Using cell-aware automatic test-pattern generation and simulation, you can find defects that other methods might miss.

BY RON PRESS, MENTOR GRAPHICS

Traditional IC pattern-generation methods focus on detecting defects at gate terminals or at interconnects. Unfortunately, a significant population of defects may occur within an IC’s gates, or cells. Many internal defects in cells can be detected with traditional test methods, but some require a unique set of stimulus to excite the defect. A cell-aware ATPG (automatic test-pattern generation) method characterizes the library cell’s physical design to produce a set of UDFMs (user-defined fault models). Thus, the method uses the actual cell-internal physical characteristics to define and target faults.

In addition to explaining how cell-aware ATPG works, I’ll also use published simulation results from two major IC companies to highlight the test method. Production silicon test results using cell-aware UDFM have shown notable improvement in DPM (defects per million) beyond what stuck-at and transition patterns detect. As a result, cell-aware UDFM is garnering attention from manufacturers in the semiconductor industry.

A brief history of IC test
“Defects” are the actual problems or production issues that cause an IC not to function properly. “Faults” are models that try to represent defects with simple properties that correlate to defects and are easy for ATPG tools to use.

When ICs were first developed, their functions were fairly simple, and tests simply checked the IC’s functional operation. An engineer would design a “functional test” that checked whether the IC functioned as intended.

As IC technology advanced, it became impractical for an engineer to manually create a thorough functional test for the device. Increasing sequential logic such as flops and latches within ICs further complicated functional test. It could take many tens of thousands of clock cycles to propagate data at the IC’s input through the sequential logic, so it became almost impossible to create a functional test that could execute in a reasonable time and provide a high level of detection for all possible defects.

The solution was to implement scan DFT (design-for-test) structures within the device. Scan logic essentially turns sequential logic into shift registers, which are control-and-observe points that a tester can load and observe. The remaining test problem is the combinational logic between the sequential logic. Thus, the entire design is turned into many sets of small combinational logic surrounded by virtual control-and-observe points. This situation lends itself to automation using scan ATPG tools. Scan testing is considered a “structural test,” because the logic gate segments are tested without specific tests of the intended function of the IC.

ATPG circumvents the need for detailed knowledge of the IC design. The scan structure also produces very high defect detection. Standard scan testing is based on a stuck-at fault model that considers a potential stuck-at-0 and stuck-at-1 fault at every gate terminal. The stuck-at fault model verifies that gate terminals are not “stuck” at logic-0 or logic-1 states.
Somewhere between the times when 130-nm and 90-nm process technologies were developed, new timing-related defects occurred that demanded special at-speed tests. One type of at-speed scan test, called transition patterns, was used to target and detect the timing-related defects. Like stuck-at scan tests, transition tests use scan cells as control-and-observe points. After a transition test loads the scan cells, however, it puts the IC in functional mode and applies two or more at-speed clock pulses.

Stuck-at and transition scan tests, therefore, are the foundation of most production test methods; they can be automated within ATPG tools, and they can achieve high test coverage because of their structural nature. In recent years, newer scan tests have been introduced to target defects that escape stuck-at and transition tests. Examples include timing-aware ATPG, deterministic bridge, multiple detect, and hold-time methods (Ref. 1). Each of these methods provides some amount of improved defect detection.

All of these scan test methods use fault models that define fault sites at the IC gate boundary. Stuck-at-fault models, however, also detect the majority of production defects such as bridges, opens, and even many defects within the gates. With more recent fabrication technologies, the population of defects occurring within cells is significant, perhaps amounting to roughly 50% of all defects (Ref. 2). Thus, it is important to ensure that you properly define fault models that target these “cell-internal” defects.

Cell-aware ATPG
To target the cell-internal defects, test engineers can now use the physical design of gate cells to drive ATPG. This involves performing a library characterization to determine where defects can occur and how they would affect the operation of each cell. The result of the characterization is a UDFM that describes all the cell inputs and responses necessary to detect the characterized defects. A cell-aware UDFM file would be
produced for a physical library for a particular technology. Then, any design using that technology library just needs the corresponding cell-aware UDFM file for cell-aware ATPG.

UDFM is a term used to describe an ATPG tool capability that lets you custom-define fault models. You might want to use a UDFM for ATPG if there is a particular type of pattern that you want to apply to a library cell, to an instance, or between instances. The definition of the UDFM is similar to stuck-at and transition patterns. You state the values at the cell or instance inputs and indicate what the expected response is for any number of desired cycles. Once the ATPG tool loads the UDFM file, it can target the custom-defined faults. UDFM provides the framework for many types of custom fault types.

### Cell-aware characterization flow

The first step in creating cell-aware tests is to characterize the cells within a technology library. First, you must perform extraction on the physical cell layout library. Then, you can use the parasitic capacitances and resistances to locate potential sites for bridges and opens. (Capacitors represent potential bridges, and resistors represent potential opens.) Next, you define the type of defects you want to model. For example, a basic hard short can be modeled by a 1-Ω resistive bridge at the capacitor locations. Studies have shown value in modeling several resistive bridge values (Refs. 3 and 4).

With the definitions in place, you can perform an analog fault simulation with the desired defects, such as a 1-Ω bridge. The simulation is performed on all possible input combinations with one defect site at a time. The results are compared to the defect-free responses. If any of the responses differ from those for the defect-free case, then that sequence is said to detect the particular defect. Once you perform the analog simulation for all cell-input sequences, for all defects being modeled, and for all cells in the library, you will have a defect matrix. Finally, you can use the defect matrix to generate the actual cell-aware UDFM file used by ATPG. **Figure 1** shows the cell-aware characterization and ATPG flow.

### Cell-aware ATPG makes a difference

Why is cell-aware ATPG necessary for finding defects that stuck-at and transition patterns presumably miss if production tests based on stuck-at and transition have been effective for many years? The need for cell-aware ATPG arises from the increased use of complex cells and the growing distribution of defects occurring within those cells.

Many library cells won’t see any advantage to performing cell-aware ATPG compared to normal stuck-at or transition ATPG. For example, a buffer, an AND gate, or an OR gate needs no special inputs to detect cell-internal defects. Consider a 3:1 mux gate. **Table 1** shows the logic on the logic for the mux. These are the values that are needed to detect all stuck-at faults (stuck-at 1 and 0 at each cell boundary pin). (continued)
**Figure 2** shows the logical view that the ATPG uses along with the physical layout of the cell. In this layout, a bridge at location R4 could cause a short from S1 to D2. If a value on D2 dominates over S1 in the presence of a bridge, then the logic-test patterns might not detect the bridge at R4.

Although the pattern set in Table 1 will achieve 100% stuck-at coverage for the mux, it doesn’t ensure that R4 or several other cell-internal bridges will be detected. In this case, the patterns in Table 2 would be needed to detect the R4 bridge. Other complex gates would have similar situations.

### Industrial results

Several IC companies have used cell-aware ATPG to improve defect detection. NXP Semiconductor used a cell-aware UDFM tool to perform cell library characterization and reported expected cell-internal detection improvement (Refs. 3 and 4). Published results can help you determine whether to model hard bridges such as a 1-Ω resistance or a variety of bridge values (Ref. 4). Cell-aware fault simulations on the 1-Ω bridge case showed an average of 1.2% cell-internal fault coverage improvement compared to stuck-at tests for 10 designs.

Bridges are the most popular type of modeled defect, but there are many types of defects that you can model using the cell-aware characterization. Another defect that some users are modeling is the internal opens defect (Ref. 4). AMD published test results based on applying cell-aware patterns to 600,000 ICs using a 45-nm process (Ref. 5). The results showed that cell-aware patterns detected defects in 32 devices that passed stuck-at and transition patterns. That correlates to a 55 DPM improvement, which is significant for many production environments. More significant DPM improvements have been observed on a 32-nm process IC using slow-speed and at-speed cell-aware patterns.

### Choosing the best tests

With cell-aware and other types of fault models and test types, you may have trouble deciding which and how much of each to use in production. Most IC test sets have stuck-at and transition patterns as a baseline. There are a few methods for choosing an effective pattern set. Effective and efficient production results require good data about the defect distribution and effectiveness of tests. Often, such data is not clear, because defect distributions vary with technology nodes, operational frequencies, slack margins, and design-for-manufacturability rules.

Here are two methods for determining an effective test set. Each requires some investment to apply the tests and determine their value:

- **Using field returns.** Field returns are devices that passed production tests and were shipped as functional, but that later failed. If you have a population of such devices, you can use them to find the value of additional tests. As a first step, retest the parts to ensure they didn’t break after shipment.

  You can apply a full set of tests for any type of potentially valuable test pattern. Then, use the percent detection and pattern set size to equate a relative value of the test type. For example, if you have 300 field returns from a production of 100,000 parts, then detecting 50 devices with cell-aware ATPG would imply you could improve DPM by 500 DPM, if cell-aware ATPG was part of production test. You can use a
similar approach if you have a thorough system-level test that finds defective parts that passed production test.

**Adaptive tests for production.** Another approach is to add a set of additional patterns to the existing stuck-at and transition pattern sets. Often, there is not much spare room to apply new pattern sets in production. The additional patterns need not be complete sets. You can add 1000 patterns for each pattern type that you are interested in. After some volume of production test, you can observe the number of unique detects from each of your additional patterns. You can use these results to increase the pattern types that are detecting more defects and decrease the size of less-effective patterns.

Data from these tests gives you some insight into the defect distributions based on the DPM detection of tests and their calculated test coverage. From that, you can extrapolate the value of a full pattern set or the detection value of using a smaller pattern set.

The test pattern types that have shown the most promise beyond stuck-at and transition patterns are timing-aware and cell-aware. Gate-exhaustive tests apply every combination of inputs to each cell. They have good detection but are unreasonably large pattern sets. Cell-aware is a subset of gate-exhaustive patterns that only include stimulus combinations that can cause the modeled defects to be detected.

The new cell-aware ATPG flow allows test engineers to target subtle shorts and open defects internal to standard cells that are not adequately detected with the standard stuck-at or transition fault models. Cell-aware testing has been proved to increase the quality of manufacturing test by providing higher defect coverage and lower DPM.

**REFERENCES**


Ron Press is the technical marketing manager of the Design for Test products at Mentor Graphics. The 25-year veteran of the test industry has presented seminars on DFT and test throughout the world. Press co-authored a patent on clock switching and reduced-pin-count testing and received the Raytheon Co. inventor’s award. Press is a member of the International Test Conference Steering Committee, and he earned his BSEE from the University of Massachusetts. ron_press@mentor.com.