



A Whitepaper:

Device Modeling Is Critical for Fast Time-to-Test

By Adam Ley
Chief Technologist
ASSET InterTech, Inc.

© 2004
ASSET InterTech, Inc.
Richardson, Texas

Table of Contents

Executive Summary 3
Starting the Test Generation Process 4
ScanWorks' First Pass 4
Seeking Opens 6
Iterative Passes..... 7
Optimizing Automatic Test Generation..... 7

Executive Summary

Automatically generating JTAG tests is a balancing act among the time it takes to develop a test (time-to-test), board safety and test coverage. Over the years, the algorithms behind the ScanWorks® test generator have continued to improve on how non-boundary-scan device models are handled, to the point where test coverage can be optimized in record time without compromising board safety. Through an iterative process involving successive passes of the ScanWorks test generation engine, the test engineer or technician can quickly produce an effective boundary-scan test. The test developer can then decide how best to invest his or her time to optimize the test further.

Starting the Test Generation Process

How ScanWorks' boundary-scan test-generation engine handles the non-boundary-scan devices in a design is critical to the time it takes to develop a test (time-to-test), the amount of coverage the test will achieve and the safety of the board. Over the years, ScanWorks' test-generation engine has been improved repeatedly with an eye toward reducing time-to-test with optimal coverage and impeccable board safety. The way that ScanWorks handles cluster (that is, non-boundary-scan) device models plays a major role in this.

A critical goal for ScanWorks is to require as little manual involvement as possible while generating tests with an acceptable level of coverage. After a test has been generated, the user can decide how much effort should be expended to improve the test coverage or to modify other aspects of the test's operation.

ScanWorks addresses this goal from the very beginning of the boundary scan test development process – the building of a complete description of the design. During this preliminary step, many of the cluster models and BSDL (Boundary Scan Description Language) files may be accessed and automatically downloaded from the device libraries on ScanWorks' maintenance benefit web pages. In addition, many user organizations maintain their own in-house model libraries from which ScanWorks can automatically extract models and incorporate them into the design description.

It is always a good practice for the test engineer or technician to make sure that all of the passive devices, such as inductors, capacitors, resistors and others have been identified and their models included in the design. As the test developer is reviewing the design, he or she should note other types of cluster devices but they need not go to the trouble of modeling them during the initial design description. Should additional coverage be required later, the developer will already know which device types to consider modeling in order to achieve greater coverage.

ScanWorks' First Pass

Once the test engineer or technician is ready for ScanWorks to take its first pass at generating a test, the test-generation engine begins its analysis of the design. ScanWorks examines the cluster models for all non-boundary-scan devices to determine each one's operating modes. For example, a typical "245" eight-bit transceiver would have three modes: isolation, where no data is being transferred because all outputs have been turned off; A-to-B mode, where A pins are enabled as inputs and B pins are enabled as outputs; and B-to-A mode, where B pins are enabled as inputs and A pins are enabled as outputs. (See Figure 1.)

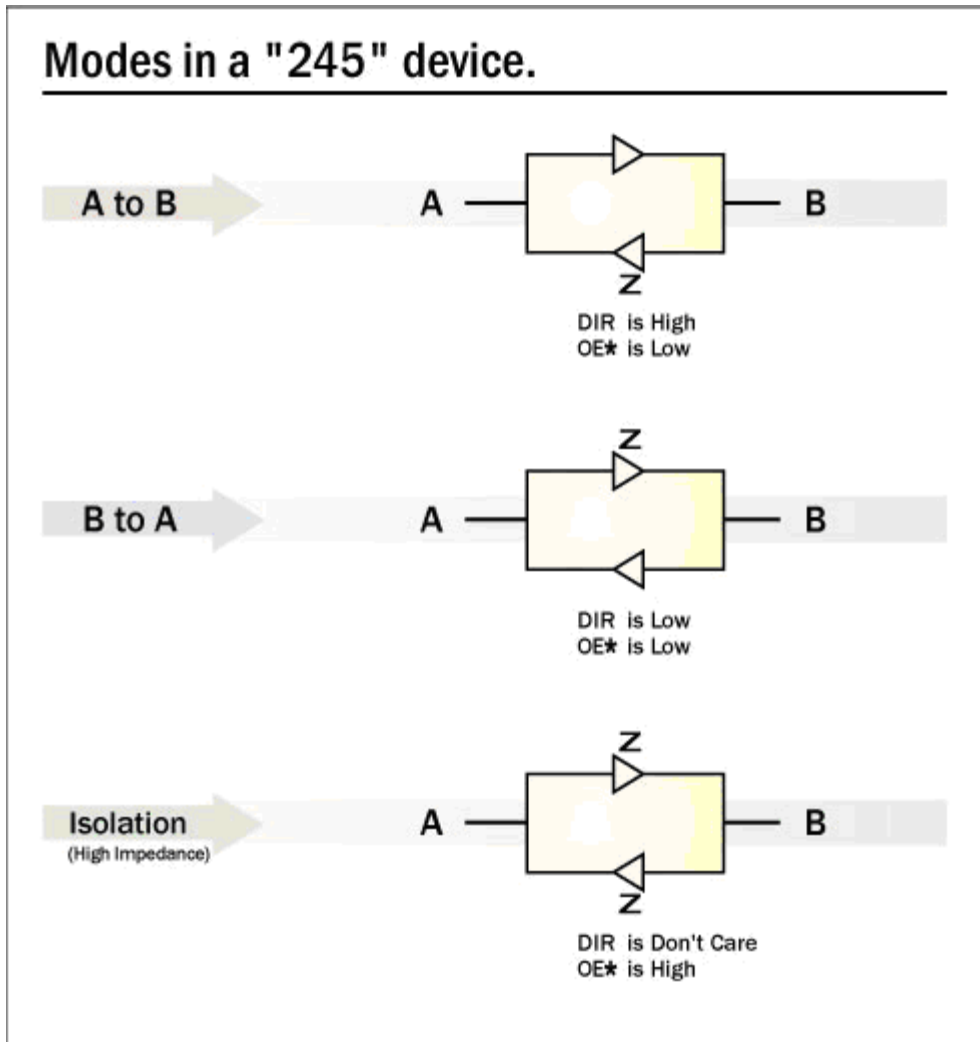


Figure 1. Modes of a "245" Device

Subsequently, the ScanWorks test generator begins to build a test by selecting each device's operating mode so that, first, the safety of the printed circuit board will be ensured and, second, test coverage for shorts will be maximized. To do this, ScanWorks must examine each device within the context of the design.

For example, if ScanWorks has access to a non-boundary-scan cluster device from one direction but does not have enough information about the device and the design to know what is beyond the device, then it cannot safely drive test patterns beyond the device. To do so would be to jeopardize the safety of the board by setting up a situation where bus contention might arise. In this case, boundary-scan test coverage would extend to the cluster device that is accessible from a boundary-scan device but not beyond it. Test coverage can be extended later, at the discretion of the test engineer or technician who is developing the test. (In Figure 2 below, test coverage can be extended beyond a non-boundary scan device because the model of the interconnected device has been included in the ScanWorks test.)

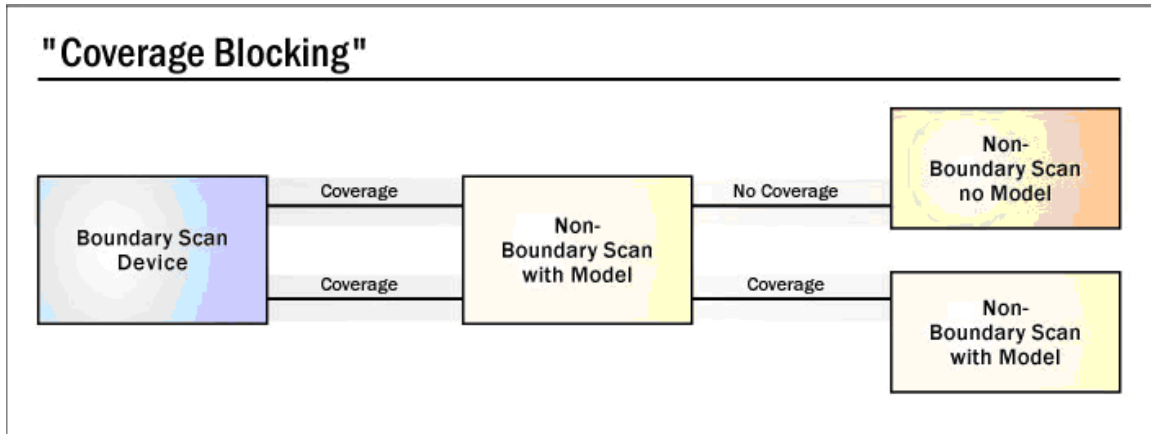


Figure 2. Test Coverage Blocking

This is a complex process because the operating mode selected for each device has a ripple effect on other devices in the design. The algorithms used by ScanWorks' test-generation engine are continually fine-tuned to optimize this process. Once operating modes are selected for all devices in the design, the test vectors for shorts are generated.

Seeking Opens

Following this first pass, the test generator automatically performs additional passes which seek to identify the operating modes that devices can be placed into so that opens can be tested. Successive passes then add test coverage for opens which was not gained during the test generator's first pass. Once this information has been processed, additional test vectors for opens are generated and included in the test set.

Of course, during any test-generation pass, the test generator identifies any cluster device's operating mode that simply must be taken into account. These types of modes are often hard-wired into the design. (See Figure 3 below for an example.) When the test generator encounters these conditions, it will constrain the test so that the hard-wired operating modes are maintained.

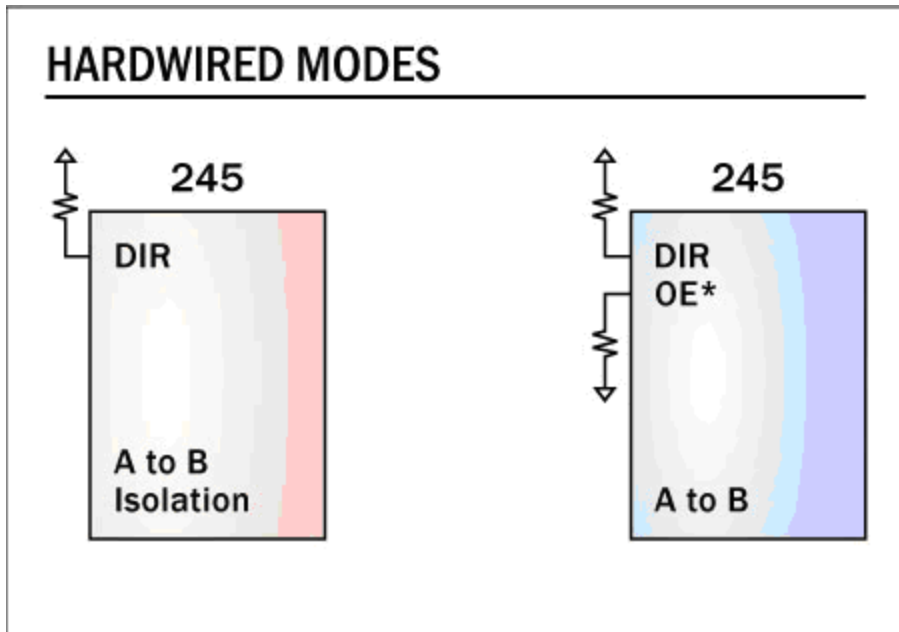


Figure 3. Hardwired Modes of a "245" Device

Iterative Passes

After the initial test generation, the test developer may continue to invoke the ScanWorks test generator to make additional passes to further maximize test coverage. At this point, the engineer or technician who is developing the test can decide to provide additional information to ScanWorks, such as cluster models for those devices that have not been modeled. This added information is used by the test generator in these successive passes to obtain higher and higher levels of test coverage.

This type of iterative process gives the test developer the option to invest his or her development time where it will have the biggest payoff in terms of test coverage or other test performance criteria. Quite often, ScanWorks will generate a test with the level of coverage required by the developer. Later, when the developer has time to devote to it, the test can be passed through the ScanWorks test-generation engine again with additional modeling information and the test coverage can be increased.

Throughout this process, ScanWorks continues to evaluate the operating modes selected for cluster devices in light of any possible bus conflicts which may ensue. Should the developer manually dictate an operating mode that might establish a hazardous bus contention, ScanWorks will notify the user of the possibility for board damage. At that point, the developer has the option of continuing with the operating mode originally selected or revising the mode to avoid the possibility of bus contention.

Optimizing Automatic Test Generation

There are many facets to automatic test generation, including time-to-test, test coverage, board safety and requirements for manual developer involvement. Of course, the most effective test-generation engine simultaneously optimizes all of these factors to provide the best balance for the inevitable tradeoffs that arise. For example, a test with high

coverage might endanger the safety of the board. The algorithms of ScanWorks' test-generation engine provide an efficient way to quickly generate high-coverage and safe tests while giving test developers the information they need to further refine the tests generated and decide how best to invest their time.