Integrating Logical and Physical Analysis Capabilities for Diagnostics

Thomas Bartenstein, Bernd Koenemann, Lee Todd, and Lisa Vallerie
Cadence Design Systems, Inc.
San Jose, CA, USA
Contact: berndk@cadence.com

Summary

Nanometer technologies are subject to complex interactions between design and process that can contribute to design-specific parametric failures or yield detractors. Due to their design-specific nature, such problems can only be debugged and analyzed on the actual product chips. Scan-based tests can be diagnosed and debugged with the help of automated diagnostic software tools. These tools traditionally operate in the logic domain and require access to design data including logic netlists and the original test vectors. Further isolation of the defects in the physical domain requires access to circuit-level and physical design information. A number of companies, mostly Integrated Device Manufacturers (IDMs), have built in-house systems that integrate the logical, electrical, and physical domains. The integration effort often involves many different file formats, software tools, and scripts.

This paper describes and shows some anticipated application examples for an integrated logical/physical analysis environment that utilizes the public-domain OpenAccess database as a key infrastructure element. OpenAccess is an emerging vehicle that can be used to consolidate multiple different files into a single database with a rich API of access routines for querying the data. This consolidation into a database avoids the complexities of finding/managing multiple files and parsing/translating the different files to connect the information content. Using an EDA database makes it possible to enrich the failure analysis and debug environment with useful EDA-type utilities and applications.

Introduction

The introduction of automated logic diagnostic systems is a growing trend in the semiconductor industry, primarily in the form of in-house efforts by Integrated Device Manufacturers (IDMs). The objectives are to accelerate first silicon debug and failure analysis to speed time to market and improve yield ramp (see [6] for a discussion for the economics of failure analysis and yield ramp). One driver behind the trend is the expectation that unpredictable design-process interactions in nanometer technologies lead to design-specific test failures. While process-driven yield limiters can be successfully anticipated and weeded out by test chips, yield monitors, and in-line inspection [1], that approach does not carry over easily to design-driven yield issues. Pre-analysis and careful consideration potential yield issues with the help of Design-For-Manufacturability (DFM) software can have a positive impact on yield [2]. However, some design-driven yield issues may manifest themselves only in the real product and, hence, can only be debugged with the real product. In some circumstances, as described in [3], the product chips may be the only viable vehicle for learning.

Many design-driven causes for electrical test failures, like crosstalk or IR-drop for example, are not visible to in-line inspection tools. Neither can test chips cover all possible design configurations to adequately predict whether problematic conditions will exist on a particular product design or not. The resulting test fails can only be analyzed by mining the information hidden in the electrical test fail data themselves. Logic fault isolation software tools, which are commercially available from several Electronic Design Automation (EDA) tools providers, help automate the analysis of scan-based logic test fails. The tools read the fail sets from scan-based logic tests and, using the netlist and original test patterns, try to determine which logic nets are most likely associated with the root cause responsible for the observed fail set. The best diagnostic resolution that can be expected from tools that operate completely in the logic domain is a single, gate-level logic net and its logically equivalent nets. Improving the diagnostic resolution beyond this fundamental resolution limit requires consideration of additional, non-logic information in the analysis.

Physical (layout) information is an obvious candidate for facilitating further defect localization. If the gate level logic netlist used for fault isolation can be linked with the layout data then the logic net name(s) called out by the diagnostic software can be associated with layout shapes in the physical domain. To further improve the analytical capabilities of debug, diagnostics and silicon characterization, additional pieces of information can be useful. Logic Bitmapping, for example, attempts to accelerate root-cause identification by automatically correlating the physical shapes linked to callouts from logic fault isolation with defect maps generated by in-line inspection systems [5, 11]. This Logic Bitmapping

methodology, in spirit similar to bitmapping for memories [7], depends on the availability of electrical fail data and defect maps captured by the surface scan equipment from the same subset of wafers.

Other interesting applications of logic diagnostics focus on finding statistically relevant design weaknesses or design-process interactions that affect yield and/or performance [11]. Properly mined for information, the electrical fail data can unearth valuable feedback to design and process engineers. Pre-sign-off design closure depends on the validity and appropriateness of models and algorithms used to predict silicon behavior. There are serious concerns, fueled by the unexpected difficulties in ramping yield for 130nm technologies, that inaccuracies in models and algorithms for nanometer technology nodes will make the success of design closure increasingly unpredictable. Trying to avoid the impact of inaccuracies by conservatively guard-banding the models and algorithms can leave significant performance, area, and power advantages on the table. Without conservative guard-banding, on the other hand, inaccurate predictions can lead to design-specific yield loss. Getting to the bottom of such yield loss can be facilitated by performing logic fault isolation from large quantities of electrical fail data and then statistically analyzing the resulting logic callouts at the physical and electrical level.

Today's typical design flows and sign-off criteria fail to preserve the predictive design data in a controlled and easily accessible data repository that can be used by design, product, and yield engineers alike. If the data are preserved at all, they reside in a multiplicity of different files, possible at different locations. That also holds true, for instance, for the linkage between the logic netlist and the layout data (GDSII stream) that is used for mask preparation. The linkage must then be laboriously reconstructed by running LVS (Layout Versus Schematic) for diagnostics. For large designs that can take many hours, provided the correct files can be found in the first place. On the other hand, the linkage information is created and used by the layout tools (e.g., in LEF/DEF) prior to sign-off, and re-running LVS could be avoided if the layout-generated information could be captured into a suitable data repository that can be accessed not only from pre-signoff design tools but also from the post-silicon environment.

The following discussion will introduce the industry-standard OpenAccess data base and illustrate how this database can be used for better integration of a more comprehensive post-silicon debug, diagnostics, and characterization environment.

OpenAccess Summary

OpenAccess is the name for a unified database that originally was created to facilitate the integration of EDA (Electronic Design Automation) tools. To be effective as a vehicle for integrating tools from different vendors, it was decided to form the so called OpenAccess coalition and to make the database publicly available [for OpenAccess information and downloads, see http://www.si2.org].

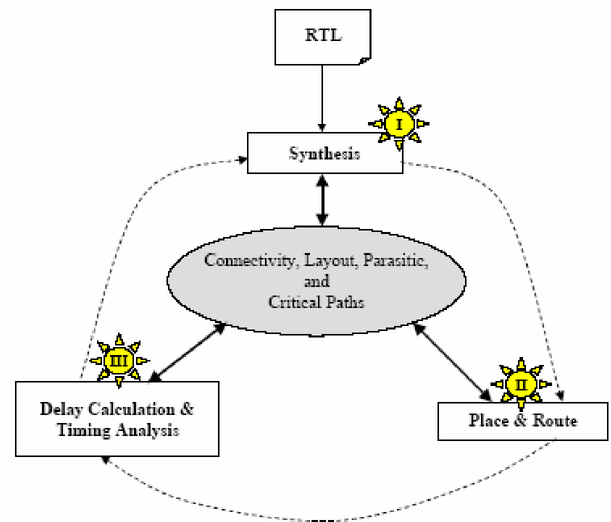

Figure 1: High Level View of EDA Tools with OpenAccess Database content elements in the Middle

Figure 1 [9] illustrates, at a high level, an integrated EDA flow. OpenAccess is intended to be a highly flexible, high-performance, and highly controllable data repository that can fill the shaded area in the center of the picture. The basic data structures in the OpenAccess database support a unified view of the structural connectivity and physical design information ranging from structural RTL (Register Transfer Level) to layout (GDS II stream equivalent) data. In addition to these design data per se, OpenAccess also can contain other types of data like design rules that express process technology information or parameter points for analysis tools.

OpenAccess has not been only been tuned for data content but also for performance such that it is possible to implement many EDA-type applications to run natively on the database. In addition to APIs (Application Programming Interfaces) for native access, there are APIs that facilitate interfacing to other existing data models. For example, GDS II stream, LEF and DEF can be written into OpenAccess very efficiently.

The OpenAccess architecture is designed for extendibility beyond being merely a traditional EDA database. For example, a UDM (Unified Data Model) workgroup has been formed to codify standardized database extensions that better link design and manufacturing [for information see http://www.si2.org/oa-udm/]. Figure 2 [10] shows a diagram that illustrates a vision of the scope of the UDM project.

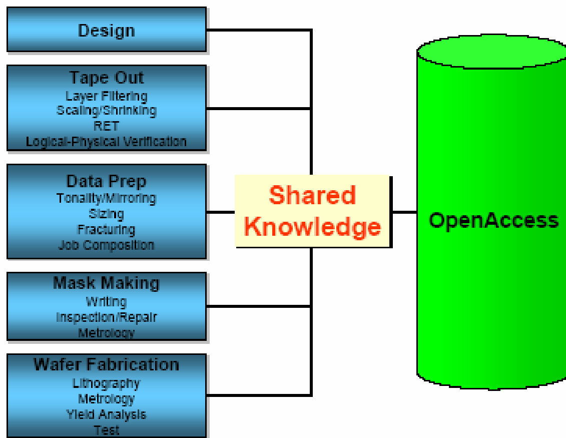

Figure 2: OpenAccess/UDM in Design and Manufacturing

What is particularly interesting about this vision for the context of post-silicon debug, diagnostics, and characterization is that OpenAccess/UDM is seen as a common data repository for both, pre-signoff and post-silicon use. The key idea is that relevant design-related information is captured into the OpenAccess/UDM data base and that database is made available for the transition to manufacturing (e.g., mask data prep, mask making) all the way to manufacturing (e.g., Lithography, Metrology) and post-silicon analysis tools.

While UDM at this stage may still be vision, there are already a number of OpenAccess releases and tool applications available today, that are suitable for a useful level of integration.

A Possible Role for OpenAccess/UDM

Modern database technology already is an increasingly recognized and important infrastructure feature for statistical yield management and advanced process control applications. Wafer processing, in-line inspection, metrology, and test equipment continuously generate vast amounts of data as a wafer proceeds through the fabricator, test, and assembly areas. Emerging commercial software vendors as well as in-house development teams use relational database technology for consolidating the data together with logistical Wafer In Progress (WIP) data in a data warehouse. The main value of the software packages is not in the database/data warehouse per se, but in intelligent data mining and analysis software that extracts useful information from literally terabytes of data snippets in the warehouse.

OpenAccess/UDM, at least currently, is not intended to supplant the very dynamic databases in the warehouse. Instead, OpenAccess/UDM should be viewed as complementary technology that captures the more static information about a product's design, design analysis, and design intent information, and that can help to make this information available in the manufacturing environment. Being an EDA database, OpenAccess/UDM is architected not only as a data repository, but to facilitate native support of EDA-type utilities and applications.

Figure 3 shows a high-level view of how OpenAccess/UDM could fit into an overall software platform for FA/Debug lab automation and yield management.

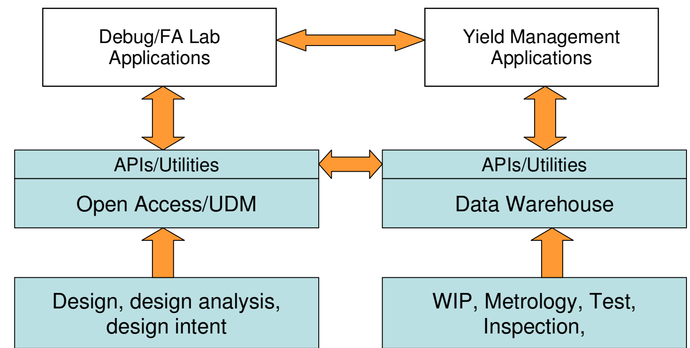

Figure 3: Possible Platform Architecture

This platform architecture envisions that the design and manufacturing databases are kept separate in recognition of their different purposes and internal structures. However, information can be interchanged between the two sides through the data access APIs/Utilities and at the applications level.

At the applications level, it is envisioned that FA/Debug lab automation would be more closely associated with the OpenAccess/UDM side, while statistical yield management is closer to the data warehouse side.

A Prototype of Logical/Physical Integration for Logic Diagnostics

An initial integrated logical/physical analysis environment for logic diagnostics has been established for prototyping purposes. It consists of a self-contained logical fault isolation application that is bi-directionally coupled to a physical navigation and shapes processing platform running natively on OpenAccess.

The logical fault isolation environment requires a gate-level netlist (e.g., in Verilog format), the original test patterns (e.g., in IEEE 1450 STIL format) from the ATPG (Automatic Test Pattern Generation) tool, and fail data logs (in a simple ASCII format) captured during test or in simulation. The tool analyses the fail information and attempts to derive the most likely logical net(s)/pin(s) associated with the cause of failure and information about the logical nature of the fail (e.g., does it appear to be a bridging-type fault or something else?). The logical environment contains a high-performance logic schematic navigation and logical analysis tool with simulation and ATPG-type features to annotate the schematic with

relevant logic values, as well as the ability to visualize and navigate the test patterns and/or associated waveforms in a pattern browsing and waveform display tool. Figure 4 shows a piece of a screenshot from the logic browser. It shows the logic gates and nets associated with a particular call-out from the logic fault isolation tool.

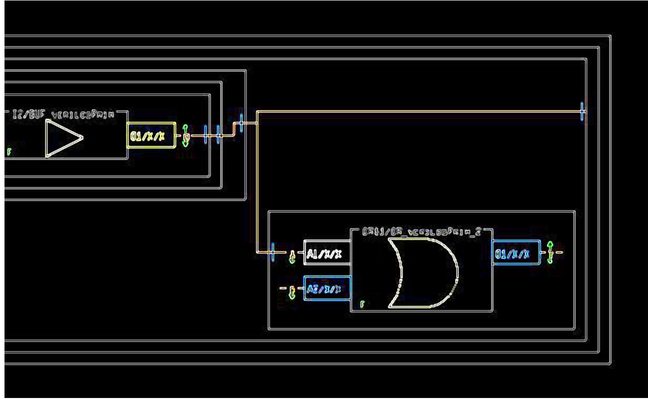

Figure 4: Partial Screenshot from the Logic Browser

A diagnostic ATPG engine is also included to optionally generate additional test vectors for better fault isolation or targeted characterization. This ATPG engine has a special mode for generating repeating tests that do not require full scan between repetitions and that, hence, are take less time than normal scan tests when using certain tools like Voltage Contrast E-Beam, Time Resolved Photon Emission, or Laser Voltage probers that employ repeated sampling of periodic waveforms to accumulate reliable signals.

The physical analysis environment is driven by layout/connectivity information that can be directly captured into OpenAccess during design or imported into OpenAccess from design files (e.g., LEF, DEF, GDS II stream).

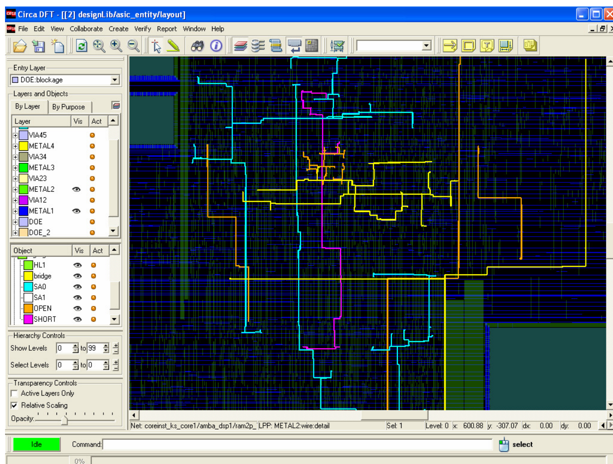

Figure 5: Screenshot from Physical Browser

A screenshot from the physical browser with several highlighted nets is shown in Figure 5.

In addition to the layout data (shapes and connectivity) it is possible to import and/or create other layers with different purposes. Automatic import facilities, for example, exist for DRC (Design Rules Check) error markers from a variety of commercial DRC tools. Another type of information that could be loaded into the database as layers, for example, is in shapes as modified by OPC (Optical Proximity Correction). Besides offering physical browsing and navigation capabilities, built-in shapes processing and connectivity extraction utilities are included. Database access APIs and scripting allow a user to perform customized queries that, for example, can return information about the physical properties (e.g., via count, or overlap/proximity with/to DRC error markers) of layout entities relevant to diagnostics. Furthermore, special attention has been given to features that allow geographically dispersed teams to collaborate.

Figure 6 illustrates an example of an annotation added to the layout view. All such annotations are added to the OpenAccess database and to an annotations list allowing others to automatically navigate to these annotations.

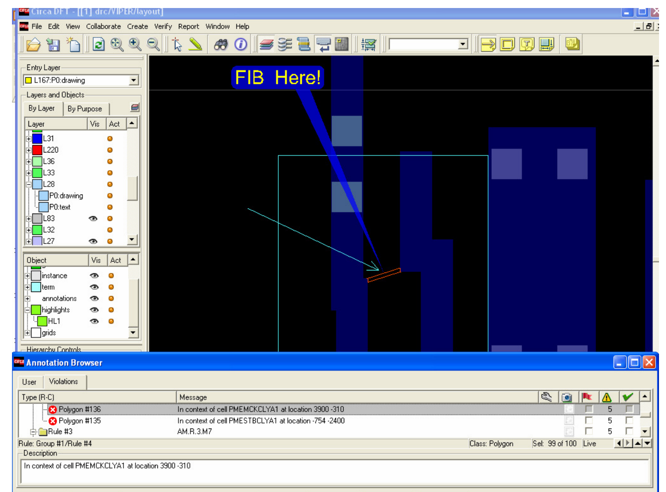

Figure 6: Annotation in Physical Browser

The logical and physical environments can be started together with built-in bi-directional cross-probing capabilities between logical and physical.

Combined Logical/Physical Analysis Examples

Signal Interaction Analysis

Unwanted interaction between logic signals can result in test fails. The interaction can be due to a bridging-type manufacturing defect or due to signal integrity issues like crosstalk. The logic analysis (fault isolation step) is primarily performed using stuck-at and/or transition fault models. In the

signal interaction scenario, the victim signal will in some pattern or patterns fail with an incorrect logic value, but the combined failure syndromes are not fully consistent with a permanent stuck-at or transition fault assumption. A special bridging-fault post-analysis of the raw stuck-at/transition analysis results is available to better classify the fail types without having to use explicit fault models for the interaction. For example, in a dominating net bridge, the victim signal is dominated by some aggressor signal and can fail as stuck at '0' (aggressor='0', victim='1') in some pattern(s) and like a stuck-at '1' (aggressor='1', victim='0') in other(s). For post-analysis a so called stuck-at 'X' assumption is available that combines stuck-at '0' and stuck-at '1' into a single fault assumption. If this combined model leads to a better callout, then a dominating net bridge is a likely scenario for the observed fail syndrome. This reasoning is not perfect, though, because in yet other pattern(s) the victim's logic state may be sensitized to a measure point such that a stuck-at fault would manifest itself. However, the aggressor happens to have the same expected value as the victim and the fault behavior is not excited. Neither a single stuck-at '0' nor a single stuck-at '1' can fully explain such fail behavior.

The next step in the analysis is trying to find possible candidates for aggressor signals. One method available for this purpose is called Invariant Analysis, which looks for nets that have the appropriate dominating value in the failing patterns (e.g., a logic '0' when the victim fails like stuck-at '0' and a logic '1' when the victim fails like stuck-at '1'). Without additional information, the logic analysis tool has no understanding of which signals are physically plausible candidates for interactions and, hence, has to search through all logic signals in the design. For large designs that list can be very large. And, it is quite conceivable that multiple nets have suitable values even if they cannot realistically be involved in the interaction.

Having an integrated physical analysis environment helps to mitigate this problem. The logic analysis tool can send the victim's net name to the physical analysis tools and query the physical design for nets and net segments that match certain proximity rules relative to the victim net. For example, the query could ask for all nets that have neighboring segments within a specified distance from some segment of the victim net at the same layer. The restricted list of candidate aggressors can then be sent back to the logic analysis environment for a more focused Invariant Analysis. Please, note that querying the physical database as part of the analysis, depending on the rules used for querying the database, can refine the result from complete nets to victim/aggressor net segments and associated layers.

In addition to Invariant Analysis, the logic analysis environment offers a flexible fault modeling capabilities (so called Pattern Faults). The physically reduced list of victim/aggressor net pairs can be used to generate a list dominating net bridge faults that are fed back into the diagnostic fault simulation engine for enhanced fault isolation. Re-running diagnostic fault simulation with the bridging fault models may be able to better resolve patterns where the victim signal is sensitized but the aggressor is at the same value. As mentioned above, analysis based on stuck-at faults would incorrectly predict a fail for such patterns while the correct bridging fault model would not.

Crosstalk analysis, incidentally, would be very similar. In that scenario, the physical analysis environment can be queried for neighboring nets that run in parallel on the same layer for a certain length. From this list, Pattern Faults can be created that require the candidate victim/aggressor net pairs to both have logic transitions in the same or opposing directions.

Probing and Circuit Edit

Modern front-side or back-side probing tools can be very helpful for improving diagnostic resolution [4] and can also be invaluable for characterization [8].The tools require access to certain design features that permit probing. For example, to acquire the time-resolved switching behavior of a logic signal, Photon Emission and/or Laser Voltage tools for back-side probing need to be positioned on the transistors capable of driving the logic net of interest. Voltage Contrast E-Beam machines, by contrast can acquire signals from unobstructed metal features from the front-side. The rules for positioning and navigating the probes, hence, are very different.

Having an EDA database like OpenAccess and EDA-type processing utilities available in the lab can help automate the step of finding and locating the features that need to be probed, be they transistors or unobstructed metal shapes. Finding transistors for probing is very much akin to EDA routines that extract circuits from layout. It entails traversing the shapes and layer stack in a way that identifies all transistors that "sit between" the net of interest and the power sources even if they are not directly connected to the net.

Finding front-side E-Beam or microprobe probe points on metal shapes requires finding locations that are close to the surface and have clear visibility/access from the front-side. For micro-probing, enough clearance is needed to allow for drilling a hole using Focused Ion Beam (FIB) technology without damaging other circuit elements.

Adding access to EDA-type simulation utilities to the software environment provides additional opportunities. For example, extracting circuits and parasitics of interest into a model for circuit simulation could be used to predict photon emission signatures for certain defect assumptions and comparing the simulated emission profiles with actual profiles acquired by Time Resolved Emission (TRE) probes [12]. Cruder predictions can be made from logic simulation [13].

In general, the usability of advanced probers for debug and failure analysis can benefit from having forward and backward

links between simulation tools and the probers. A designer using simulation and waveform display tools could request the acquisition certain waveforms from a prober and have the software automatically set up the prober. The acquired signals (e.g., photon count histograms from TRE equipment) would be back-translated into the waveform paradigm familiar to the designers [13].

It is envisioned that circuit editing by FIB could likewise be driven from the more familiar design tools environment and the modified circuit after FIB could be extracted back into the design environment for post-analysis.

To achieve sufficient flexibility it is important the software environment allows for customizing the database queries and navigation whenever new equipment or new technologies are brought in.

Future Work

The initial release of the logical/physical environment does not yet take full advantage of OpenAccess. Only the physical navigation and shapes processing utilities are running natively on OpenAccess. The logic schematic navigation and analysis tools still use an existing test tool database. For the future it would be desirable to have both, the logical and the physical navigation and analysis utilities running natively on OpenAccess.

Furthermore, the initial release comes with a limited number of built-in integrated query and analysis applications that only scratch the surface of what is possible. More experience is needed to find out what extensions should be put into the OpenAccess data model.

References

1. Skumanich and E. Ryabova, "Methodology for Targeted Defect Reduction and Inspection Optimization", IEEE/SEMI Advanced Semiconductor Manufacturing Conference, pp. 72-76, 2002.

2. Ouyang, et al., "Maximizing Wafer Productivity Through Layout Optimization", Thirteenth International Conference on VLSI Design, pp. 192 - 197, 2000.

3. J. Khare and H.T. Heineken, "Defect-Based Testing for Fabless Companies", IEEE International Workshop on Defect Based Testing, pp. 23-29, 2000

4. Bodoh et al., "Defect Localization Using Time-Resolved Photon Emission on SOI Devices that Fail Scan Tests", International Symposium for Testing and Failure Analysis, pp. 655-661, 2002.

5. B.R. Benware, "Driving Baseline Yields on ASICs using Logic Mapping", International Symposium for Testing and Failure Analysis, pp. 579-586, 2002.

6. M. Lane, "Making Failure Analysis a Value Add Proposition in today’s High Speed Low Cost PC Environment", International Symposium for Testing and Failure Analysis, pp. 609-615, 2002.

7. L. Andrade, "Targeted Defect Analysis for Yield Improvement", International Symposium for Testing and Failure Analysis, pp. 571-577, 2002.

8. A.J. Weger et al., "Timing Characterization of Multiple Timing Domains", International Symposium for Testing and Failure Analysis, pp. 663-666, 2002.

9. T. Blanchard, "The OpenAccess Coalition -- The Drive to an Open Industry Standard Information Model, API, and Reference Implementation for IC Design Data", Proceedings of the International Symposium on Quality Electronic Design, 2002.

10. M. Bales, "Facilitating EDA Flow Interoperability with the OpenAccess Design Database", Electronic Design Processes (EDP), 2003.

11. G. Maier and S. Smith, “Electric Test Process Limited Yield”, IBM Micronews, Vol. 6 No.1, 2000.

12. P. Song, et al., “S/390 G5 CMOS microprocessor diagnostics”, IBM J. Res. Developm. VOL. 43 NO. 5/6, pp. 899-914, 1999.

13. R. Desplats, et al., “Fault Locaization Using Time Resolved Photon Emission and STIL Waveforms”, International Test Conference, pp. 254-263, 2003.