IEEE P1500-Compliant Test Wrapper Design for Hierarchical Cores*

Anuja Sehgal[1] Sandeep Kumar Goel[2] Erik Jan Marinissen[2] Krishnendu Chakrabarty[1]

[1] Duke University
Department of Electrical and Computer Engineering
110 Hudson Hall, Box 90291
Durham, NC 27708, U.S.A.
as@ee.duke.edu
krish@ee.duke.edu

[2] Philips Research
Department of Digital Design & Test
Prof. Holstlaan 4 – WAY-41
5656 AA Eindhoven, The Netherlands
erik.jan.marinissen@philips.com
sandeepkumar.goel@philips.com

Abstract

Most system-on-chips (SOCs) today contain hierarchical cores that have multiple levels of design hierarchy. An efficient wrapper design for hierarchical cores is necessary to facilitate modular testing of SOCs. In most of the prior work on wrapper design for embedded cores, all the cores are assumed to have a flattened hierarchy. In this paper, we present a hierarchical core model and a generic IEEE P1500-compliant wrapper architecture for hierarchical cores. We assume that the embedded cores within the hierarchical cores are hard cores, since they are wrapped by the core vendor a priori and they have their own TAM architecture. Unlike prior wrapper design methods that assume a single test mode for hierarchical core wrappers, we present a general architecture for hierarchical core wrappers and describe various modes of operation of the wrapper. We design reconfigurable wrappers for hierarchical cores that can operate efficiently in all the test modes, thereby minimizing the overall time required to test the hierarchical core for any given TAM width. We propose a heuristic approach to solve the problem of hierarchical core wrapper design, and present experimental results for two hierarchical cores present in an ITC'02 benchmark SOC.

1 Introduction

Advances in semiconductor design and process technology have facilitated the integration of a complete system on a single chip. System-on-chips (SOCs) typically contain a heterogeneous mix of digital logic, embedded memories, and analog modules. In order to speed up the development of such large chips, their designers increasingly use pre-designed and pre-verified third-party cores such as CPUs, DSPs, media co-processors, memories, and mixed-signal cores. The design hierarchy of SOCs typically spans multiple levels. As an example, [1, 2] describe SOCs for digital video, for which the design is partitioned into chiplets, which in turn consist of cores. Also, most of the industrial SOCs in the set of ITC'02 benchmarks [3] contain multiple hierarchy levels.

Large SOCs are often tested in a modular fashion, i.e., the various SOC modules are tested as stand-alone units [2]. For embedded non-logic components, such as memories and analog modules, and also for black-boxed third-party cores, modular testing is mandatory [4]. However, a modular test approach is an attractive proposition for the remaining SOC logic also, due to the benefits of ATPG complexity containment ('divide-n-conquer') and test reuse over subsequent derivative SOC designs [5]. Modular SOC testing requires an on-chip test access infrastructure, consisting of test wrappers and test access mechanisms (TAMs). Test wrappers isolate the various modules from their surrounding circuitry during test, while TAMs transport test stimuli and responses between SOC pins and module terminals and vice versa. The design of wrappers and TAMs has a large impact on the SOC's test length, which determines the test application time as well as the required test vector storage depth on the test equipment. Recently, many papers have been published that optimize the design of test wrappers and/or TAMs [5, 6, 7, 8, 9, 10, 11]. Unfortunately, most of these papers for reasons of simplicity assume only two levels of hierarchy, i.e., SOC and core, and hence ignore the fact that most realistic SOCs have a test hierarchy of three or more levels. Support for multiple levels of hierarchy requires substantial modifications to the approaches published so far.

This paper addresses the issue of wrapper design for hierarchical SOC cores, i.e., cores which contain other cores. To the best of our knowledge, no previous in-depth studies on this topic have been published before. Our significant contributions in this paper are the following.

- We present a generic model for SOCs with a multi-level test hierarchy, and describe four different practical wrapper design scenarios that occur between two adjacent hierarchy levels.
- We identify a new category of terminals for hierarchical cores and explore the alternatives of connecting them into the TAM.
- We propose an IEEE P1500-compliant wrapper architecture for hierarchical cores which is reconfigurable into two complementary INTEST modes.
- For a frequently occurring practical design scenario of the four, we formulate for both INTEST modes the problem of wrapper design for minimal test length, and propose heuristic algorithms for both problems.

* The work of A. Sehgal and K. Chakrabarty was supported by the National Science Foundation under grants CCR-9875324 and CCR-0204077, and by the Semiconductor Research Corporation under Contract no. 2004-TJ-1174.

Our wrapper design approach can serve as an integral procedure of a still-to-be-developed TAM-wrapper co-optimization technique for hierarchical SOCs.

The sequel of this paper is organized as follows. In Section 2, we review prior work on core test wrappers and hierarchical SOCs. In Section 3, we present a generic hierarchical core model. Section 4 describes the proposed wrapper architecture for hierarchical cores and outlines the various components and modes of the wrapper. In Section 5 and Section 6, we formulate the problems of wrapper design in the two complementary INTEST modes and propose heuristic approaches to design efficient wrappers that minimize the overall test length of the parent core. Finally, in Section 7, we present experimental results for two benchmark parent cores from the ITC'02 benchmark suite [3]. Section 8 concludes the paper.

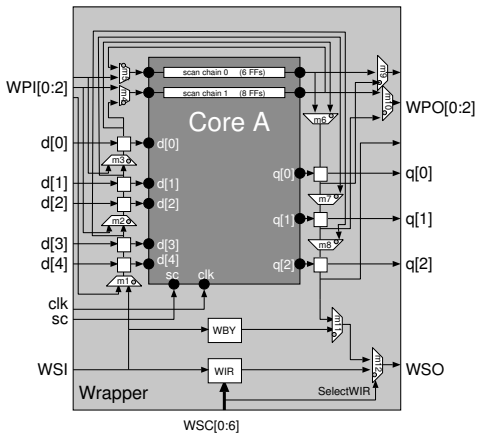

Figure 1. An example of an IEEE P1500 Core Test Wrapper [15].

2 Review of Prior Work

A considerable amount of research has been done in test wrapper architectures and design techniques for non-hierarchical SOCs. Marinissen et al. [12] proposed a core test wrapper named TestShell, and Varma and Bhatia [13] described a very similar wrapper called TestCollar. The IEEE P1500 Standard for Embedded Core Test standardizes a core test wrapper [14] that is very similar to the TestShell and TestCollar. All of these wrappers have a scalable width TAM, and the TestShell and the IEEE P1500 wrapper support both TestBus and TestRail TAM architecture. The IEEE P1500 core test wrapper supports the normal functional mode, the INTEST mode, the EXTEST mode, and the Bypass mode. Figure 1 shows an example of an IEEE P1500 wrapper. It has Wrapper Boundary Cells and a Wrapper Instruction Register (WIR). It has a single-bit TAM with called wrapper serial input (WSI) that is capable of transporting both test stimuli and test control signals; It can be used to program the WIR. It addition to WSI, the IEEE P1500 wrapper can support an optional multi-bit TAM called wrapper parallel input (WPI), which can be used to transport test data only. The P1500 wrapper also allows unequal number of inputs and outputs. Also, as seen in Figure 1, the P1500 wrapper has the feature of wrapper-wide bypass (WBY) that allows the bypass of an entire core in one clock cycle. In addition to the above described work on non-hierarchical wrapper architectures, recently, Koranne [6] proposed reconfigurable wrappers for non-hierarchical cores that can adapt to a TAM width dynamically during test scheduling.

Heuristic approaches for efficient wrapper design in non-hierarchical cores were presented in [7, 8]. It was shown in [7] that there exists a relationship between test length of a core with its scan-in and scan-out times, through wrapper cells and core-internal scan chains. The test time T of a core can be defined as

$$T = \{1 + \max(s_i, s_o)\}.p + \min(s_i, s_o),$$

where p denotes the number of test patterns, and s_i and s_o denote respectively the maximum scan-in and scan-out time for a core. In [7], it was also shown that the wrappers should be designed such that the maximum scan-in and scan-out times of the core are minimized, so that the overall test length of the module is minimized. Scan chains and wrapper cells are referred to as TAM items in [7]; the number of TAM items is often greater than the number of available TAM wires for testing. It was shown in this paper that it is necessary to partition the set of TAM items into a number of subsets equal to the number of available TAM chains. The partitioning of TAM items over TAM wires determines the scan-in time s_i and scan-out time s_o for the core, and hence determines its test length. In [7], it was shown that for hard cores, where scan chain redesign is not possible, the problem of calculating s_i and s_o is equivalent to the well-known $\mathcal{N}P$-hard problems of Bin Design and Multi-Processor Scheduling. It was also shown that the Least Processing Time (LPT), Best Fit Decreasing (BFD) and COMBINE heuristics can be used to solve the problem. The BFD heuristic approach was used in [8] to solve the wrapper design problem.

Although a significant amount of research has been carried out on wrapper design and TAM optimization for non-hierarchical cores [5, 8, 9, 10, 11], comparatively little has been published on hierarchical cores. Only limited work has been done on test architecture design for hierarchical cores [16, 17, 18]; however, the focus has not been on optimizing wrappers or TAM architectures in these papers. While these papers have addressed the design of hierarchical TAM architectures, they have not focussed on optimizing the test architectures and presenting results for industrial benchmarks. In [19], TAM optimization techniques for non-hierarchical SOCs have been used to optimize multi-level TAM architectures. However the constraints on the modes of operation of the hierarchical cores have been neglected in this work.

Recently, in [20], Goel proposes a wrapper architecture for hierarchical cores that allows a core to operate in INTEST and EXTEST mode concurrently, thereby allowing parallel testing of

cores at different levels of hierarchy. However, the wrapper cells used in this approach have a high area overhead, compared to the conventional IEEE P1500 wrapper cells, and may not be feasible for all applications. In our approach we address the problem of designing a IEEE P1500-compliant wrapper for hierarchical cores. A solution to this problem will make the the IEEE P1500 standard more useful for today's SOCs.

3 Hierarchical Core Model

In this section, we present a generalized hierarchical core model that can be adapted to a wide range of industrial designs. We describe the various components of a hierarchical core and enumerate the different design scenarios that may exist in practice.

Hierarchical cores may have multiple levels of test hierarchy. They have embedded cores, which in turn may have their own embedded cores in deeper levels of hierarchy. In order to express test hierarchy, we use the convention of naming levels based on a depth-first traversal of the test hierarchy tree. The levels correspond to the depth in the test hierarchy tree. The top-level of hierarchy is the SOC itself, which is on Level 0. A core embedded in a core of Level n is on Level $n+1$. The core on Level n is called a *parent* core with respect to the cores on Level $n+1$, and conversely the cores on Level $n+1$ are called *child* cores with respect to the core on Level n. Parent cores may have multiple child cores, which in turn may be parent cores to cores on a deeper level. Hence, the hierarchical core model is a recursive model. Figure 2(a) illustrates an example of an hierarchical SOC, and Figure 2(b) illustrates its test hierarchy tree. Every wrapped parent core has an *external TAM* and an *internal TAM*. The TAM that connects externally to the parent core is called its external TAM, and the TAM architecture of its child cores is called its internal TAM. In Figure 2, Core C and Core D are hierarchical cores, hence they have internal and external TAMs. Core C has three levels of hierarchy, it has Child Cores D, E and F, and Child Core D has its own Child Core G.

We distinguish between the problems of wrapper design for cores with *hard* and *soft* implementations of their scan chains, wrapper and/or TAM architecture. A hard implementation is one which is fixed *a priori* to the overall test architecture design and implementation. On the other hand, a soft implementation is not fixed and lends itself to design changes and optimization. Soft implementation of scan chains, wrappers, and TAM architecture are more suitable for optimization, since the design of test architecture can be adapted for a globally optimal test length. In [7], it was shown that the problem of wrapper design for a core with fixed scan chains is $\mathcal{NP}$-hard; this is infact true for most optimization problems involving a hard implementation of a design. On the other hand, the problem of optimizing a soft implementation is simpler [7] since it can be redesigned in accordance to the available resources.

Based on the hard or soft implementation of scan chains, wrapper and TAM architecture, a parent or child core can be classified as hard, wrapped and TAMed, or hard, unwrapped and not TAMed, etc. A TAM architecture for a core can be implemented only after the cores have been wrapped, and a core can be wrapped only after the scan chains have been designed. Hence, there is a design flow in which the scan chains have to be designed before the wrapper, and the wrapper has to be designed before the TAM architecture. A soft implementation of a design cannot precede a hard implementation of a design in the design flow. For example, it is not possible to have a fixed TAM architecture for an unwrapped core. There are four feasible combinations of hard and soft implementations that are possible; these are enumerated in Table 1. Both child cores and parent cores can have any of these combinations of hard and soft implementations. However, the child cores have to be wrapped and TAMed before the parent core can be wrapped or TAMed. Thus, the child cores can be be soft only if the parent cores have a soft implementation of their wrapper and external TAM. The parent core scan chains can be soft as long as the parent core has not been wrapped, regardless of the child core architecture.

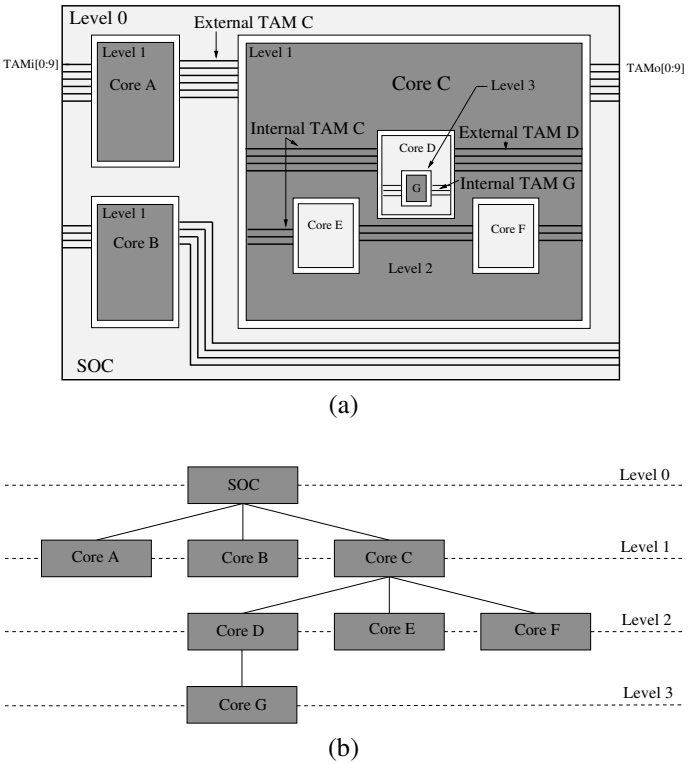

Figure 2. (a) An example of a hierarchical SOC, (b) test hierarchy tree for the SOC.

In this paper, we solve the problem of wrapper design for a parent core in Level n, which has one or more child cores in Level $n+1$. Since the hierarchical core model is a recursive model, it is possible to extend the technique to any number of levels. The technique has to be applied in a bottom-up manner, starting from the child cores in the deepest level of the test hierarchy tree

Case	Scan Chains	Wrapper	TAM Architecture
1	hard	hard	hard
2	hard	hard	soft
3	hard	soft	soft
4	soft	soft	soft

Table 1. Combinations of hard and soft implementation of child core scan chains, wrapper and TAM architecture.

In this paper, we consider the case in which the child cores have fixed scan chains, are wrapped, and TAMed, and the parent cores have hard scan chains, soft wrappers, and a soft TAM architecture. The problem of optimizing the parent core wrappers for test length is more challenging for this case. This is because the the child cores have a fixed TAM architecture, which cannot be optimized through redesign.

4 Wrapper Architecture

This section describes core terminals of a hierarchical core and the architecture of the hierarchical core wrapper. It also details the various test modes of the wrapper.

Based on test access requirements, core terminals for non-hierarchical cores are classified as: functional-only, test data, and control terminals [7]. The functional only terminals include functional inputs and outputs that require wrapper cells to apply the test stimuli and store the responses. The test-data terminals are the terminals directly connected to the scan chains of the core, hence they do not require wrapper cells. Control terminals are used to apply control signals to the various components of the wrapper. A parent core, in addition to the terminals mentioned above, has terminals that provide test access to the wrapped child cores; we have named these terminals CTAM terminals. CTAM terminals are inputs or outputs at the parent core level that are directly connected to the TAM wires of the child core TAM architecture. Thus, in a wrapped hierarchical core, they connect to both the internal TAM and external TAM of a parent core. Every wire of the internal TAM is termed *CTAM chain*. Unlike the functional terminals, CTAM terminals do not require wrapper cells at the parent core wrapper since the child cores are wrapped, and have their own wrapper cells at their terminals. These terminals also differ from the test data terminals, since they operate in both INTEST and EXTEST modes. However, it should be noted that with the conventional wrapper cells used in our architecture, the INTEST and EXTEST mode for a core have to be time-multiplexed. Figure 3 illustrates an example of an unwrapped hierarchical core. In this example, the *SI* and *SO* terminals are the test data inputs and outputs respectively for the parent core scan chains, the *FI* and *FO* terminals are the functional inputs and outputs of the parent core, and similarly the *CTAMI* and *CTAMO* terminals are the CTAM inputs and outputs for the child cores.

The parent core wrapper architecture shares several components in common with the IEEE P1500 wrapper architecture described for flat cores in [14]. TAM ports (WPI and WSI), wrapper cells, WBY and WIR, as described in [14], have similar functionality in the parent core wrapper. However, additional components have been added to facilitate efficient test access to child cores embedded in the parent core. Also, the test modes of the parent core wrapper differ significantly from the test modes of the non-hierarchical core wrappers.

In the parent core, we have identified an EXTEST mode and two INTEST modes. The parent core EXTEST mode is similar to the EXTEST mode of the non-hierarchical cores. In this mode, the parent core terminals are used for core-external testing. The functional wrapper cells are used in this testing, and the parent core scan chains can be bypassed. The child cores can be in INTEST or EXTEST mode, since they do not participate in the parent core wrapper functionality, and are internal to the parent core. The two INTEST modes that we have identified in a parent core are significantly different from the INTEST mode of a non-hierarchical SOC.

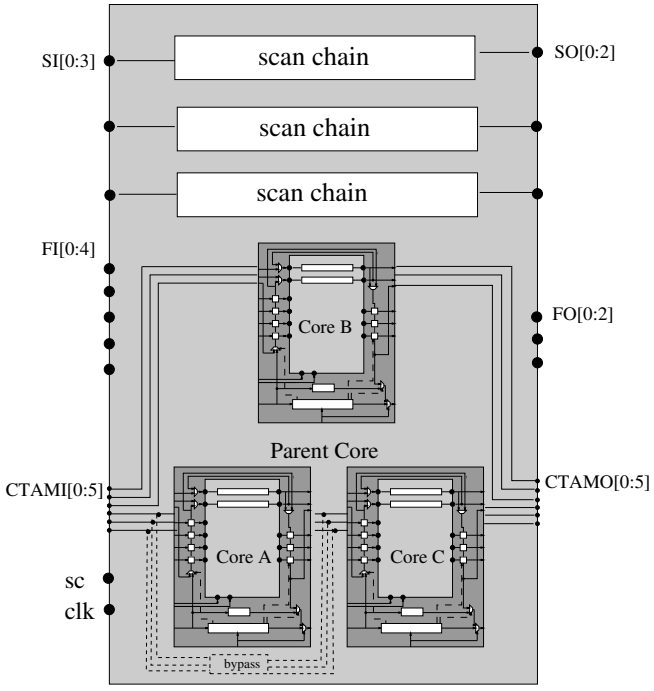

Figure 3. Example of a wrapper design for a parent core with two child cores.

1. **Parent INTEST mode** (INTEST_P): In this mode, parent core internal testing is done. Test data is scanned through the parent core scan chains, the parent core wrapper cells and the child core wrapper cells. However, the child cores have to be in the EXTEST mode to ensure complete core-internal testing of the parent core. As a result, test data has to be scanned through both the parent core and the child cores. Hence, the available TAM wires have be distributed between both the parent core scan chains and wrapper

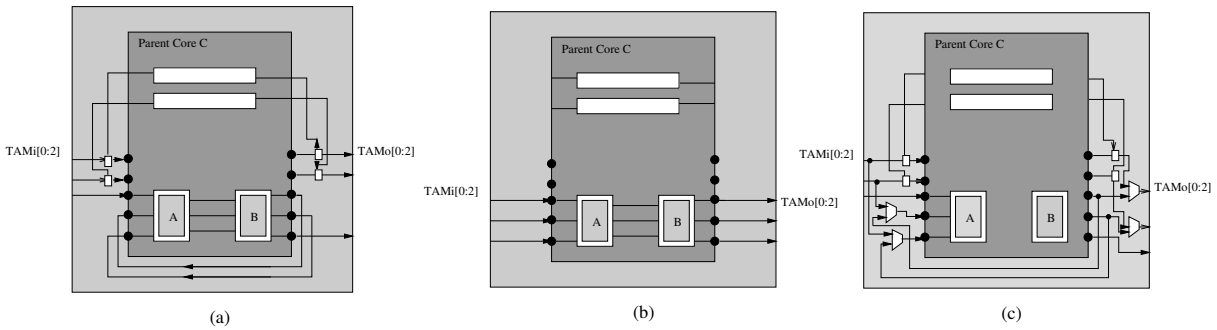

Figure 4. (a) Parent core wrapper in Parent INTEST mode; (b) Parent core wrapper in Child INTEST mode; (c) Unified parent core wrapper.

cells, as well as the child core TAM architecture. This was ignored in the prior work done on TAM optimization for hierarchical cores in [19]. For example, all the TAM wires were devoted to parent core testing during the INTEST mode of the hierarchical core.

2. **Child INTEST mode** (INTEST$_C$): In this mode, child core internal testing is done; all child cores are in INTEST mode. The parent core wrapper elements can be in any mode of operation since the TAM inputs will be able to transport data to the child core terminals regardless of the mode of operation of the parent core itself. Thus, in this mode all the TAM wires can be utilized by the child cores for their INTEST testing.

It should be noted that the two INTEST modes described above have to be time-multiplexed, since both the modes require the TAM input terminals to be active. In this paper, we focus on designing a wrapper that is efficient, in terms of test length, for both the INTEST modes. However, since the two modes are time-multiplexed, the top-level TAM wires available for the overall parent core testing at the parent core wrapper interface can be used in both modes. Multiplexers can be used to route the TAM wires to the child cores as well as the parent core scan chains and wrapper cells. These multiplexers can be controlled by control inputs that select the appropriate inputs depending on the mode of operation. Now, the problem of wrapper design can be considered as a two-fold problem. Two wrappers can be designed independently for the two INTEST modes, and can be merged using multiplexers. We elaborate on this with the help of an example.

Figure 4(a) shows an example of a parent core wrapper configuration when the core is in INTEST$_P$ mode. In this wrapper, the available TAM wires are used to access the parent core scan chains, parent core wrapper cells, and the child core CTAM inputs. Figure 4(b) shows the wrapper configuration for the same parent core in INTEST$_C$ mode. In this configuration, the available TAM wires can be distributed between the CTAM inputs alone. Figure 4(c) shows how the two wrapper configurations can be merged to form one wrapper, using multiplexers. The multiplexer select bits can be chosen appropriately to access the CTAM chains alone in the INTEST$_C$,or they can be chosen such that the TAM items participating in INTEST$_P$ mode are selected.

An alternative approach to using the above described merging technique, is to partition the available TAM width into two dedicated TAM partitions for the parent core and child core TAM architecture. In the parent INTEST mode, both the parent TAM partition and child TAM partition can be used, and in the child INTEST mode, only the child TAM partition can be used. As a result, the TAM width available for child core testing in the $InTest_C$ mode is smaller than that available in our proposed approach.

In addition to the wrapper features described so far, optional bypasses can also be implemented that can be used in any of the three modes. In [15], two types of bypasses have been defined. The wrapper-wide bypass allow the bypass of an entire core. Single registers are used as bypasses to avoid delay effects, thus it takes one clock cycle to bypass the core; Figure 3 shows Core A equipped with a wrapper-wide bypass. A wrapper-wide bypass can exist outside or inside the wrapper. Also, scan chain bypasses can be available that allow the scan chains of a core to be bypassed. Since the child cores are wrapped by the core provider, the use of bypasses in child cores would be left to the core provider's choice. In our experimental results, we also study the impact of bypasses on the test time of the wrapped hierarchical cores.

In Section 5 and Section 6, we present the wrapper design problems for each of the modes in more detail, and propose heuristic algorithms to solve the problems.

5 INTEST$_P$ Mode Wrapper Optimization

In Section 2 we explained that the scan-in and scan-out times of a core should be minimized in order to optimize the overall test length of a core. For a hierarchical core, the *TAM items* that participate in the scanning in and out of test patterns are (1) parent core wrapper input cells, (2) parent core scan chains, (3) parent core wrapper output cells, and (4) child core wrapper cells. The child core wrappers cells, together with child core scan chains, form CTAM chains. Two or more CTAM chains can be daisy-chained to form *TAM chains*. The sum of the number of wrapper cells and the length of the scan chains in a CTAM (TAM) chain

is referred to as the *scan-length* of the CTAM (TAM) chain. The scan-lengths of child cores in INTEST$_P$ mode can be reduced if scan-chain bypasses are enabled for child cores.

Prior to wrapper design, for a parent core, the information about the number of scan chains and their lengths at the parent core level is provided by the core provider. The number of wrapper input and output cells is equal to the number of functional inputs and outputs respectively. Also, information about the scan-lengths, scan-in and scan-out times of the child cores can be obtained from the core provider in the form of *test protocols*. Test protocols, provided for each child core by the core provider, carry information about the test stimuli and the scan times of the child core [22]. Thus, given the scan chain lengths, number of input and output wrapper cells at the parent core level, and the scan-lengths of the child cores, we have all the information required to determine the maximum scan-in and scan-out times of the parent core in INTEST$_P$ mode configuration. The scan-length of each CTAM chain can be computed from the given information. In the INTEST$_P$ mode we have used the total scan-lengths of the CTAM chains, instead of the scan-in and scan-out lengths, to calculate the overall test time of the core. This assumption may increase the overall test times of the core negligibly, however it reduces the complexity of the problem. Thus, with this information we can proceed to define and solve the wrapper design problem for parent cores in the INTEST$_P$ mode configuration.

In many practical cases, the number of TAM items is much larger than the number of available external TAM wires. In such cases, the set of TAM items has to be partitioned into a number of subsets equal to the number of available TAM wires. The partitions should be made such that the maximum scan-in and scan-out times of the parent core are minimized. The wrapper design problem for the parent core in the INTEST$_P$ mode can now be formalized as a partitioning problem as follows.

Problem 1 [Wrapper design in INTEST$_P$ mode]
Given:

1. A set $\mathcal{WI} = \{WI_1, WI_2, \ldots, WI_x\}$ of wrapper input cells, each wrapper input cell having length $l(WI_i) = 1$;
2. A set $\mathcal{S} = S_1, S_2, \ldots, S_y$ of parent core internal scan chains, where scan chain S_i has length $l(S_i)$;
3. A set $WO = \{WO_1, WO_2, \ldots, WO_z\}$ of wrapper output cells, each wrapper cell having a length $l(WO_i) = 1$;
4. A set $\mathcal{S}_c = \{S_{c,1}, S_{c,2}, \ldots, S_{c,v}\}$ of CTAM scan-lengths, each scan-length has a length $l(S_{c,i})$;
5. A set of w TAM wires.

Define:

1. For any $X \subset \mathcal{WI} \cup \mathcal{S} \cup \mathcal{WO} \cup \mathcal{S}_c$, $l(X) = \sum_{x \in X} l(x)$;
2. TAM partition $\mathcal{P} = \{P_1, P_2, \ldots, P_w\}$ of $\mathcal{WI} \cup \mathcal{S} \cup \mathcal{WO} \cup \mathcal{S}_c$ into w disjoint sets, one for each TAM wire;
3. Input set $IN_i = P_i \backslash \mathcal{WO}$;
4. Output set $OUT_i = P_i \backslash \mathcal{WI}$;
5. Scan-in length for TAM partition $\mathcal{P}$, $si(\mathcal{P}) = \max_{1 \geq i \geq w} l(IN_i)$;
6. Scan-out length for TAM partition $\mathcal{P}$, $so(\mathcal{P}) = \max_{1 \geq i \geq w} l(OUT_i)$.

Find a TAM partition $\mathcal{P}^\star$, having w subsets, such that the overall test length of the core is minimized, i.e., $\mathcal{P}^\star$ satisfies $\max(si(\mathcal{P}^\star), so(\mathcal{P}^\star)) \geq \max(si(\mathcal{P}), so(\mathcal{P}))$ for all partitions $\mathcal{P}$ of $\mathcal{WI} \cup \mathcal{S} \cup \mathcal{WO} \cup \mathcal{S}_c$. □

The above problem is similar to, but more general than, the *partitioning of TAM chain items* (PTI) problem described in [7]. TAM chains described in [7] are subsets of the set of parent TAM items, since they do not include child core scan lengths. The PTI problem has been shown to be $\mathcal{NP}$-hard in [7]. Thus Problem 1, as described above, is also an $\mathcal{NP}$-hard problem.

We use a three-step approach, similar to that described in [7], to solve Problem 1.

1. Assign the parent core internal scan chains $\mathcal{S}$ and the child core scan-lengths $\mathcal{S}_c$ to TAM chains, such that the maximum sum of scan-lengths assigned to a TAM chain is minimized. The resulting partition is named $\mathcal{P}_S$.
2. Assign the wrapper input cells in $\mathcal{WI}$ to TAM chains on top of $\mathcal{P}_S$, such that the maximum scan-in times of all the TAM chains is minimized.
3. Assign the wrapper output cells in $\mathcal{WO}$ to TAM chains on top of $\mathcal{P}_S$, such that the maximum scan-out times of all the TAM chains is minimized.

Step 1 described above can be formalized as the *Partitioning of Scan Chains* (PSC) problem as described in [7] and can be solved using the Largest Processing Time (LPT) algorithm as described in [7].

6 INTEST$_C$ Mode Wrapper Optimization

Next, we proceed to define the wrapper design problem for the hierarchical core in INTEST$_C$ mode. In this mode of operation, the test stimuli have to be transported to the child cores only, hence all the TAM wires available at the parent core wrapper interface can be utilized for child core testing.

Problem 2 [Wrapper design in INTEST$_C$ mode]
Given a set of CTAM chains M and child cores C, and for each child core $c \in C$, the number of test patterns p_c, total scan-length, scan-in, and scan-out times $sl_{c,k}$, $si_{c,k}$ and $so_{c,k}$ respectively on k chains ($k \in M$). Furthermore we are given a number w that represents the maximum number of parent core level

TAM wires available for testing. Determine a wrapper design for the parent core, such that the overall test length (in clock cycles) required to test all the child cores is minimized and w is not exceeded. □

The total test length of testing all the child cores depends on the number of available TAM wires and the child core TAM architecture. Let us consider two cases: (1) the number of available TAM wires is greater than or equal to the number of CTAM chains; (2) the number of available TAM wires is less than the number of CTAM chains.

Case I : $w \geq \mid M \mid$.

The test time taken to test Core i, if the number of available TAM wires w is *equal to* the number of CTAM chains $\mid M \mid$, is given by:

$$T_i = \{1 + \max(si_i, so_i)\} \cdot p_i + \min(si_i, so_i). \quad (1)$$

In Equation 1, si_i and so_i are the maximum scan-in and scan-out lengths of Core i respectively, defined as

$$si_i = \max_k(si_{i,k}), \quad 1 \leq k \leq \mid M \mid.$$
$$so_i = \max_k(so_{i,k}), \quad 1 \leq k \leq \mid M \mid.$$

The total test time taken to test all the $\mid C \mid$ cores on $w \geq \mid M \mid$ TAM wires is the maximum of the testing time on any of the CTAM chains. Let y_{ij} be a binary variable such that,

$$y_{ij} = \begin{cases} 1, & \text{if a test for Core } i \text{ involves CTAM chain } j \\ 0, & \text{otherwise} \end{cases}$$

Core i involves CTAM j, if $si_{i,k} + so_{i,k} + sl_{i,k} \neq 0$. Now, the total testing time of the child cores can be expressed as:

$$T(w) = \max_j \sum_i T_i \cdot y_{ij}, \quad 1 \leq i \leq \mid C \mid, \quad and \quad 1 \leq j \leq \mid M \mid. \quad (2)$$

Case II: $w < \mid M \mid$.
If the number of available TAM wires w is less than the number of CTAM chains $\mid M \mid$, then the available TAM wires have to be distributed among the CTAM chains, such that the overall test time of the child cores is minimized. Two or more CTAM chains can be daisy-chained to form TAM chains that share the same TAM wire. However, the scan lengths of the cores are subject to change depending on the daisy-chaining of the TAM chains. The scan-in and scan-out times of a core can now be defined as follows.

Let two CTAM chains t_1 and t_2 be daisy-chained to form TAM chain t. Let P_{t_1} and P_{t_2} be the set of cores on CTAM terminal t_1 and t_2 respectively. Assuming that t_1 precedes t_2, the scan-in and scan-out times of a Core i on t_2 can now be defined for TAM chain t as follows:

$$\begin{aligned} si_{i,t} &= \sum_{j \in P_{t_1}} sl_{j,t_1} + si_{i,t_2}. \\ so_{i,t} &= so_{i,t_2}. \end{aligned} \quad (3)$$

On the other hand, if t_2 were to precede t_1, the scan-in and scan-out times of a core on t_2 can now be defined for TAM chain t as follows:

$$\begin{aligned} si_{i,t} &= si_{i,t_2}. \\ so_{i,t} &= \sum_{j \in P_{t_2}} sl_{j,t_1} + so_{i,t_2}. \end{aligned} \quad (4)$$

It should be noted that a core can have its TAM items connected to both t_1 and t_2, in which case the maximum of scan-in and scan-out times obtained from the above expressions is chosen as the scan-in and scan-out times of the core on t.

The scan-in and scan-out times of every core are calculated as the maximum of the scan-in and scan-out times on every TAM chain respectively. The number of TAM chains formed by daisy-chaining CTAM chains is equal to the number of available TAM wires w. Hence the scan-in and scan-out times of the cores can be determined by taking the maximum of their scan times on the w TAM chains. The scan-in and scan-out times of a Core i are now expressed as

$$\begin{aligned} si_i &= \max_j(si_{i,j}), \quad 1 \leq j \leq w. \\ so_i &= \max_j(so_{i,j}), \quad 1 \leq j \leq w. \end{aligned} \quad (5)$$

The overall test time for testing the child cores in INTEST$_C$ mode can be determined using Equations (1) and (2).

Next, we propose a heuristic approach to arrive at a distribution of the available TAM wires between the CTAM terminal pairs in INTEST$_C$ mode, such that the overall test time of child cores is minimized. Procedure *corewrap*(w, M, C), as presented in Figure 5, describes the various steps of the proposed heuristic approach.

We define an operator ϕ, which determines the test time of all the cores connected to a CTAM terminal or to a TAM chain formed from daisy-chaining two or more CTAM chains. Let TC_i be a set of CTAM chains daisy-chained together, and let z_{cx} be the logical OR of $y_{cx}, \forall x \in TC_i$. The function $\phi(TC_i)$ is defined as

$$\phi(TC_i) = (\sum_{c \in C} T_c \cdot z_{cx}). \quad (6)$$

Procedure *corewrap*(w, M, C)

1. **sort** CTAM chains such that
   $\phi(t_1) \leq \phi(t_2) \leq \cdots \leq \phi(t_{|M|})$;
2. **for** $i := 1$ **to** $\mid M \mid$ **do** TC_i:=t_i **od**;
3. **for** $i := w + 1$ **to** $\mid M \mid$ {merge extra TAM chains}
4. **do** select $k \in \{j \mid \phi(TC_j) = \min_{1 \leq x \leq |M|} \phi(TC_x)\}$;
   {find smallest time TAM chain}
5. **for** $n := 1$ **to** $|M|$
   {find best daisy-chain candidate}
6. **do if** $n \neq k$
7. **do** $TC_{temp} := TC_k \cup TC_n$;
8. **if** $(\phi(TC_{temp}) < \phi(TC_{best}))$
9. **do** $best := n$;
8. **od**;
9. **od**;
10. **od**;
11. $TC_k := TC_k \cup TC_{best}$;
12. $TC_{best} := \emptyset$;
13. **od**;
14. **return** $\{TC_1, \ldots, TC_{|M|}\}$;

Figure 5. Pseudocode for procedure *corewrap(w,M,C)*.

In procedure *corewrap*(w, M, C), if the number of TAM wires is less than the number of CTAM chains, then starting with the smallest TAM chains, in terms of testing time, the CTAM chains are daisy-chained. Initially, all CTAM chains are considered to be TAM chains (Line 2). Then the TAM chains are selected and daisy-chained in an iterative manner until the number of TAM chains is equal to the number of available TAM wires. The *for* loop on Line 3 is the main loop in which the TAM chain with minimum testing time is daisy-chained with another TAM chain. For every TAM chain selected in Line 4, all the remaining TAM chains are considered as a potential daisy-chain candidates. After evaluating the daisy-chaining of every potential candidate with the selected TAM chain, the TAM chain that yields the smallest testing time is selected. The *for* loop on Line 5 searches for the best daisy-chain mate for the TAM chain selected on Line 5. This iterative procedure continues until $|M| - w$ TAM chains have been daisy-chained with other TAM chains.

Wrapper-wide bypasses in child cores can help reduce test time in the INTEST$_C$ mode. If wrapper-wide bypasses are present, the daisy-chaining of TAM chains does not increase the scan-in and scan-out times of the cores. In this case, when a child core is being tested on a particular TAM chain, the wrapper-bypasses of all other cores on that TAM chain can be activated. As a result, the test stimuli for the core under test do not have to be scanned through other cores, this can minimize the test time of the core. However, wrapper-bypasses cannot be activated during INTEST$_P$ mode, since the wrapper cells of all cores have to participate in the testing of the parent core. In the INTEST$_P$ mode it is advantageous for the child cores to have scan-chain bypasses, since this will minimize the scan-lengths of the cores.

Although the proposed algorithm is a greedy algorithm, we employ an enumerative approach in merging two TAM chains. Hence the complexity of this algorithm is O(MN^2), where M is the number of available TAM wires and N is the number of CTAM chains. Since M and N are not very large in practice, the heuristic has reasonable execution times (less than 10 secs on a SunW, Ultra-5_10).

7 Experimental Results

In this section, we present two case studies for the test wrapper techniques discussed for the two INTEST modes in Sections 5 and 6. A direct comparison with competing approaches is difficult due to the lack of prior work on wrapper design for hierarchical cores. We present experimental results for two hierarchical cores from the ITC'02 test benchmark SOC p22810 [3]. The details of the two hierarchical cores, along with their child core TAM architecture are presented in Tables 2 and 3.

In this paper we have assumed that the child cores have a hard implementation of their scan chains, wrappers and TAM architecture. Thus, we use the TAM optimization tool TR-ARCHITECT [5] to create a TAM architecture for the child cores. We assume that the TAM architecture for the child core has 15 bit-wide internal TAM in Parent Core 1, and a 16 bit-wide internal TAM in Parent Core 4. Given an internal TAM width, the scan-chain lengths, input and output terminals, and number of test patterns of each of the child cores, TR-ARCHITECT designs a wrapper for each child core, and it builds a TAM architecture which is optimized for the overall test length of the child cores. From the results, it is possible to determine the scan-in, scan-out and scan-lengths of the cores on each CTAM chain. In Tables 2 and 3, only the maximum scan-in, scan-out and total scan-lengths of the child cores are presented for brevity.

Hierarchical Core 1 (p22810)			
Parent core			
number of scan chains	10		
scan-chain lengths	130, 111, 111, 110, 110, 110, 110, 110, 110, 110		
number of inputs	28		
number of outputs	56		
number of test patterns	785		
Child core TAM Architecture (CTAM chains= 15)			
Child-core	# test patterns	CTAMs used	scan-times†
2	12324	1-12	(4,3,7)
3	3108	13-15	(16,22,38)
4	222	13-15	(13,9,22)

† : scan-times are presented as (max scan-in, max scan-out, max scan-length)

Table 2. SOC core parameters for Hierarchical Core 1 of SOC p22810.

Hierarchical Core 4 (p22810)	
Parent core	
number of scan chains	29
scan-chain lengths	214,106,106,105 105,103 102,101,101,101,100,93,92 84,84,75,75,73,73,73,73,27 27,27,27,27,27,27,27
number of inputs	112
number of outputs	112
number of test patterns	202

Child core TAM Architecture (CTAM chains = 16)			
Child-core	# test patterns	CTAMs used	scan-times†
5	712	1-12	(7,6,13)
6	26348	1-12	(7,6,13)
7	2628	13-16	(9,4,13)

† : scan-times are presented as (max scan-in, max scan-out, max scan-length)

Table 3. SOC core parameters for Hierarchical Core 4 of SOC p22810.

TAM wire	Input wrapper cells	Scan chains	Output wrapper cells
TAM[0]		{7,110}	
TAM[1]	{1,1,1,1,1,1,1,1,1}	{7,110}	{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}
TAM[2]	{1,1,1,1,1,1,1,1,1}	{7,110}	{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}
TAM[3]	{1,1,1,1,1,1,1,1}	{7,110}	{1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}
TAM[4]	{1,1}	{13,110}	{1,1,1,1,1,1,1,1,1}
TAM[5]		{21,111}	
TAM[6]		{110,111}	
TAM[7]		{6,110,130}	

Table 4. Partitions of TAM chains items in INTEST$_P$ mode for p22810 Core 1 ($w = 8$).

We first performed wrapper optimization for Parent Core 1 for several TAM widths. For the INTEST$_P$ mode, the set of TAM items of the parent core is partitioned into subsets, such that the number of subsets is equal to the number of available external TAM wires w. In INTEST$_C$ mode, the CTAM chains are daisy-chained such that the resulting TAM chains are equal in number to the number of available TAM wires. Table 4 shows the partitioning of the TAM chain items of Core 1 in the INTEST$_P$ mode for a parent core level TAM width of 8 ($w = 8$). It can be seen that the partitions are made such that the scan-in and scan-out times are mostly balanced over the different partitions, and the overall test length is minimized. The total test length obtained in INTEST$_P$ mode in this case was 194141 clock cycles.

In the INTEST$_C$ mode, the number of CTAM chains exceeds the number of available TAM wires by seven, hence seven CTAM chains have to be daisy-chained with other CTAM chains. The proposed heuristic gives the following result: TAM chains (1,2,13,14,15), (3,12) and (4,5,6) are daisy-chained, while TAM chains 7, 8, 9, 10 and 11 remain as is. It can be seen from Table 2 that the CTAM chains that have the largest core in terms of test length is given more dedicated TAM width than the smaller cores. The total test length obtained in the INTEST$_C$ mode in this case is 868046 clock cycles; the child cores are assumed to have no bypasses in this case.

Hierarchical Core 1 (p22810)			
w	INTEST$_P$ test time†	INTEST$_C$ test time†	
		with bypass	without bypass
2	564333	1020275	1140625
4	326189	789419	868078
6	239729	653855	868060
8	194141	481319	868046
10	155627	308783	868032
12	150125	136240	855707
14	108467	136240	855707
16	108467	74617	74617

Hierarchical Core 4 (p22810)			
w	INTEST$_P$ test time†	INTEST$_C$ test time†	
		with bypass	without bypass
2	285620	342755	742529
4	157933	305259	505061
6	115709	267763	505061
8	90334	227635	447614
10	88913	106787	276358
12	73079	52848	189362
14	70846	52848	184122
16	64147	26764	26764

†: All test times are presented in clock cycles.

Table 5. Test times of p22810 hierarchical Cores 1 and 4.

Figure 6 shows an example of a wrapped IEEE P1500-compliant hierarchical core. The parent core has scan-chain bypasses that can be activated during the EXTEST mode of the parent core. Child Core A has a wrapper-wide bypass that can help reduce the overall test length of the core in the INTEST$_C$ mode. All the multiplexers can be controlled by the wrapper instruction register in both the modes. In this example, an external TAM width of six bits is available. In INTEST$_P$ mode four wires are used for the parent core scan chains and its wrapper cells, and the remaining two are used for the child cores. In INTEST$_C$ all the six wires are connected to the six bit wide internal TAM.

In Table 5 we present the test times obtained in the two wrapper configurations for several external TAM widths. We also compare the test lengths of cores in the INTEST$_c$ mode, with and without wrapper-wide bypasses. While embedded cores are often designed with a wrapper-wide bypass, many cores do not contain this feature [2]; therefore, we consider both cases in our experiments. The child cores in Parent Core 1 and 4 do not have scan chains; they have wrapper cells only. Hence, scan chain bypasses have not been considered. We vary the external TAM width from 1 to 16 in steps of 2 for both the INTEST modes.

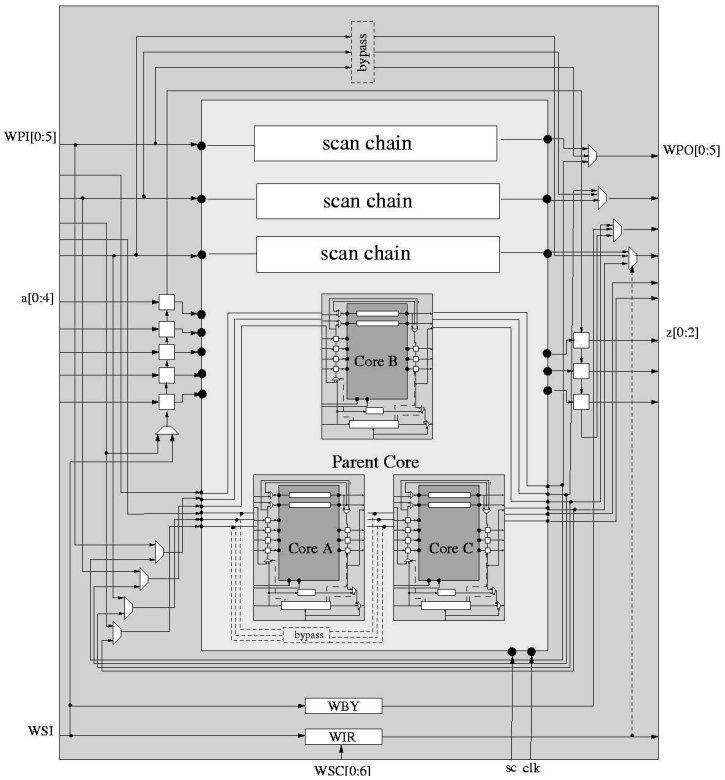

Figure 6. An example of a wrapped hierarchical core.

8 Conclusion

In this paper, we have addressed various issues involved in wrapper design for hierarchical cores. We have presented an IEEE P1500-compliant hierarchical core model and a hierarchical core wrapper architecture that can adapt itself to a wide range of industrial hierarchical cores. We have formulated the wrapper design problem as a two-fold problem and proposed heuristic approaches to solve it. We have also presented results for hierarchical cores in a benchmark SOC from industry and studied the impact of TAM-wide bypasses on the test time of hierarchical cores. As future work, we intend to develop wrapper design techniques for the other design scenarios we have described in Section 3.

References

[1] Santanu Dutta, Rune Jenson, and Alf Rieckmann. Viper: A Multiprocessor SOC for Advanced Set-Top Box and Digital TV Systems. *IEEE Design & Test of Computers*, 18(5):21–31, September 2001.

[2] Sandeep Kumar Goel, Kuoshu Chiu, Erik Jan Marinissen, Toan Nguyen, and Steven Oostdijk. Test Infrastructure Design for the Nexperia^{TM} Home Platform PNX8550 System Chip. In *Proceedings Design, Automation, and Test in Europe (DATE)*, pages 108–113 (Designer's Forum Proceedings), Paris, February 2004.

[3] Erik Jan Marinissen, Vikram Iyengar, and Krishnendu Chakrabarty. A Set of Benchmarks for Modular Testing of SOCs. In *Proceedings IEEE International Test Conference (ITC)*, pages 519–528, Baltimore, MD, October 2002, (see: http://www.extra.research.philips.com/itc02socbenchm/).

[4] Yervant Zorian, Erik Jan Marinissen, and Sujit Dey. Testing Embedded-Core-Based System Chips. *IEEE Computer*, 32(6):52–60, June 1999.

[5] Sandeep Kumar Goel and Erik Jan Marinissen. Effective and Efficient Test Architecture Design for SOCs. In *Proceedings IEEE International Test Conference (ITC)*, pages 529–538, Baltimore, MD, October 2002.

[6] Sandeep Koranne. A Novel Reconfigurable Wrapper for Testing of Embedded Core-Based SOCs and its Associated Scheduling Algorithm. *Journal of Electronic Testing: Theory and Applications*, 18(4/5):415–434, August 2002.

[7] Erik Jan Marinissen, Sandeep Kumar Goel, and Maurice Lousberg. Wrapper Design for Embedded Core Test. In *Proceedings IEEE International Test Conference (ITC)*, pages 911–920, Atlantic City, NJ, October 2000.

[8] Vikram Iyengar, Krishnendu Chakrabarty, and Erik Jan Marinissen. Co-Optimization of Test Wrapper and Test Access Architecture for Embedded Cores. *Journal of Electronic Testing: Theory and Applications*, 18(2):213–230, April 2002.

[9] Huan-Shan Hsu et al. Test Scheduling and Test Access Architecture Optimization for System-on-Chip. In *Proceedings IEEE Asian Test Symposium (ATS)*, pages 411–416, Tamuning, Guam, USA, November 2002.

[10] Erik Larsson and Zebo Peng. An Integrated Framework for the Design and Optimization of SOC Test Solutions. *Journal of Electronic Testing: Theory and Applications*, 18(4/5):385–400, August 2002.

[11] Yu Huang, Sudhakar M. Reddy, and Wu-Tung Cheng. Core-Clustering Based SOC Test Scheduling Optimization. In *Proceedings IEEE Asian Test Symposium (ATS)*, pages 405–410, Tamuning, Guam, USA, November 2002.

[12] Erik Jan Marinissen et al. A Structured And Scalable Mechanism for Test Access to Embedded Reusable Cores. In *Proceedings IEEE International Test Conference (ITC)*, pages 284–293, Washington, DC, October 1998.

[13] Prab Varma and Sandeep Bhatia. A Structured Test Re-Use Methodology for Core-Based System Chips. In *Proceedings IEEE International Test Conference (ITC)*, pages 294–302, Washington, DC, October 1998.

[14] Francisco DaSilva, Yervant Zorian, Lee Whetsel, Karim Arabi, and Rohit Kapur. Overview of the IEEE P1500 Standard. In *Proceedings IEEE International Test Conference (ITC)*, pages 988–997, Charlotte, NC, September 2003.

[15] Erik Jan Marinissen et al. On IEEE P1500's Standard for Embedded Core Test. *Journal of Electronic Testing: Theory and Applications*, 18(4/5):365–383, August 2002.

[16] Alfredo Benso et al. HD^{2}BIST: A Hierarchical Framework for BIST Scheduling, Data Patterns Delivering and Diagnosis in SoCs. In *Proceedings IEEE International Test Conference (ITC)*, pages 892–901, Atlantic City, NJ, October 2000.

[17] Tapan J. Chakraborty, Sudipta Bhawmik, and Chen-Huan Chiang. Test Access Methodology for System-On-Chip Testing. In *Digest of Papers of IEEE International Workshop on Testing Embedded Core-Based Systems (TECS)*, pages 1.1–1–7, Montreal, Canada, May 2000.

[18] Mounir Benabdenbi, Walid Maroufi, and Meryem Marzouki. CAS-BUS: A Test Access Mechanism and a Toolbox Environment for Core-Based System Chip Testing. *Journal of Electronic Testing: Theory and Applications*, 18(4/5):455–473, August 2002.

[19] Vikram Iyengar, Krishnendu Chakrabarty, Mark D. Krasniewski, and Gopind N. Kumar. Design and Optimization of Multi-level TAM Architectures for Hierarchical SOCs. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 299–304, Napa, CA, April 2003.

[20] Sandeep Kumar Goel. An Improved Wrapper Architecture for Parallel Testing of Hierarchical Cores. In *Proceedings IEEE European Test Symposium (ETS)*, pages 147–152, Corsica, France, May 2004.

[21] Erik Jan Marinissen, Rohit Kapur, and Yervant Zorian. On Using IEEE P1500 SECT for Test Plug-n-Play. In *Proceedings IEEE International Test Conference (ITC)*, pages 770–777, Atlantic City, NJ, October 2000.

[22] Erik Jan Marinissen. The Role of Test Protocols in Automated Test Generation for Embedded-Core-Based System ICs. *Journal of Electronic Testing: Theory and Applications*, 18(4/5):435–454, August 2002.