

## Design for test

**In light of the recent chip recalls, design for test is mandatory. Too often, design for test has been relegated an academic exercise. Now, it must more closely approximate the actual device application.**

**By Howard Woo**

---

DESIGN FOR TEST HAS BECOME an academic exercise with little relation to the production test world. Research shows that to achieve a defect rate of less than 500 dpm, certain design elements are necessary. These design elements are automatic test-pattern generation (ATPG), scan insertion, built-in self-test (BIST), and IC quiescent current (Iddq). Technology is heading toward deeper submicron—0.18  $\mu\text{m}$  and below—with the attendant increased clock speeds of greater than 500 MHz and decreased noise margins. As technology moves in this direction, you will begin to see extraneous effects similar to those you already see in mixed-signal designs. Typically, in digital circuits, when you apply a stimulus to the device, you expect an observable event to happen. A part that behaves with the expected behavior is a “pass,” and one that does not is a “fail.” This is what all simulators predict, but it is a simplistic view. The next-generation IC will no longer enjoy the large noise margins of the current chips. As a result, devices that pass will also fail given the same stimulus. Design for test needs to approximate more closely the actual application of the device. Looking at mixed-mode testing, you can appreciate the magnitude of the problem.

Historically, chip designers were responsible for generating the test vectors to exercise the logic on the chip. Designers manually generated these test vectors based on the expected operation of the chip. However, there was no rigor to this test generation, and, consequently, some design engineers would generate too many vectors, and others would not generate enough. Typically, design engineers would generate hundreds of vectors to test initial functions and performance. Then later, they would remove vectors that had duplicate functions for the production program. These steps would take much of the designers' time, and design times would take a long time. ATPG provided a rigorous method for generating test vectors based on the design netlist, and fault-grading tools provided a measure of test coverage. Scan provided a complementary test method to catch stuck-at-one or stuck-at-zero logic gates. The drawback to ATPG was that it did not cover all the gates of the design; the drawback to scan was that it did not cover the functional operation of the circuit. Scan typically happens at speeds of less than 10 MHz. Test-coverage issues remained, and Iddq was another method that addressed the shortcomings of scan and ATPG. Meanwhile, the demands of integration increased such that logic structures existed that you could not access from outside the chip. BIST exercised this inaccessible logic. When you use it properly, the combination of these methodologies provides an estimated coverage of less than 500 dpm.

The ideal solution is that designers can design a structure so that the chip can identify itself as a pass or a fail. This solution, however, will not happen anytime soon. We are thus left trying to improve our test methodologies. To guarantee total performance, the ideal testing procedure is to test the IC at fully rated speed. The IC should be stimulated with a full test-vector set that approximates how the application will use the device. This step is easier said than done: Imagine testing a CPU and the exception-handling circuitry. It would be difficult to cover every conceivable case. If a diverse set of applications uses the IC, additional vector sets must be generated for each application. None of the current test methodologies does this.

ATPG is not based on the application of the circuit. Test generation is based on taking the netlist and applying a deterministic or random methodology. Deterministic test generation is more expensive than random test generation but produces shorter vectors and fewer vectors that overlap. ATPG already is an abstraction from the application that centers on single stuck faults further breaking down the process into fault-oriented or fault-independent test generation. Furthermore, many of the generated test patterns are not run at full speed because of limitations in the tester or tester hardware.

Scan has little relation to the application and runs at relatively low speeds. The scan method introduces registers to provide a controlled stimulus and observation points. Scan insertion can increase the die size by 5 to 15%. Scan testing is slow. Typically the scan vectors run at rates of less than 10 MHz. Depending on the test platform, you should keep scan chains at less than 4 Mbits.

Only BIST is a worthy design-for-test methodology. BIST can run at full speed, and, when you properly design it, it can approximate the actual application. However, BIST is difficult to implement. For large, complex chips, it requires a design team within a design team. Finally, Iddq does not approximate the application of the chip. Iddq's purpose is to detect faults through a statistical methodology. Iddq notes the amount of current detected under certain conditions. If the IC varies from this detected current, then the chip is potentially a fail. None of these methodologies provides an application-based test. You must then decide what is a "fair" test of the IC. All of these tests combined equate to a fair test. Results show that using all of these techniques for a digital circuit provides a defect rate of less than 500 dpm.

In contrast, mixed-mode designs typically have a defect rate of less than 1000 dpm. This fact is because of the interaction of the system noise with the performance of the chip. Instances exist in which a given chip fails in customer A's application but passes in customer B's. I expect that in deep-submicron designs, these instances—along with passing parts failing and failing parts passing—will become commonplace. You can already see these challenges in mixed-signal designs. The main challenge is control of noise. Control of tester noise, control of test-hardware noise, and control of device noise are crucial for device-testing consistency. Generally, several circuit elements are common to mixed-signal designs. First and foremost, phase-locked loops (PLLs), ADCs, DACs, and bandgap-reference PLLs are showing up more and more in digital circuits. PLLs provide frequency synthesis, and you use them to reduce clock skew. You use ADCs to digitize incoming signals for manipulation by DSP techniques. You use DACs to translate the digital signals back into the analog domain. The bandgap reference provides a "stable" source for the A/D or D/A conversions. Each subcircuit is susceptible to noise. This noise prevents you from determining whether the part is a pass or a fail.

Mixed-signal production-test development initially progressed along two fronts. Remember that all circuit elements are on the same substrate. One approach tested each individual circuit block. The belief was that if all the circuit blocks behaved according to the expected behavior, then the entire circuit was a pass. The other approach tested the entire chip by approximating the application. If the chip behaved as was necessary in an application, then the chip was a pass. Eventually, engineers found the second approach better. A percentage of chips tested block by block would not function in the application. Another percentage failed the block-by-block test but passed the application-based testing. Engineers found that the block-by-block test failed to consider the chip noise. They tested each block in isolation from the other circuit blocks. They also found that testing the device in a manner not used in the application could cause excessive failures. The test stimulus had to approximate a stimulus that the chip would see in the actual application. The reason for this necessity was that certain stimuli would cause ex-

cessive noise, resulting in device failure. Test development then proceeded with the entire chip running as it would in an application. Stimuli were similar to those you would see in an actual application.

Production-test development approximates what you would see in an application. Layouts of the production-test board approximate the application board. Applied stimuli closely approximate those you see in the application. Even the power-on and -off sequences resemble those in the application.

A concrete example is a video encoder. The video encoder is part of the system that takes signals off a DVD and converts the signals for viewing on a television or monitor. Essentially, three D/A conversions occur. Digital signals are converted into the necessary analog components for display on a screen. The chip also contains a bandgap reference and a PLL. Typically, observing the output of the encoder on a CRT screen, you see clear, distinct lines and colors. Occasionally, objects on the screen “shimmer.” If this shimmer is not an exception, it is a desired special effect. Testing each individual block fails to reveal this problem, but testing the chip as a whole does reveal it. An analysis of the shimmer shows that it results from a skew or setup-and-hold-time problem. Changing the temperature or the applied voltage of the IC could cause the effect to come and go. Even placing pressure on the package could cause the effect to change. A failure analysis of the problem reveals that the shimmer is an outcome of an instability in the PLL or the bandgap circuitry. The root cause is a process problem in the metallization, resulting in partially cleared vias. This partial contact results in a nonstandard noise pattern that the remaining circuitry picks up.

The lessons from this mixed-signal test development show that you can no longer test future designs in abstract. A relationship must exist between the test methodology—whether it is ATPG, scan, BIST, or Iddq—and the application. For instance, you cannot test a memory block in isolation and consider it a pass without the other circuit elements functioning. This may mean that a blanket register read/write test cannot guarantee a functioning register and that a register read/write test related to an actual read/write sequence is necessary.