

Is scan (alone) sufficient to test today's microprocessors? Not quite, but we can't get the job done without it.

Grady Giles
Advanced Micro Devices, Inc.
Austin TX

It's a provocative question. Can we test contemporary state-of-the-art microprocessors with nothing but scan techniques? The objective is to hit the market window with an economically viable, competitive, defect-free product based on a reliable design. So much has to fall precisely into place and time is of the essence. Scan techniques have great value in achieving the objective.

Scan is economical. Designers are a scarce resource and scan makes very productive use of design resources. To implementation designers, scan insertion is one of the systematic tasks they perform and it is not one of their more challenging tasks. They just do it, and they can easily tell when they have completed the task. That investment enables vastly more productive test generation (let's face it, test generation engineers are designers too) and design debug than can be had without scan. Some organizations fret about the "overhead", but who has ever seen a design center adopt scan methodology and later decide to go back to the good old days before scan?

The measurable fault coverage achievable with scan patterns is somewhere between good and excellent. More to the point, it is at least as good as the coverage of functional patterns as quantified by fault simulation, because the fault simulators employ the same fault models as ATPG.

A scan architecture coupled with a few special clock control tricks and some software, makes an excellent platform for design debug. This debug use of scan greatly diminishes the risk of a protracted debug phase and consequential late product release and/or yield issues.

So the question really comes back to the sufficiency of the quality provided by scan patterns on today's microprocessors. State-of-the-art microprocessor designs and manufacturing processes are always pushing the envelope. (The competitive part of the objective stated above.) This aggressive posture with respect to the underlying technology is one cause of the demise of the stuck-at fault model and its cousins. ATPG is only as good as its fault model. Scan has less utility for testing some of the very analog features of today's microprocessors such as gigabyte/sec differential busses. Though

scan has been used in the past for I/O timing spec and speed bin testing, this is less applicable than it once was because the silicon is *SO* much faster than the ATE.

I recommend using best at-speed scan, ATPG, and BIST practices to achieve quantifiably high fault coverage, and also performing functional sequences under stress conditions. The functional sequences generally aren't modeled for faults because it is not economical to do so.

Where these functional sequences manifest systematic marginalities (yield problems), the scan debug system is there to aid in diagnosis of the problem. Most of these instances turn out to be cases where an appropriate fault model either doesn't exist or is computationally intractable. The debug often points to areas where the design rules can be augmented and thus make fault models more applicable.. For example, the root cause of failure at high frequency may be diagnosed as crosstalk (coupling between signals) and the design rules changed to make this failure mode less likely. Also the diagnosis may provide insight to make ATPG and/or the application of the scan patterns on the tester more robust.

My bottom line is that modern microprocessors are such huge and complex systems that there is no choice but to rely on systematic and automated methods such as scan, ATPG and LBIST for the great bulk of test coverage. (And of course, we need more comprehensive and tractable fault models.)

(The opinions expressed are my own and not my employer's)