SYNOPSYS®

# Hierarchical Scan Synthesis Methodology Using Test Models

## High-Capacity/Performance Scan Synthesis Technology Backgrounder

October 2001

Contents


- Overview
- Introduction
- Hierarchical Scan Synthesis with Test Models
- Design Flow for Hierarchical Scan Synthesis with Test Models in DFT Compiler
- Interface Logic Models (ILMs) and Test Models
- Conclusion


©2001 Synopsys, Inc.

## Overview

As the design community moves towards system-on-a-chip (SoC) technology and design complexity soars to multi-million gate implementations, the swift convergence of timing, area, power and test becomes essential. Teams of designers use Synopsys' logic and physical synthesis technology to implement today's complex designs. To handle the test synthesis of such large designs at the chip level, some level of abstraction is required. This allows the system/chip integrator to implement test synthesis and reduce iterations to achieve both timing and design for test (DFT) closure. Abstracting DFT information in the form of a test model, along with timing and placement information in the logical and physical synthesis domains, helps the designer make fundamental decisions to design the test structures very early in the design cycle. This abstraction also enables quick hierarchical test implementation of multi-million gate designs, which requires less memory and significantly improves run-time performance.

## Introduction

In today's complex designs, DFT is becoming an integral part of the design process to achieve testability goals of the design and enable efficient manufacturing test. DFT Compiler today supports a 1-Pass scan synthesis flow that tightly integrates scan with both logical and physical synthesis. It allows for seamless scan insertion and meets all design constraints in the presence of test logic. It also considers the physical information of the design to minimize routing congestion due to scan chain stitching.

For complex designs, a hierarchical approach is used as a preferred methodology where system architects specify the system using hardware description languages (HDLs) and synthesis tool command scripts. They then partition the system into constrained modules. Preliminary floorplanning information can be used to set realistic constraints. There are two types of module constraints for logic synthesis. *Design rule* constraints reflect technology-specific constraints that functional designs must meet. These include the maximum net fanout etc., and are typically determined by the target technology library and placement information. *Optimization* constraints are design goals and restrictions that do not prevent the design from functioning. Examples include maximum delay and area. Testability is addressed at each of these constrained sub-modules and the test structures are then assembled at the chip level to meet all testability requirements along with timing, area, power and physical constraints. System architects integrate signed-off modules to produce system descriptions. System netlists are exported to physical design tools for floorplanning, placement and routing. Post-layout timing information is extracted and back-annotated onto the synthesized system description. Constraints are checked and the design is re-optimized if any constraints are violated. The result is a system that meets optimization and design rule constraints. Scan DFT is now mandatory for many high-quality SoC implementations. For designers with challenging functional goals, it is essential that they take DFT into consideration early in the design cycle, and scan must be integrated into their synthesis methodologies.

Synopsys recommends two broad categories for integrating scan into hierarchical synthesis methodologies. The first is a *top-down* methodology [Figure 1] that defers scan insertion until system integration is complete. Module designers make sure that their designs pass pre-scan design rule constraints, and system integrators ensure that the integrated system also passes pre-scan design rule constraints. They then devise appropriate scan constraints and run the system through scan insertion, design rule checks and reoptimization. During layout, physical design information such as physical cell locations may become available allowing layout-driven scan chain reordering. System integrators need to run post-scan design rule checks to ensure that reordering has not introduced any scan design rule violations. Some top-down methodologies defer scan chain routing until physical design information is available. Synopsys has recently introduced the integration of DFT Compiler within Physical Compiler to perform 1-Pass scan ordering. This integration does synthesis and placement, and also stitches scan chains based on placement information to achieve both DFT and timing closure with less routing congestion.

©2001 Synopsys, Inc.

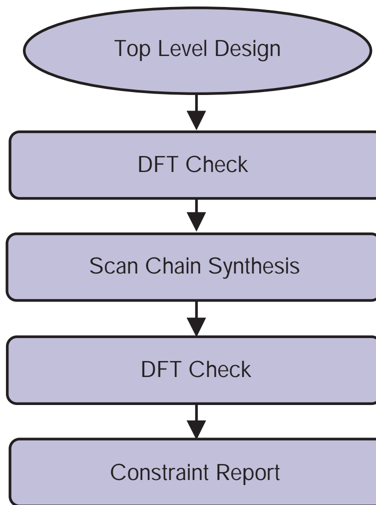


**Figure 1: Top-Down Scan Synthesis Flow**

The second hierarchical methodology recommended by Synopsys is the *bottom-up* methodology [Figure 2] for scan synthesis of large designs. System architects devise design and scan constraints and communicate them to individual module designers. Module designers perform scan synthesis and make sure that the module meets the functional timing constraints. System architects ensure that the integrated system passes through pre-scan design rule checks and they perform scan synthesis at the top-level. After layout, system architects can order scan chains using physical information in the Physical Compiler environment.

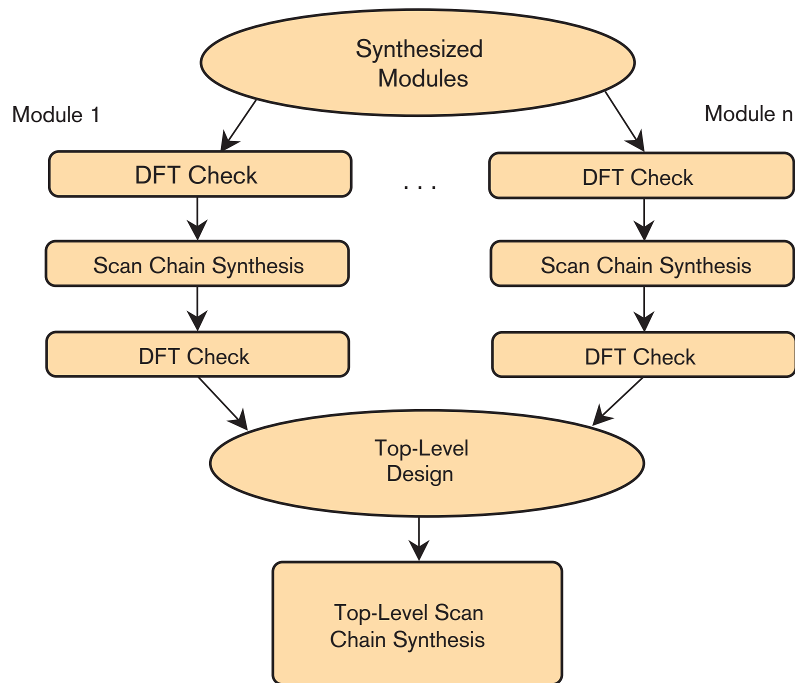


**Figure 2: Traditional Hierarchical Bottom-Up Scan Synthesis Flow**

©2001 Synopsys, Inc.

Bottom-up scan insertion has several advantages, including:
• Identifying scan design rule violations early in the design flow;
• Reducing the risks of time-consuming, top-level reoptimization steps;
• Facilitating parallel development and sign-off of scan modules.

Top-down scan insertion has the virtue of simplicity, but has implications for large multi-million gate designs due to the capacity and run-time limitations of synthesis tools. However, with designs reaching millions of gates and sub-modules also increasing in relative size, synthesis tools have reached their capacity limit for top-level scan synthesis, even with the bottom-up approach. Hence, some level of abstraction is needed at the sub-module level, and integration of these models at the top-level enhances capacity as well as performance for large designs. Synopsys has introduced the hierarchical scan-synthesis methodology with test models, which capitalizes on the proposed IEEE 1450.6 Core Test Language (CTL) standard to perform scan synthesis for the next generation of complex designs and SoCs.

## Hierarchical Scan Synthesis with Test Models

As part of an industry-wide effort, the IEEE is in the process of standardizing the elements of test technology so that plug and play can be achieved when testing SoC designs. CTL is a proposed standard, which aims at describing the necessary information for test infrastructure along with test patterns reuse, while satisfying all test needs during system integration.

Synopsys has pioneered efforts to take advantage of the proposed CTL standard to abstract scan and other test-related information into a test model (ctl-db). This test model is created during scan synthesis with DFT Compiler and is completely transparent to the user [Figure 3]. The user can then write the test model, which only has the test relevant information, along with the full-gate netlist information usually done in a typical bottom-up flow. At the top-level, only test models are read into memory along with the top-level netlist, and scan-stitching is performed without having to read in all the gate-level information of the sub-modules. This capability to read in test models and perform a top-level scan design rule checks along with scan-stitching significantly improves the capacity and performance of large designs.

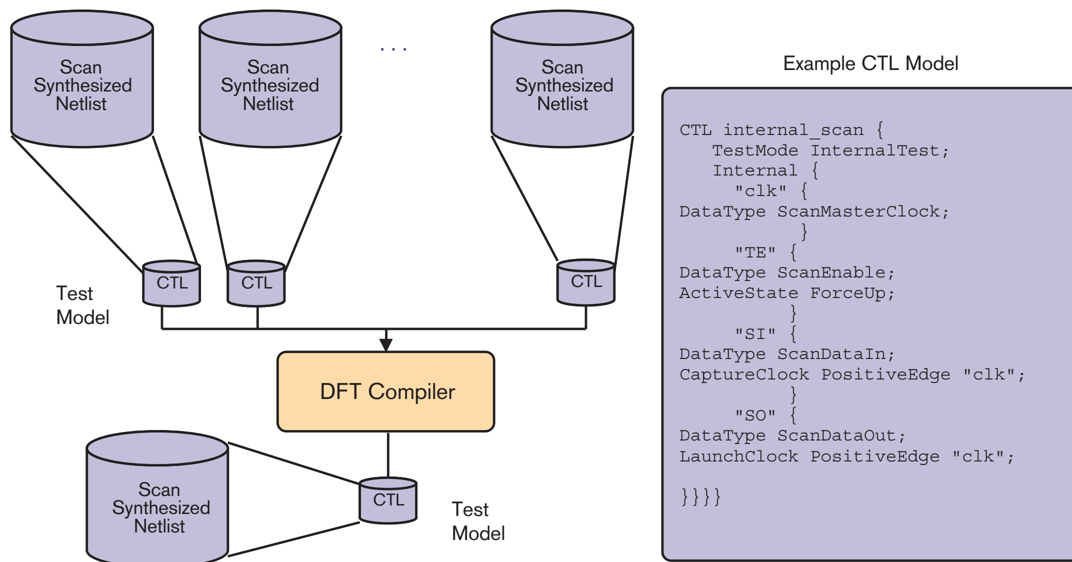


**Figure 3: Hierarchical Scan Synthesis with Test Models**

©2001 Synopsys, Inc.

## Design Flow for Hierarchical Scan Synthesis with Test Models in DFT Compiler

Figure 4 shows the typical bottom-up scan synthesis flow for sub-modules, which could be in the order of 500K to one million gates in today's ASICs and SoCs. The same bottom-up flow is extended to write out the test model that is created automatically and stored in memory after scan-chain synthesis in DFT Compiler. The test information of each sub-module is represented in the test model, and can be saved in a Synopsys internal database (ctl-db) format apart from a full gate-level netlist in an HDL format or database format. This test model is very small in size in comparison to the full gate-level database with timing and other design attributes.

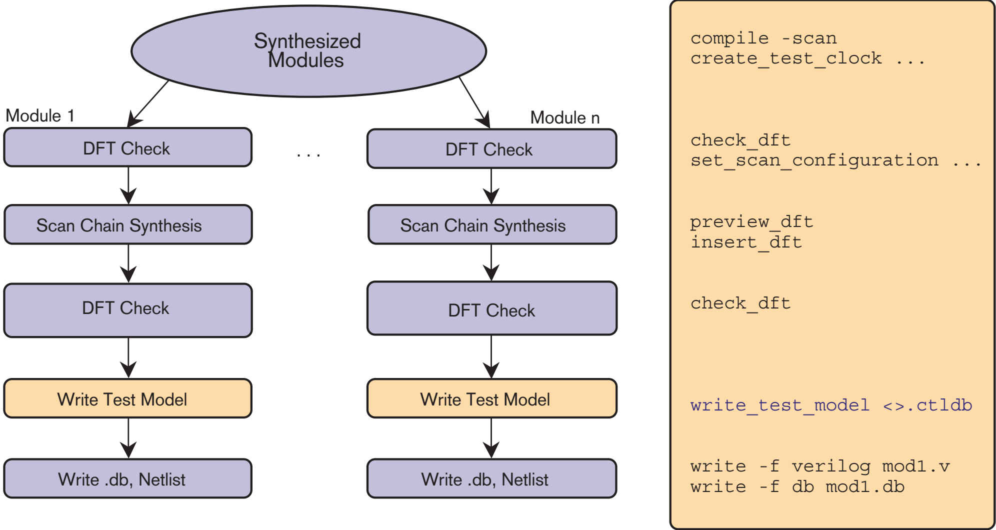


**Figure 4: Module-Level Scan Synthesis Flow**

During the top-level chip assembly [Figure 5], the test models for all the sub-modules, along with the top-level design description, are loaded into DFT Compiler. This avoids reading in the full gate-level database of each sub-module. Scan design rule checks for the top-level design can be performed easily and the scan-chain architecting can be performed to balance and stitch the top-level scan chains. This methodology allows existing users to continue using the same DFT Compiler bottom-up flows, and yet take advantage of this powerful new capability to achieve significant run-time performance and reduced memory usage.

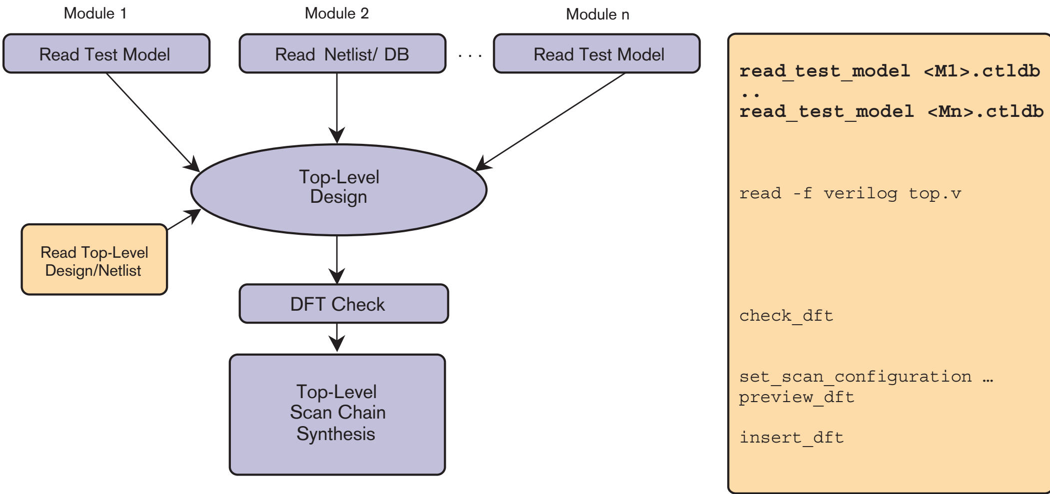


**Figure 5: Top-Level Scan Synthesis Flow with Test Models**

©2001 Synopsys, Inc.

## Interface Logic Models (ILMs) and Test Models

Interface logic models (ILMs) provide an accurate and robust model generation solution for hierarchical design flow within PrimeTime, Design Compiler and Physical Compiler. ILMs embody a structural approach to model generation, where the original gate-level netlist for a block is modeled by another gate-level netlist that contains the interface logic on a block. Interface logic contains all the circuitry leading from I/O ports to edge-triggered registers called interface registers. The clock tree leading to the interface registers is preserved in an ILM. Logic that is only contained in register-to-register paths on a block is not in an ILM [Figure 6].

ILM benefits include:

- By preserving interface logic without any modification, ILMs are highly accurate representations of the original design. ILMs do not abstract, they simply discard what is not required for modeling boundary timing. For typical designs and technologies in post-layout flows, ILMs preserve the original block timing to within ± 10ps.
- Differences in the timing characteristics of an ILM, when compared to the original netlist, are easy to debug because the interface logic is maintained. Clock and data paths are preserved without modification, cell and net names are identical, and the cause for discrepancies can easily be isolated.
- Model generation runtime using ILMs is fast because identifying the interface logic for a design is a structural operation that is performed by analyzing circuit topology.
- There are certain types of designs, for example pure combinational logic blocks or latch-based designs with many levels of time borrowing, where ILMs do not offer much reduction in model size. The interface logic for such designs tends to contain much of the original design.

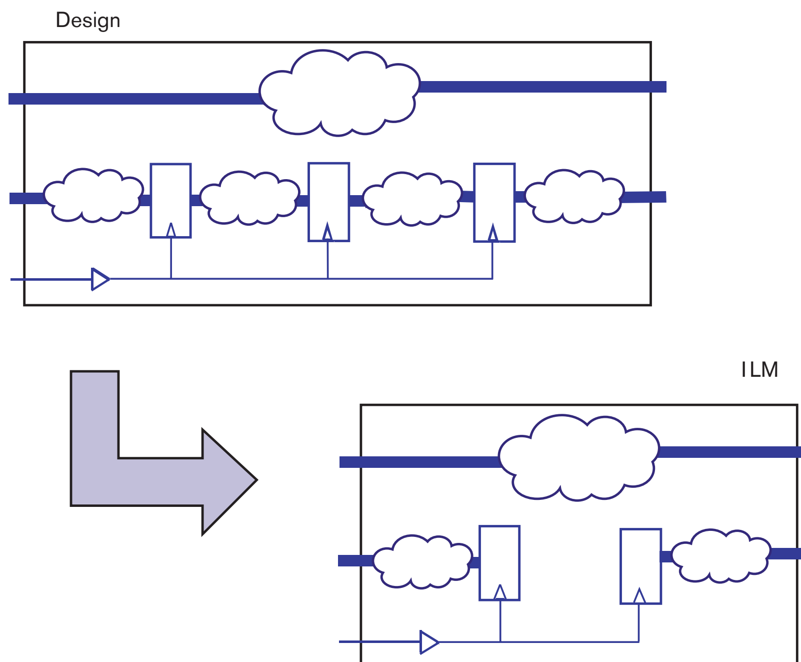


**Figure 6: Interface Logic Model (ILM)**

ILMs and test-models are now integrated within Design Compiler and Physical Compiler environments to maintain the existing hierarchical design flows for large designs and meet all requirements of timing, physical constraints and test. The ILM is extracted from the design and saved along with the test information and physical locations for the block. Figure 7 shows the integration of test models with ILMs and the process for generating ILMs with test and physical information in Physical Compiler. The sub-module goes through the typical physical synthesis and 1-Pass scan ordering process in Physical Compiler. Then the sub-module ILM can be extracted and saved as a Synopsys db format that contains the boundary timing information and physical locations for the ports of the sub-module, as well as the scan-chains and other test-related information.

At the top-level, all the ILMs (db) can be read into Physical Compiler along with the top-level floorplan. The top-level design can be a combination of hard-macros, glue logic and ILMs. The same physical synthesis design flow can be applied to the top-level design and scan chains can be ordered based on physical locations. This hierarchical approach to physical design, along with optimization with timing, significantly boosts productivity by reducing design time and iterations while enabling quick timing closure.

The same flow can be adopted within Design Compiler to extract a sub-module ILM with only timing and test information, and integrate these sub-module ILMs at the top level to perform hierarchical scan-chain synthesis that meets all timing and test constraints.

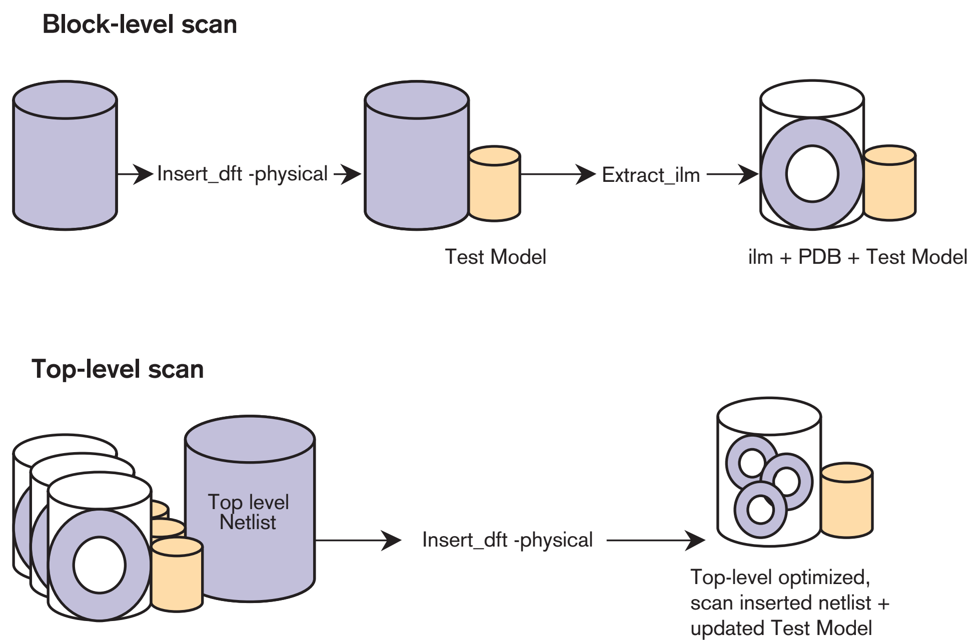


**Figure 7: Flow for Extracting ILMs and Test Models in Physical Compiler**

## Conclusion

As the design community moves to very large ASIC and SoC designs, it is mandatory that designers and system architects adopt hierarchical bottom-up scan synthesis flows to quickly meet timing and DFT closure. However, with the sub-modules also increasing in size, capacity issues and large run-times are the biggest challenge to performing scan synthesis on large blocks. The current challenge for large designs is to perform scan design rule checks at the chip level along with scan-chain architecting to achieve timing closure with minimum design iterations.

With the introduction of hierarchical scan synthesis using test models in DFT Compiler, designers can achieve significant performance over their current solution. On typical multi-million gate chip-level designs, the hierarchical scan synthesis approach with test models has achieved over a 7x improvement in run time and has reduced memory usage by 3x times [Figure 8].

With the automatic abstraction of test information into test models in DFT Compiler, the flow becomes transparent to the designer. Powerful modeling technology drives superior performance and complete automation for hierarchical scan synthesis, thereby enabling easier adoption and improved design productivity. This helps designers and system architects manage huge DFT complexity with advanced hierarchical DFT flows.

The integration of test models with ILMs provides a complete solution for hierarchical chip-level scan synthesis that meets all test and timing constraints in Design Compiler. In the Physical Compiler environment, hierarchical 1-Pass physical scan synthesis can be performed on very large designs using the ILM and test model technology to achieve test, timing and physical constraints with minimum design iterations and significant productivity enhancement.

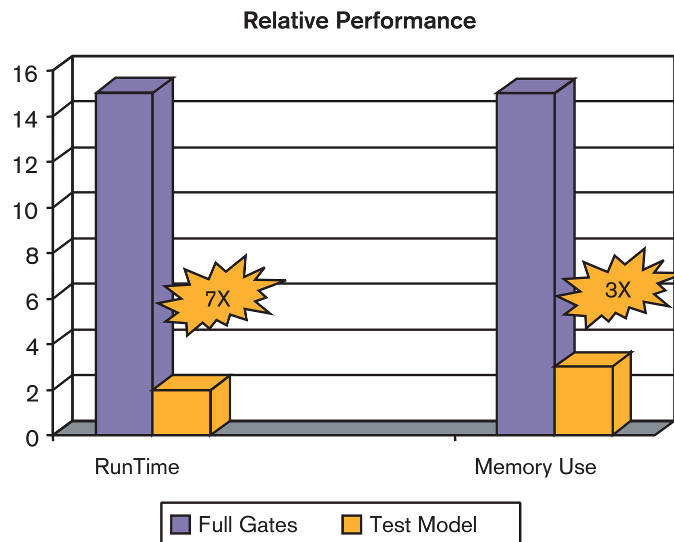


**Figure 8: Results Using Test Models in Hierarchical Scan Synthesis**

SYNOPSYS®

**700 East Middlefield Road, Mountain View, CA 94043 T 650 584 5000 www.synopsys.com**
**For product related training, call 1-800-793-3448 or visit the Web at www.synopsys.com/services**

Synopsys, the Synopsys logo, Physical Compiler and Design Compiler are registered trademarks. All other products or service names mentioned herein are trademarks of their respective holders and should be treated as such. All rights reserved. Printed in the U.S.A.
©2001 Synopsys, Inc. 10/00.PS