Test&Measurement WORLD®

DEVICE TEST

Rethink fault models for submicron-IC test

The design and test community must examine stuck-at, stuck-open, transition, path-delay, and I_{DDQ} fault models to cut test escapes.

Rudy Garcia
Schlumberger Semiconductor Solutions
San Jose, CA

The electronics industry is in the midst of a transformation that is drastically changing product design and manufacture. Deep submicron process technology puts more gates on a chip, and the gate-count escalation enables the industry to make major strides toward producing smaller and faster computing, communications, and entertainment products.

The increasing design complexity and reduced error margins in semiconductor manufacturing are forcing design and test engineers to take a new look at fault models. Although ATE has improved substantially, the improvements have not kept pace with all test challenges. Rapidly shrinking feature sizes raise the specter of new types of defects, and increasing gate counts have increased the number of locations where such defects can occur.

By representing the device under test (DUT) as a gate-level model, fault models increase test-generation efficiency. Yet, applying fault models isn't easy. Each of the five major model types has weaknesses and strengths that make it ideal for some defects and a poor choice for others. The semiconductor industry needs to move from its reliance on the single stuck-at fault (SSAF) model to a more nuanced approach that recognizes the strengths and weaknesses of each of the major fault models.

Problems in the test area

Defects can be introduced at any step during manufacturing (Ref. 1). Common defects generated in the front end of the fab line include drain-to-bulk shorts on PMOS transistors, source-to-bulk shorts on NMOS transistors, and gate-to-drain or gate-to-source shorts on any transistors. The back end of the line can generate equally annoying defects, such as missing polysilicon, salicide opens and shorts, extra metal causing shorts, or resistive shorts.

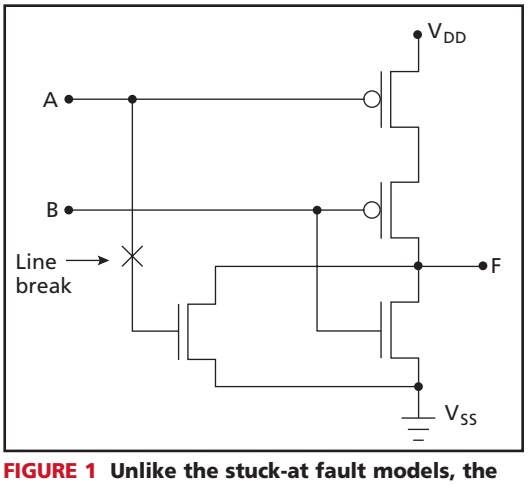

FIGURE 1 **Unlike the stuck-at fault models, the stuck-open model does not assume a short to V_{DD} or ground, but it misses opens inside gates.**

One long-time concern of semiconductor companies is the cost of testing for electrical defects. According to the International Technology Roadmap for Semiconductors (ITRS) (Ref. 2), the cost for ATE has remained essentially flat for the past 20 years at around $10,000 to $12,000 per pin, dropping recently to about $8,000 per pin. But the

demands generated by increasing design complexity have quickly offset this improvement. At the same time, while tester accuracy for timing-signal resolution has improved at a rate of 12% per year, semiconductor speeds have increased at 30% per year.

Typically, a variety of in-process tests, performed on samples, weed out gross manufacturing defects and verify process control. Examples include metrology and electrical-parameter screens as well as optical and electron-optical inspections on pre-operable dies. Tests are often performed on special test structures or on product dies. But, because of relatively low defect densities and limitations in the "in-process test" coverage, dies with random defects still exist and must be screened by later testing.

Manufacturing tests should identify these remaining random defects. Because of the impracticality of ever evaluating every possible defect in a million-gate device, fault models are used to flag detectable erroneous behavior. Although some defects, such as a high-impedance resistive bridge, may not be sufficiently pronounced at test time to be detectable, a later burn-in step or electrical stress test can precipitate many of these latent defects by accelerating their failure mechanism, eliminating "infant mortality" defects (Ref. 3).

A fault model is a description of the behavior and assumptions of how elements in a defective circuit behave. The goal of fault modeling is to model a high percentage of the physical defects that can occur in the device at the highest possible level of abstraction. The high level of abstraction reduces the number of individual defects that must be considered and lowers the complexity of the device description used in generating the test. The result is that test generation can occur earlier in the design cycle in less time with less expensive computing resources.

The gate-level model is widely accepted as the best compromise between abstraction level and the ability to represent most of the defects in the DUT. Register-transfer-level (RTL) modeling is too abstract to accurately represent many of the fault types, and switch-level modeling is too computation-intensive for most devices, although is sometimes used in defect-based testing models.

Single stuck-at fault model

The SSAF model is the most popular fault model, first published in 1961 (Ref. 4). It makes the assumption that only one line in the gate or cell is faulty at one time and that the fault is permanent as opposed to transient. The effect of the fault is the same as if the faulty node is tied to V_{DD} (a logical 1) or Gnd (a logical 0), while the other gates in the circuit are not affected by the fault. The SSAF model covers many of the possible manufacturing defects in CMOS circuits, such as missing features, source-drain shorts, diffusion contaminants, and metallization shorts.

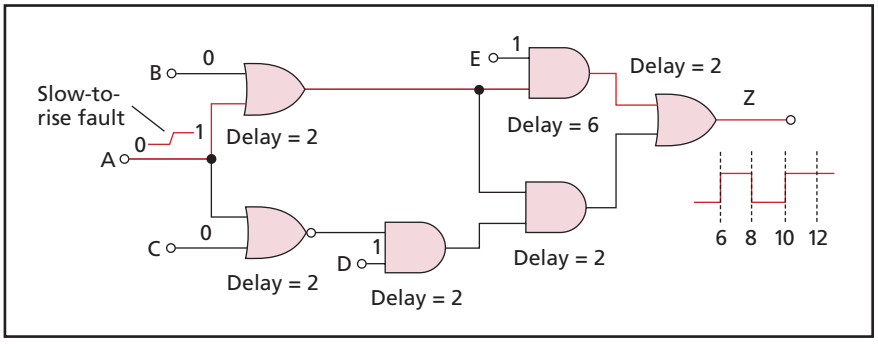

FIGURE 2 Delays can ripple across a typical circuit, engendering complexities that you must consider when modeling gate delays.

SSAF coverage can be increased by mapping other fault models such as stuck-open and bridging faults onto sequences of stuck-at faults. A key advantage of this approach is that the fault can be applied at the gate level, keeping computational requirements reasonable. SSAF also results in a reasonable number of faults—twice the number of gate nodes. In addition, algorithms for automatic test-pattern generation (ATPG) and fault simulation of combinational networks with SSAF are well developed and efficient.

The flip side is that the 1999 ITRS indicates that SSAF covers only about 70% of the possible manufacturing defects in CMOS circuits, leaving 30% of the possible defects potentially undetected. Another key problem is that SSAF does not do well on gate primitives with 3-state structures, which are commonly used in certain design styles.

Multiple stuck-at fault model

The multiple stuck-at fault (MSAF) model makes the same basic assumptions as SSAF, except it allows two or more lines in the circuit to be faulty at the same time. Although MSAF covers a greater number of defects, that increase may not be great enough to compensate for the much larger number of faults that must be analyzed under this approach: 3^n–1 for n circuit nodes. Furthermore, the algorithms for ATPG and fault simulation are much more complex and not as well developed, and commercial simulators do not support MSAF well (although some proprietary simulators have been developed around specific processes). Therefore, MSAF fault simulation is rarely used.

Stuck-open model

The stuck-open-fault (SOF) model assumes that a single physical line in the circuit is broken and that the resulting open node is not tied to either V_{DD} or Gnd. The advantage of this approach is that it covers defects that can't be detected by SSAF or MSAF models but that can be tested with vector sequences (vector pairs) of SAF tests (Ref. 5).

But if you just check SOFs at the gate-pin level, then you'll likely miss situations where the open occurs inside the gate. In **Figure 1**, if a break occurs between input A and the N-channel transistor on the bottom of the drawing, then the gate suddenly develops a "memory effect" that changes the behavior of the circuit. If an AB = 10 input is applied as part of an SAF test, there is no path from either V_{DD} or V_{SS} to the output, so the output retains its previous value. If the previous test vector had applied a 01 input, then the gate output would already be at logic 0, and the newly applied test would pass, causing a test escape.

To ensure fault detection, you need a vector pair: First, set AB = 00 (test for F s-a-0) to force F to V_{DD}. Then, follow it with AB = 10 (test for A s-a-0) to see if you can force F to Gnd, and observe the results. These kinds of open defects can occur inside any CMOS logic-gate primitive, producing similar consequences, and they could go undetected when you exclusively use the SAF model for test-vector generation.

To make matters worse, an open P- or N-channel transistor in a CMOS transmission

gate would most likely pass an SAF or a SOF vector set, because the transmission gate is not "stuck"—it just may propagate signals through it at a slower speed. (Note that transmission gates can propagate signals in either direction, and ATPG algorithms don't take the reverse propagation into account. Downstream fault propagation can produce unexpected results during test.)

SOF modeling requires a much larger number of test-vector sequences for each fault compared to the SSAF model. Algorithms for generating SOF patterns and for simulating SOFs are also very computation- and memory-intensive and not offered by many commercial ATPG tools. Many potential SOF sites require a lower-level circuit description, down to the switch level in most cases, for the development of the fault list. Alternatively, a "test model" at the gate level could map SOFs into SAFs. This approach results in a circuit with two or more times the number of gates than in the original design.

Detection of open defects is important in the new aggressive processes, and some researchers have reported that SAF vector sets with greater than 100% fault coverage do improve the likelihood of detecting many open defects (Ref. 6). The technique is termed "N-way" detection. Normally, the ATPG algorithm removes a fault from the fault list once it has generated a vector that detects the fault; thus, no additional CPU time is "wasted" coming up with other vectors that cover the same fault. In N-way detection, the automatic test-program generator does not remove the fault from the list, but is allowed to produce additional vectors that cover it (thus producing greater than 100% fault coverage).

The hope is that the additional vectors will include vector pairs that will stimulate the fault conditions. Many of the additional vectors, however, may not produce the desired results and may simply burden the ATE with additional pattern-storage requirements (some researches advocate that N should be set at about 15 for very good collateral defect detection). Moreover, N-way detection operates on the SAF model; no SOF coverage metric is produced. You know the vector set performs better defect detection, but you don't really have a metric to tell you how much better it is.

Bridging model

The bridging fault model assumes that two nodes of a circuit are shorted together. This failure mode becomes more important as the linewidths and pitch get smaller and their aspect ratio increases. The bridge is usually assumed to be a low-resistance path. Three classes of bridging are usually considered:

- bridging within a logic element, such as shorts between transistor gates, sources, or drains,
- bridging between logic nodes, such as inputs or outputs of logic elements, without feedback, and
- bridging between logic nodes with feedback.

Bridging of non-logical nodes between logic elements, such as transistor shorts across logic elements, is usually not considered.

Bridging covers a large percentage of physical faults not covered by the SSAF model, but the ATPG algorithms become complex because testing requires setting the two bridged nodes to opposite values and observing the effect. Most commercial ATPG fault models are not realistic because they assume that bridged nodes will behave as a wired AND or a wired OR; they don't consider voltage effects. In the real world, if you have a clock tree net bridged to a signal driving only two gates, the result won't be either a wired AND or a wired OR—it might well be whatever the clock signal wants it to be because of its fan-out strength.

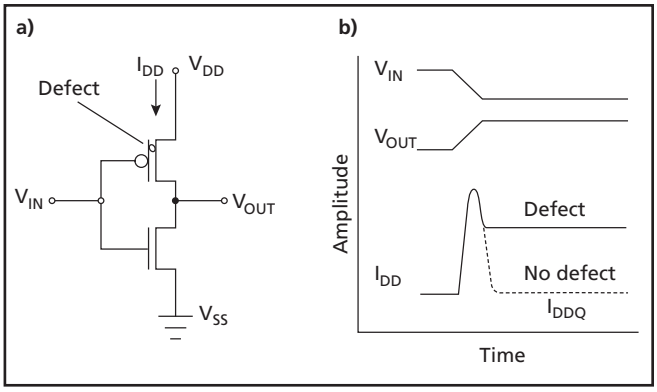

FIGURE 3 A typical defect such as a gate resistive short can be detected with I_{DDQ} tests but not by tests based on an SSAF model.

Transition delay model

Even though a circuit doesn't have a logical defect, it may have some physical defect, such as a process variation, that creates a large enough gate delay to cause problems. The transition-delay fault model detects such a fault by assuming that the logic function of the circuit under test is error-free but that a gate output may be slow to rise or slow to fall and that this time is longer than a predefined level. If the delay fault is large enough, the transition-delay fault behaves as a SAF and can be modeled using that method.

The primary weaknesses of the transition-delay model are that you need two pattern sequences for initialization and transition detection, and the minimum achievable delay fault size is difficult to determine because of timing hazards. Consequently, a whole mission clock cycle is usually used.

Figure 2 (Ref. 7) shows the complexities of gate-delay modeling. The varying levels of delays throughout the circuit create a complex pattern at the output while the internal circuit nodes settle to their final values. In this simple case, a delay fault at least as large as the time of measure (12 units) is detectable. In a more general situation, it becomes difficult to determine how small a delay fault can be before it's detectable because of the complexity of the hazards involved.

Path-delay model

The path-delay fault model is similar to the transition-delay model in assuming that the logic of the circuit under test is error-free. But instead of modeling the fault as if a single gate delay in the circuit is faulty, this model assumes that the total delay in a path from input to output exceeds some maximum value. The path-delay model overcomes a possible problem with the transition-delay model, in which other faster gates in the path may compensate for the delay of a faulty gate.

The path-delay algorithm can eliminate this problem by considering the contribution of the entire chain of possible delays. This delay model can be used with an aggressive design philosophy that pushes the process to the limit by recognizing that all gate delays are almost never simultaneously at maximum levels. This technique makes it possible to

specify a greater clock speed by statistically determining the highest level of delays that are likely to occur in the real world. A problem with this model is that the number of possible paths grows exponentially with the number of nets.

I_{DDQ} model

I_{DDQ} modeling takes an entirely different approach, measuring power-supply current rather than making the Boolean voltage measurements made by all the other fault models. The I_{DDQ} approach is based on the fact that a fully static CMOS gate consumes significant current only when switching and that the quiescent current for a static CMOS transistor is typically in the nA range.

Many types of defects raise this current by a detectable amount. **Figure 3** examines an inverter with a gate-resistive short in the P transistor. When the input voltage goes low and this device turns on, the I_{DDQ} increases substantially as current flows between the source and gate. This defect can't be detected with SAF testing because the gate output goes high at the right time. Although this defect doesn't affect device function, most semiconductor manufacturers still want to detect it because it may cause a functional fault after a certain period of operating time.

A key advantage of I_{DDQ} testing is that test generation is relatively easy: Faults simply need to be activated and measured at the power supply. This makes it possible to detect many problems that can't be seen with stuck-at models, such as bridging faults, gate-oxide defects, and shorts between any two transistor terminals. Many defects in the examples above may not affect the logic of a circuit but may affect reliability.

A disadvantage of this approach is that fixturing constraints lengthen measurement times. Furthermore, there is no absolute way to determine the I_{DDQ} pass/fail threshold for bad devices, forcing reliance on an arbitrary or empirically derived threshold. Also, the circuit under test must contain only static devices, ruling out the use of dynamic circuitry, pull-ups or pull-downs on I/O buffers, and speed-optimized circuitry such as RAM sense amps that draw significant static current. Another problem with I_{DDQ} modeling is that the increasing background leakage found in deep submicron circuitry interferes with measurements.

One size doesn't fit all

The bottom line is that each model is valuable in certain situations yet also has limitations. Problems with observing and controlling the states of circuit nodes, along with the smarts of the ATPG algorithm itself, can cause undetectable, abandoned, or redundant SAFs during ATPG. SOF testing is limited because most commercial automatic test-pattern generators do not offer SOF models, and larger designs require enormous computing resources. The key limitation of the bridging model is that most commercial ATPG models unrealistically model the bridged nets as ORs or ANDs and therefore don't consider voltage effects as they would occur in a resistive-bridge situation.

The two delay-fault models, transition and path models, are not offered on some commercial automatic test-pattern generators, and they often require enormous computing resources. In addition, the vector pairs required for detecting many SOFs and delay faults may not be deliverable to the DUT. For example, if the vector needed to sensitize the path is 1011 and the vector required to launch and or detect the transition is 1101, you can't apply the vectors using "D-mux" scan flip flops. Thus, the inability to deliver the correct states leads to untestable faults and loss of defect coverage.

Finally, with I_{DDQ} testing, relatively long test times limit the number of internal states on which measurements can be made, while rising transistor counts, together with the lowering of threshold voltages, are increasing the background leakage current, tending to mask many theoretically detectable I_{DDQ} faults. This was first quantified in 1996 (Ref. 8). Specifically, for every 70-mV drop in the threshold voltage, V_t, there is a full order of magnitude increase in leakage current.

Note that the trend toward deeper submicron geometries is altering the statistical distribution of defects. Generally, the number of defects that are detectable by SAF models is being reduced. The incidence of open vias, resistive bridges, salicide opens, gate-oxide defects, and so on is on the increase. This trend, combined with a continuing reduction in allowable defects-per-million levels, now falling below 500 for microprocessors and chipsets, means that SAF models often by themselves no longer cover enough defects to meet required quality levels.

The move toward deep submicron geometries is also creating new classes of timing-related defects that currently have no adequate fault models. Delay defects produced by crosstalk, noise coupling between signal nets and supply rails, and interactions between the analog and digital section of the chip are but a few examples (Ref. 9).

Clearly, the semiconductor industry's reliance on SAF models needs be replaced by an effort to take advantage of multiple fault models. Unfortunately, the wide range of designs and processes in use means that no single approach is a panacea. You need to evaluate each application to determine which fault model or, more likely, which combination of models will provide the required level of defect coverage. **T&MW**

References

1. Wolf, Wayne Hendrix, *Modern VLSI Design: Systems on Silicon*, 2nd ed., Prentice Hall PTR, 1998.
2. "International Technology Roadmap for Semiconductors, 1999 Edition," public.itrs.net.
3. Vigrass, William J., "Calculation of Semiconductor Failure Rates," Harris Semiconductor, available for download at rel.semi.harris.com/docs/rel/index.html.
4. Roth, et al. "Techniques for the Diagnosis of Switching Circuit Failures," Proceedings of the Second Annual Symposium on Switching Circuit Theory and Logical Design, October 1961.
5. Wadsack, R.L., "Technology Dependent Logic Faults," Bell System Technical Journal, Vol. 57, May–June 1978.
6. For example, see McCluskey, Tseng, "Stuck-Fault Tests vs. Actual Defects", *Proceedings of the International Test Conference 2000.*
7. Waicukauski, J.A., et. al., "Transitional Fault Simulation," *IEEE Design and Test*, April 1987.
8. Williams, T.W., et. al., "I_{ddq} Testing for High Performance CMOS—The Next Ten Years," *Proceedings of The European Design and Test Conference, 1996,* IEEE. computer.org/proceedings/EDTC96/TOC.HTM.
9. Cheng, K.T., S. Dey, M. Rodgers, and K. Roy, "Test Challenges for Deep Sub-Micron Technologies," Design Automation Conference, 2000.

Rudy Garcia *is the strategic marketing manager at Schlumberger Semiconductor Solutions, ATE Division. He has several patents covering aspects of ATE architectures, and he was Chairman of the Virtual Socket Interface Alliance (VSIA) Test Development Group in 1998–99. E-mail: garciar@san-jose.tt.slb.com.*