

# EMBEDDED DIAGNOSIS IP

Stephen Pateras

LogicVision  
101 Metro Drive, Third Floor  
San Jose, CA 95110

## Abstract

*Today's market conditions are driving increasingly shorter time to market requirements for semiconductor devices. Effective techniques for achieving quick and accurate debug and fault diagnosis of increasingly complex SOC devices are therefore becoming indispensable. This presentation covers new embedded test based IP and related software tools that provide the desired level of debug and diagnosis.*

## 1. Introduction

System-on-a-chip (SOC) technology continues to drive escalating functionality while integrating extremely diverse IP cores. At the same time, these sophisticated SOCs (such as those used in advanced data communications and wireless products) are also generating increasingly difficult design and test problems. Managing these design and test challenges are key to eliminating adverse impact on time to market of these leading-edge designs.

In fact, problems for SOC designers are cropping up across the board — manufacturing test, diagnosis, and measurement. Many of the issues are ignored, perceived as back-end concerns. Unless these difficult test and validation problems are solved at the front end, the negative impact on time to market, device quality, and overall IC costs will continue to escalate. This increase in test cost is driven by higher densities, higher capital costs, multi-pass testing, complex test development and longer test times. The result of this trend is that per-unit test costs are rising more rapidly than any other aspect of chip manufacturing, significantly lowering profit margins of new designs.

Embedded test and diagnosis installs intellectual property (IP) that allows each chip to test itself and help with diagnosis. The technique integrates the tester functionality into each major function on the SOC, taking up only about one to two percent of the total silicon die area. With the capability to place most test intelligence on-chip, the need for test programs, test vector generation, and test data transfer is eliminated. This approach lessens the complexity of the hand-over from design to manufacturing, while also significantly reducing

manufacturing tests ramp time and debug from months to merely weeks.

## 2. Embedded Test and Diagnosis Capabilities

The two largest components of SOCs are random logic and embedded memories. For embedded memories, a local BIST controller is used to generate the test patterns and observe the test responses. Multiplexing logic chooses between address and data originating from the BIST controller or from the system logic

The memory BIST controller consists of three main blocks:

- A signal generation block contains the circuitry to produce the address, data, and control values necessary to create each test pattern that is to be applied to the memory. This block typically contains an up/down counter for generating the address sequences needed by most memory test algorithms.
- A comparator to compare the values read out of the memory with expected values generated by the signal generation block on a cycle-by-cycle basis. The result of each comparison is accumulated into a status flip-flop in order to provide a go/no-go result at the end of the test. Often the comparison result is brought out to a chip-pin for real-time monitoring.
- A finite state machine (FSM) is used to control the overall sequence of events. It determines, for example, if the address counter should be counting up or down or if the data being generated should be a marching 0 or marching 1 pattern.

Several features can be added to a memory BIST controller for diagnosing failures. The DONE output of the controller can be used to indicate whether the failure occurred in the memory itself or the controller. When multiple memories are tested in parallel, a separate *go/no-go* signal can be generated for each one in addition to the global *go/no-go* signal to locate any failing memories. Optional comparator status outputs can be monitored in real time to log each compare error that occurred. The diagnostic resolution is adjustable down to the bit level.

Another memory diagnostic capability is a stop-on- $n^{\text{th}}$ -error feature that can stop the controller after the  $n^{\text{th}}$  misc

scan out all the relevant information on the failure through a serial interface—typically the IEEE 1149.1 TAP interface. The BIST controller can then resume testing and stop on the next error.

For testing and diagnosing logic, existing BIST capabilities are typically based on the extension of a scan circuit (usually full scan) into a self-test capability. This self-test is performed using a pseudo-random pattern generator (PRPG) as stimuli generator and a multiple-input signature register (MISR)-based cyclic-redundancy checker (CRC) for output results compression. The logic BIST controller consists of the PRPG, the MISR, and a state machine capable of operating at full application speeds.

The sequence of operation for logic BIST is the following:

1. Random patterns are applied to the circuit through scan data-in signals to the scan paths.
2. Responses are taken into the MISR from the scan data-out outputs from the scan paths.
3. At the end of the self-test cycle, when the last random pattern has been applied, the final signature is read from the MISR and compared with the expected final result to produce a pass/fail decision. Optionally, the signature can be scanned out for comparison to a reference signature by the external ATE.

A logic BIST controller can not only be used to obtain a pass/fail result, but perhaps more importantly, it can provide diagnosis down to the flip-flop level. To achieve this level of diagnostic resolution, the basic execution flow must be enhanced to allow for an iterative search process. Assume that the signature scanned out of the MISR after typically thousands of trials is incorrect and thus a failure has been detected. This means that one or more of the trials resulted in one or more incorrect bit (flop) values being scanned into the MISR.

The most basic diagnosis would be to find the first of these failing trials. To achieve this, a binary search algorithm is used. The algorithm is based on having the logic BIST controller apply increasingly smaller (by one half) trial subsets until a single failing trail is found.

If it is desired to know which flops are failing within the failing trial, the PRPG can be scan initialized to have the controller reapply the failing trial. However, instead of unloading and compressing the scan chain values into the MISR, the scan chains are reconfigured into one long scan chain and its contents are scanned out and compared with expected values

### 3. Integrated Solution

As described in the previous section, in order to achieve a much finer grain diagnostic resolution than a simple go/no-go, it is often necessary to run an embedded test controller multiple times with different initializations in order to accumulate more fail data or zero in on a failing element. In many cases, the

setup requirements for each embedded test run depend on the results of one or more of the previous runs. This means that the test engineer, or more often the design engineer, cannot create test programs in advance to diagnose failures, but must do so while the diagnosis is taking place.

To solve this problem, a software system architecture has been developed which provides a seamless integration of embedded test within the conventional ATE environment. The new software removes the need for all of the batch oriented test pattern generation described above. This is accomplished as follows: automation tools used by designers to create and integrate embedded test blocks into the chip also generate an Embedded Test IP Access Data file for the chip. This file contains all of the information about the embedded test blocks that is needed by *Embedded Test Access* (ETA) software running directly on the tester. Since the ETA software interacts directly with the tester, it can automatically perform the necessary iterations to achieve any level of diagnosis.

The combination of the embedded test controllers and the ETA software provides for real-time at speed diagnosis of memories and logic. Full failure bit maps can be generated for embedded memories while logic can be diagnosed down to failing flip-flops or gates. Due to the at-speed nature of the embedded test controllers, diagnosis is achieved not only for static defects but for delay defects as well. Also, because of the tight integration of the ETA software with the tester, diagnosis is typically very rapid. For example, on a million gate design, diagnosis down to the first failing flip-flop requires less than one minute. This represents a tremendous time savings over the days or weeks often needed by a functional pattern based diagnostic flow.

### 4. Conclusion

This presentation describes a new approach to address the debug and testing of SOC designs. Embedded test and diagnosis provides a significant positive impact on test costs and time-to-market for new SOC designs. The combination of IP and real-time software provides a solution which enables the automated control of production go/no-go testing, production datalogging, and advanced failure diagnosis.