OCI: Open Compression Interface

Bruce Cory, Rohit Kapur, Mick Tegethoff, Mark Kassab, Brion Keller,
Kee Sup Kim, Dwayne Burek, Steve Oakland, and Benoit Nadeau-Dostie

Abstract

Before on-chip scan compression, it was possible to use different EDA tool vendors to do scan insertion, pattern generation, and diagnosis. On-chip scan compression changed that use model since each tool vendor supplied a different type of scan compression logic and had tool-specific ways to pass necessary information from scan insertion to pattern generation and from pattern generation to diagnosis. OCI (Open Compression Interface) is a standardization of how the necessary data is passed from test logic insertion to pattern generation to diagnosis such that different vendors can be used for each step independent of the on-chip scan compression logic used. This document discusses the need for OCI and gives a conceptual overview of the OCI standard.

1. Introduction

As technology shrinks, more and more logic is able to fit into the same die area. While a few design companies are shrinking the size of their die, most are combining multiple functionalities or increasing the power of the functionality available and keeping the total die size the same or even larger. This, plus the need for more types of test patterns to maintain quality of test levels at lower technology nodes, is causing the amount of test data needed to verify that the parts are manufactured correctly to skyrocket. The rate is far greater than the growth of tester memory can economically accommodate and also causes issues with the time required to test each part. The main way used to handle these problems is to use more and more complex on-chip scan compression logic. The use of on-chip scan compression logic has been steadily growing for the last several years.

Due to growth in demand for on-chip scan compression, solutions are available from most EDA test vendors. Some example products include OPMISR, TestKompress, DFT compiler Max, VirtualScan, and ELT-Comp. Some design companies have published papers with details of their in-house compression schemes. One example is X-compact. Academic papers on test compression have been published for many years with many of the concepts being used by industry. One example of this is Illinois scan. Each of these compression schemes has similar concepts but different implementation details.

Flows for scan-based test tools are broken into three main stages. Test logic insertion, pattern generation, and diagnosis. Test logic insertion does insertion and verification of test logic. Pattern generation uses the test logic to make test patterns that can be used to verify if the design is fabricated correctly. Diagnosis is used to identify the failing location in a specific device. Diagnosis information can be used to increase future yield and to solve problems that keep a design from going to market.

Most on-chip scan compression schemes are only supported by a limited number of test vendors with most only supported by one vendor. This causes many issues in the test community. Figure 1 shows one example of the issues caused by vendor-specific on-chip scan compression logic and flows.

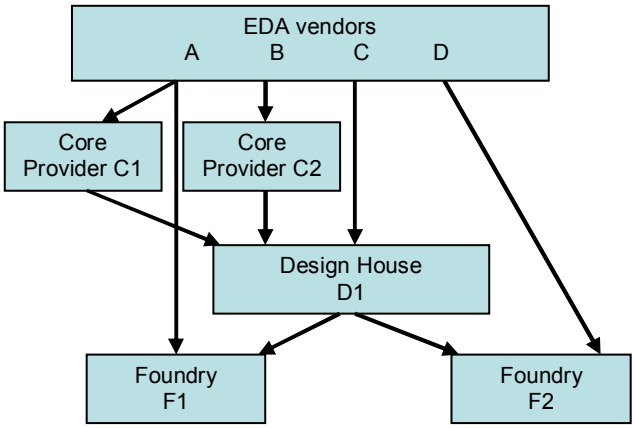

Figure 1: Example use model when on-chip scan compression is not interoperable

Companies A, B, C, and D each have different compression logic and flows only supported by their tool. Hard core provider C1 requires in-core scan compression logic using tool A while hard core provider C2 requires in-core scan compression logic using tool B. Both core developers need to own one tool to insert and verify the on-chip scan compression logic. The design house (D1) is making a design using core C1, C2, plus other in-house developed logic and uses tool C for its in-house on-chip scan compression insertion, pattern generation, and diagnosis. The design house must own three vendor tools to do pattern generation and diagnosis for the different compression structures of the design. Design house D1 then plans to produce the design using foundries F1 and F2. F1 uses tool A for diagnosis while F2 uses tool D for diagnosis. Foundries F1 and F2 need to support 3 or 4 different tools for diagnosis. The quality tradeoffs, support burden, design size overhead, and expense incurred by the design house and foundry because tool C and D can't do pattern generation and diagnosis using the

1-4244-0292-1/06/$20.00 © 2006 IEEE