Why Can't DFT Totally Eliminate Functional Test?

Bulent Dervisoglu, Cadence Design Systems, Inc., bulent@cadence.com

Growing complexity of IC test issues demand using multiple approaches that complement each other in order to achieve cost effective solutions. As target IC size (i.e. equivalent number of logic gates) increases there is an obvious problem due to increased computational effort that is needed to generate test vectors. From a purely algorithmic point of view, it should be obvious that Design for Testability (DFT) is our best weapon for generating test vectors with high test coverage. Technological advances have made traditional reasons (excuses!) for not implementing adequate DFT become invalid. Thus, whereas additional gates are needed for implementing scan (internal or boundary scan) the relative cost of such gates have been decreasing rapidly, almost to the point of becoming not worth mentioning at all.

Whereas in the past it has been found adequate to deal almost exclusively with *stuck-at-fault* models this is no longer acceptable. Traditionally, ATPG algorithms have been targeted to detect only the stuck-at faults since it has been argued that by far the largest number of defects cause stuck-at behavior of the faulty circuit. Additionally, it was assumed (using empirical justification) that by targeting the stuck-at faults other types of defects are also detected in an opportunistic fashion. DFT techniques have been developed primarily to assist ATPG algorithms so that test vectors can be developed more easily and with higher fault coverage. This approach is markedly different from the earlier *Functional Test* approach which relied on the human engineers' ability to write diagnostic programs that exercise all possible functions in the target device in such a way that functions that are not executed correctly will lead to externally visible behavior that allows detection of underlining defects. Being mostly a manual effort, this is not cost effective. Furthermore, there are no adequate models that can be used to calibrate how well the functional test patterns cover the range of all potential defects. Finally, functional test patterns are more difficult to apply to the target IC, both because they tend to be too large (i.e. require too much tester memory and test cycles) and also because they require more elaborate test set setups that require increased accuracy of signal timing.

Since the beginning, arguments described above have been used to pit *Functional Test* against *Structural Test*, which is exemplified by the use of DFT and *Scan* techniques. Built-in Self-Test (BIST) is an extension of scan and uses on-chip generation of test vectors as well as using result compaction. *Path-Delay* or *Transition-Delay* test techniques are also extensions of structural testing and have been developed to catch timing related defects. However, due to the extremely large number of possible paths in an IC, Path-Delay tests can only be targeted to cover just a fraction of all possible paths. Using BIST with double-clock capture suffers from the same shortcoming. Transition-Delay tests can be applied to cover all nodes (i.e. Flip-flop outputs) but they can not be made to follow specific circuit paths. This leaves *Functional Test* as an alternative to cover those areas that are not covered adequately with the other, existing approaches. Functional testing executes at the true speed of the target device and can be more effective in catching certain timing failures compared to scan based approaches which are executed in a *start-and-stop* style.

Not only does Functional Test exist as a viable companion to the other test approaches but it has also become a method of choice in newly emerging situations. For example, *Design Verification* has become a very serious issue and as much as 55% of the total design effort might be spent in developing self-checking verification programs (plus the test benches to execute them). This presents an excellent opportunity to re map the verification programs for (re)use in IC test so that test development efforts can be amortized over multiple uses. Furthermore, the emergence of System on a Chip (SoC) has created a need for *System-Level Debug* during post-silicon verification/debug. An SoC device represents a unique blend of system-level and IC component-level features and the test/debug needs that accompany them. The best approach to address these problems is using a mixture of DFT and Functional Test concepts. For example, implementing on-chip logic analyzer [1] to capture signal sequences during regular functional operation and analyze the results off-line may lead to faster debug of internal timing issues or design errors. The same approach can be used on Automatic Test Equipment (ATE) where, with the help of additional support circuitry (to be included on the DUT board), self-checking functional diagnostics can be executed to complement structural testing.

Today, the most prudent approach for test and debug of SoC devices is to use a blend of multiple testing styles comprising of DFT-based scan, BIST, path/transition delay, Iddq testing and *Functional Test*.

References:

B. I. Dervisoglu, *Design for Testability: It is time to deliver it for Time to Market",* paper 41.3, pp. 1102-1111, ITC'99.

Proceedings of the International Test Conference (ITC'00)
1089-3539/00 $17.00 © 2003 IEEE