices with small-delay defects. Comparison with the TDF patterns showed that the SDD tests detected four times as many defective devices as the TDF patterns. Two of the defective devices were further characterized with path delay tests and the defects confirmed through PFA. The presented study validates the existence of SDDs and establishes the need for SDD testing in today's smaller technologies.

1.5 Acknowledgements

We thank Frank Lee, Charles Liu, Wei-Pin Changchien, Nan-Hsin Tseng, Saurabh Gupta, and Kuo-Yin Chen (all with TSMC) for their support during this work.

2 Cell-Aware Library Characterization for Advanced Technology Nodes and Production Test Results from a 32nm Processor

Friedrich Hapke – Mentor Graphics; Jason Rivers – AMD

This paper describes a new approach for significantly improving overall defect coverage for CMOS-based designs. Here we will present results from a defect-oriented cell-aware library characterization and pattern generation flow, developed by Mentor Graphics, and its application for 1,900 cells of a 32nm technology at AMD. The cell-aware flow enabled AMD to detect cell-internal bridges and opens that caused static, gross-delay and small-delay defects. Finally, we will present high-volume production test results from a 32nm notebook processor where cell-aware test patterns were applied, including the defect rate reduction in PPM that was achieved.

2.1 Introduction

Physical defects like shorts and opens may occur during any step of the fabrication process. Well-known fault models like stuck-at (SA) [7], transition (TR) [8], N-detect (ND) [9], gate-exhaustive (GE) [10], embedded-multi-detect (EMD) [11], and also layout-aware fault models on inter-connect lines have been presented and used in production for a long time. These fault models are insufficient for today's technologies that require low defect rates. In our previous work, where we introduced the cell-aware (CA) methodology [12, 13], we showed that the classical SA, TR, ND, and GE approaches do not target all real defects within library cells or are too expensive for production tests. First production test results we presented in [14] for slow-speed cell-aware, gate-exhaustive, and transition N-detect tests. In this paper we now focus on a full application of the cell-aware methodology which includes at-speed cell-aware tests for bridges and opens for a 32nm technology. Also included are production test results from 800,000 tested ICs of a 32nm AMD notebook processor.

2.2 Cell-aware Library Characterization

The library characterization is performed once per technology node to create the needed cell-aware view for each cell of the standard library. As illustrated in Figure 2.1, the flow starts with the layout extraction from a given layout (gds2) of a standard cell. This is then followed by an analog fault simulation step and a cell-aware synthesis step which creates the cell-aware test view of a cell. This cell-aware view is passed on to the cell-aware ATPG which generates the high-quality cell-aware test patterns.

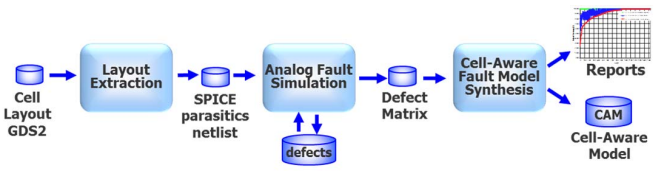

Figure 2.1: Cell-aware characterization flow.

The Defect Coverage Graph

This section presents cell-aware defect coverage results obtained from analyzing 1,900 cells from a 32nm technology. The graph in Figure 2.2 shows the deficiency of the state-of-the-art stuck-at and transition pattern.

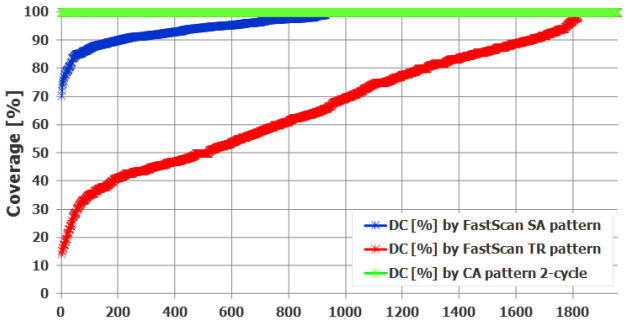

Figure 2.2: Defect coverage graph.

The horizontal axis represents the library cells numbered from 0 to 1,900. The vertical axis represents three defect coverage rates in percent as follows: The blue line is the defect coverage rate for bridges and opens that is achieved with state-of-the-art stuck-at patterns. The red line is the defect coverage rate for bridges and opens that is achieved with state-of-the-art transition patterns. And the green line is the defect coverage rate (which is always at 100%) for cell-aware slow-speed and at-speed patterns. As can be seen, for about 50% of the cells the defect coverage of stuck-at patterns is less than 100% and some cells just reach a defect coverage of 70%. The situation is worse for transition patterns, where roughly 10% of the cells reach a defect coverage of 100%. In fact, many cells (about 600) have a defect coverage of less than 50% while some cells reach just 20%.

Coverage Gain per Cell Type

For this analysis, we performed the cell-aware library characterization as described above We then performed various fault simulation and ATPG runs with those newly generated cell-aware library views and the selected processor design. We began by fault-simulating the existing stuck-at and transition patterns with the new cell-aware fault simulator to obtain the defect coverage of SA and TR patterns. Next, we generated a new cell-aware slow-speed and at-speed test pattern set to obtain the defect coverage increase compared to SA and TR patterns.

An overview of the defect coverage gain by cell type, that is achieved with the at-speed cell-aware patterns is given in Figure 2.3. The red line shows the coverage gain, and the blue line shows the number of cell-aware detected defects related to each cell type. As can be seen, the multiplexer with three data inputs (imux3) achieves the biggest coverage gain (about 2.4%). The

Nand gate with four inputs (nd4) achieves a coverage gain of 1.8% and the and-or (aoi) cell achieves a coverage gain of 0.6%. On average, a coverage gain of 0.8% is achieved.

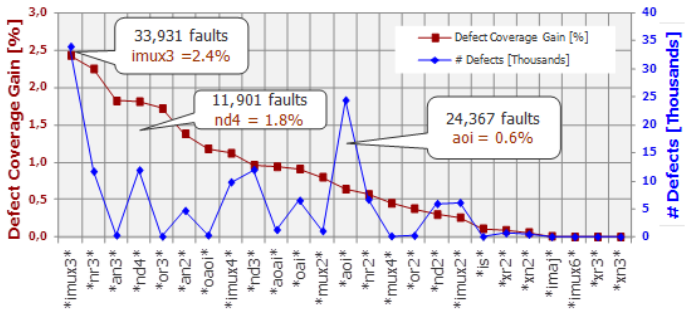

Figure 2.3: At-speed cell-aware defect coverage gain.

Cell-aware Detections in imux3 Cells

Deeper investigation into the multiplexer cell (imux3) shows the potential defects that are typically not detected with the traditional SA and TR tests.

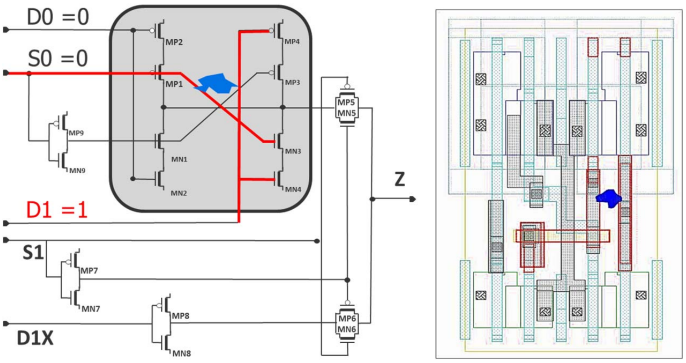

Figure 2.4: Cell-aware detections in imux3.

An example of those elusive defects is shown in Figure 2.4. This is a bridge between the select input S0 and the data input D0. Normal SA and TR ATPG tools are not forced to assign a value to a data input (e.g. D1) when the D0 is currently selected by S0.

2.3 Production Test of a 32nm Design

For evaluating the effectiveness of the cell-aware slow-speed and at-speed patterns in relation to the normal production test, we have chosen a 32nm notebook processor design.

Experimental Test Flow

To investigate the effectiveness of the cell-aware patterns in relation to SA and TR patterns, we added the experimental patterns to the test program and changed the test flow to log unique fails of the CA patterns as shown in Figure 2.5.

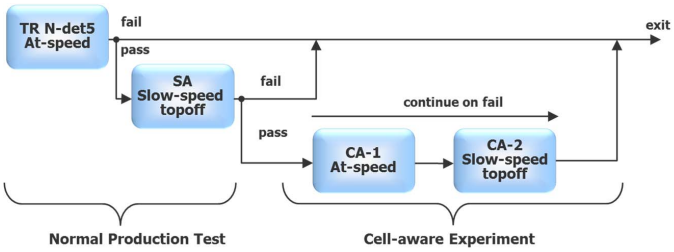

Figure 2.5: Production test flow.

The production test consists of an at-speed TR- N-Detect5 test and a slow-speed SA topoff test. All experimental patterns were in data-collection mode, otherwise known as “continue on fail”. We applied the experimental tests to all die where all four cores passed the existing production test.

Production Results

After having tested 800,000 ICs, we summarize the fail and PPM reduction results in Figure 2.6.

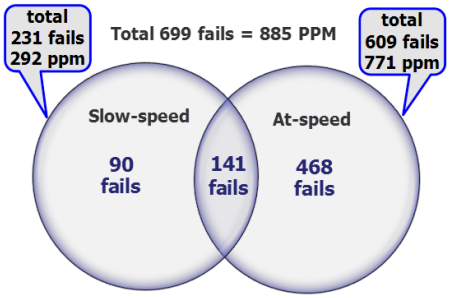

Figure 2.6: PPM reduction.

This data shows that the slow-speed cell-aware patterns detect a total of 231 defects that state-of-the-art transition and stuck-at patterns do not detect. The at-speed cell-aware patterns detect a total of 609 defects. These fail counts can be easily transformed into PPM rates, i.e., the slow-speed CA patterns are reducing the defect rate by 292 PPM and the at-speed CA patterns by 771 PPM. The Venn diagram also shows the overlap between the two CA tests. In total the CA tests are reducing the PPM rates for this 32nm design by 885 PPM.

Of the 699 parts the cell-aware patterns detected, 446 were packaged and tested through the production ATE and system level flows for yield analysis. 48% of the units failed a core test at package ATE and reconfigured as two-core units. Of the remaining four-core units tested at system level, 46% ended up passing at a slower than top frequency. The package data validates the unique cell-aware fails do result in first-time right yield loss.

2.4 Conclusion and Future Work

We presented library characterization from 1,900 cells from a 32nm technology and cell-aware production test results from a 32nm AMD processor. Based on the analysis, we can state that

the CA method is detecting various otherwise undetected defects and as such the overall defect rate can be reduced significantly. For the selected 32nm process the defect rate could be reduced in total by 885 PPM and improved first-time right package yield.

We will apply the CA method on more designs and other technologies. We will perform Layout-aware diagnosis runs on the failing CA parts and we will perform physical diagnosis for selected failing ICs.

2.5 Acknowledgements

The authors thank Michael Reese, Andreas Glowatz, Andrew Over, Wilfried Redemund, Grady Giles, Jürgen Schlöffel, Jeff Rearick, Janusz Rajski, Daniela Toneva, and Anja Fast for their assistance, valuable discussion, implementations, and insight over the course of this project.

---

3 Using Slack-Based Transition Patterns for Ultra-High Defect Coverage

Nikolaus Mittermair – Synopsys; Swapnil Bahl – ST Microelectronics

With shrinking geometries, test engineers are challenged to target new defect types. Most of those defect types are related to additional net- and cell-delays introduced during manufacturing. In order to target those, new fault models have been introduced by semiconductor and EDA companies. One of these models is called Transition Delay, which enables at-speed testing of integrated circuits. Here two patterns are generated which do first launch a transition along a timing path and second capture the transition into the timing end point, which could either be a scan flop or a primary output. The transition delay fault mode has one drawback. It is not aware of any circuit timing, so a transition could propagate along any timing path, even if the slack (period minus actual path delay) is still very big. This would cause small delay defect to pass testing. In order to overcome this issue ST Microelectronics as well as others do now use Slack-Based Transition Delay ATPG, were ATPG is constrained to propagate the transition along the path with minimum slack.

3.1 Slack-Based Transition Delay ATPG

Slack-Based transition delay (TD) ATPG testing is feasible only if the ATPG tool can make efficient decisions – often thousands per pattern – based on accurate timing information of the design. Yet the need to dynamically perform timing calculations inside the ATPG tool (based on Standard Delay Format data, for example) compromises run time performance and can produce results that do not correlate well with sign-off timing analysis. TetraMAX ATPG avoids these issues by accessing data generated by Synopsys' PrimeTime static timing analysis, the industry's defacto signoff solution widely used by design groups at ST Microelectronics.

Two basic kinds of timing information are required: slack data and timing exception data. The combined flow for generating this information is depicted in Figure 3.1.

1. A Standard Test Interface Language (STIL) procedure file (SPF) describes the test mode protocols for launch and capture.

2. The TetraMAX ATPG design rule checker (DRC) interprets the launch and capture timing requirements and produces an SDC file defining the launch and capture mode timing constraints.

3a. PrimeTime interprets the launch and capture mode timing constraints and produces an SDC file defining the test mode timing exceptions.

3b. PrimeTime generates a report containing the test mode timing slacks for each fault site.

4. TetraMAX ATPG reads in the slack and timing exception information produced by PrimeTime to generate slack-based transition delay patterns.

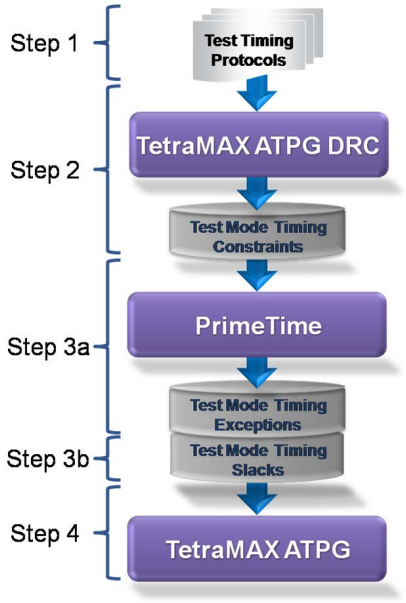

Figure 3.1: Production test flow.

3.2 Silicon Testing Results

The Synopsys flow for slack-based transition ATPG described in the previous section is supported in the signoff methodology at ST Microelectronics, where it is considered key to reducing DPPM (defective parts per million) levels across the company's product lines. To convey the quality advantage of slack-based testing over standard transition delay testing (or any other type of test in use today), we now present failure statistics related to an automotive IC designed and manufactured by ST Microelectronics. The design, consisting of approximately one million equivalent gates, was manufactured in a 90-nm CMOS process.

Figure 3.2 displays a Venn diagram of data collected from testing hundreds of thousands of parts using four fundamental types of tests: stuck-at, static bridging, dynamic bridging, and delay tests. The delay tests were composed of standard TD patterns and slack-based TD patterns. The data indicate that about 94.5% of all failing parts were covered by the delay tests, and that 20% of failing parts were covered only by the delay tests.

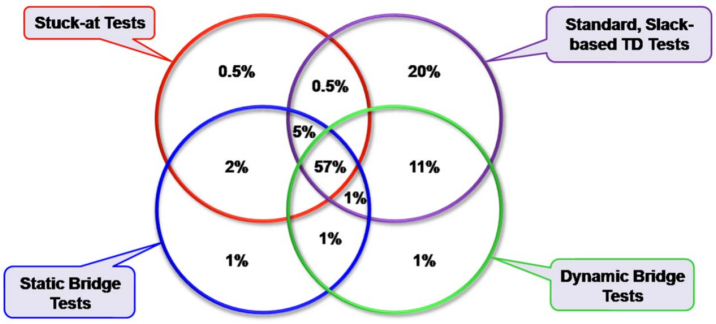

Figure 3.2: Venn diagram showing the percentage of failing parts covered by each type of test.

Upon examining the data for the 20% of defective parts covered only by the delay tests, we observed that 63% of these parts were covered only by the slack-based TD patterns. The remaining 37% covered by standard TD patterns were also covered by slack-based TD patterns. Test engineers at ST Microelectronics have observed results similar to these for other designs; slack-based TD tests consistently screen more failures than any other type of production test in use at ST Microelectronics, reducing DPPM levels relative to rates achievable using standard TD tests.

3.3 Small Delay Predictions

Recent advances in design automation technologies have made it possible for ST Microelectronics and other semiconductor companies to efficiently target small delay defects during manufacturing test. This development has ensured that, in spite of Moore's Law, very high defect coverage will be achievable on a consistent basis in the years ahead: designers will increasingly adopt SDD testing as the primary means to maintain low DPPM levels as they take advantage of ever-smaller geometries to squeeze more functionality on a chip. The growing number of worldwide deployments of TetraMAX ATPG to target small delay defects has stemmed from the fact that the Synopsys methodology requires no significant change to existing delay test flows and leverages the PrimeTime analysis engine, used widely by engineers for timing signoff, to efficiently generate the patterns. For these reasons, we believe that it will only be a matter of time before small delay defect ATPG replaces transition delay ATPG as the preferred approach to testing digital ICs.

References

[1] S. Mitra et al. Delay Defect Screening using Process Monitor Structures. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 43–48, April 2004.

[2] B. Kruseman et al. On Hazard-Free Patterns for Fine-Delay Fault Testing. In *Proceedings IEEE International Test Conference (ITC)*, pages 213–222, October 2004.

[3] H. Villacorta et al. Reliability Analysis of Small Delay Defects in Vias Located in Signal Paths. pages 1–6, November 2010.

[4] R. Turakhia et al. Silicon Evaluation of Longest Path Avoidance Testing for Small Delay Defects. In *Proceedings IEEE International Test Conference (ITC)*, pages 1–10, October 2007.

[5] C-J.J. Chang and T. Kobayashi. Test Quality Improvement with Timing-Aware ATPG: Screening Small Delay Defect Case Study. In *Proceedings IEEE International Test Conference (ITC)*, page 1, October 2008.

[6] S.K. Goel, N. Devta-Prasanna, and R.P. Turakhia. Effective and Efficient Test Pattern Generation for Small Delay Defects. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 111–116, May 2009.

[7] K.C.Y. Mei. Bridging and Stuck-At Faults. *IEEE Transactions on Computers*, C-23(7):720–727, July 1974.

[8] H. Cox and J. Rajski. Stuck-Open and Transition Fault Testing in CMOS Complex Gates. In *Proceedings IEEE International Test Conference (ITC)*, pages 688–694, 1988.

[9] I. Pomeranz and S.M. Reddy. On N-Detection Test Sets and Variable N-Detection Test Sets for Transition Faults. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 173–180, April 1999.

[10] K.Y. Cho, S. Mitra, and E.J. McCluskey. Gate Exhaustive Testing. In *Proceedings IEEE International Test Conference (ITC)*, pages 1–7, 2005.

[11] J. Geuzebroek et al. Embedded Multi-Detect ATPG and Its Effect on the Detection of Unmodeled Defects. In *Proceedings IEEE International Test Conference (ITC)*, pages 1–10, 2007.

[12] F. Hapke et al. Defect-Oriented Cell-Aware ATPG and Fault Simulation for Industrial Cell Libraries and Designs. In *Proceedings IEEE International Test Conference (ITC)*, pages 1–10, November 2009.

[13] F. Hapke et al. Defect-Oriented Cell-Internal Testing. In *Proceedings IEEE International Test Conference (ITC)*, pages 1–10, November 2010.

[14] F. Hapke et al. Cell-Aware Analysis for Small-Delay Effects and Production Test Results from Different Fault Models. In *Proceedings IEEE International Test Conference (ITC)*, pages 1–10, September 2011.