Hierarchy-Aware and Area-Efficient Test Infrastructure Design for Core-Based System Chips[1]

Anuja Sehgal[†], Sandeep Kumar Goel[‡], Erik Jan Marinissen[‡] and Krishnendu Chakrabarty[§]

[†]Advanced Micro Devices, Sunnyvale, CA 94085, USA

[‡]Philips Research Labs, 5656 AA Eindhoven, The Netherlands

[§]Duke University, Durham, NC 27708, USA

Abstract

Multiple levels of design hierarchy are common in current-generation system-on-chip (SOC) integrated circuits. However, most prior work on test access mechanism (TAM) optimization and test scheduling is based on a flattened design hierarchy. We investigate hierarchy-aware test infrastructure design, wherein wrapper/TAM optimization and test scheduling are carried out for hierarchical SOCs for two practical design scenarios. In the first scenario, the wrapper and TAM implementation for the embedded child cores in hierarchical (parent) cores are delivered in a hard form by the core provider. In the second scenario, the wrapper and TAM architecture of the child cores embedded in the parent cores are implemented by the system integrator. Experimental results are presented for the ITC'02 SOC test benchmarks.

1 Introduction

The integration of a complete electronic system on a single chip is now commonplace. A system-on-chip (SOC) typically integrates a heterogeneous mix of digital logic, embedded memories, and analog blocks. To reduce test cost, SOCs are being increasingly tested in a modular fashion, i.e., their various design modules are tested separately [1, 2]. Such an approach is mandatory for embedded non-logic components such as memories and analog modules, as well as for black-boxed third-party cores. However, also for other parts of the SOC, modular testing brings advantages, including reduced ATPG complexity and greater test reuse. Modular testing uses an on-chip test access infrastructure, which consists of test wrappers and test access mechanisms (TAMs) [1]. Test wrappers isolate the various modules from their surrounding circuitry during test. TAMs transport test stimuli between SOC pins and module terminals, and test responses vice versa.

Multiple levels of design and test hierarchy in current-generation SOCs are quite common. A *'parent'* core may contain several *'child'* cores, which in turn may contain their own embedded (*'grandchild'*) cores. For example, [3, 4] describe SOCs for digital video, where the SOC design is partitioned into chiplets, which in turn consist of embedded cores. Despite this, most prior research on SOC testing (unrealistically) assumed that there is no design and test hierarchy, i.e., that all cores are at the same hierarchy level [6, 7, 2]. Test wrappers, TAMs, and test schedules designed for non-hierarchical SOCs are typically not valid for SOCs with hierarchical cores. The hierarchy imposes a number of constraints on the manner in which tests must be applied to parent cores and their embedded child cores [8]; hierarchy-oblivious methods make no attempt to satisfy these constraints.

In this paper, we describe a test infrastructure design approach for hierarchical SOCs; our approach is based on hierarchy-aware wrapper design for parent cores, TAM optimization techniques for the SOC and the parent cores, and chip-level test scheduling. We consider two different test infrastructure design scenarios. In Scenario 1, we assume that the wrappers and TAM architectures for the child cores are given and fixed (*hard*), while the wrappers and TAM architectures for the parent cores are to be determined by our approach (*soft*). In Scenario 2, the wrapper and TAM for both parent and child cores are assumed to be soft.

The sequel of this paper is organized as follows. In Section 2, we review the limitations of prior work. In Section 3, we discuss various DfT techniques that can be used to reduce test length. Section 4 describes our approach for Scenario 1, while Section 5 addresses Scenario 2. We derive lower bounds on the test time in Section 6. In Section 7, we present test application times and lower bounds for the ITC'02 SOC test benchmarks [5]. Finally, Section 8 concludes the paper.

2 Limitations of Prior Work

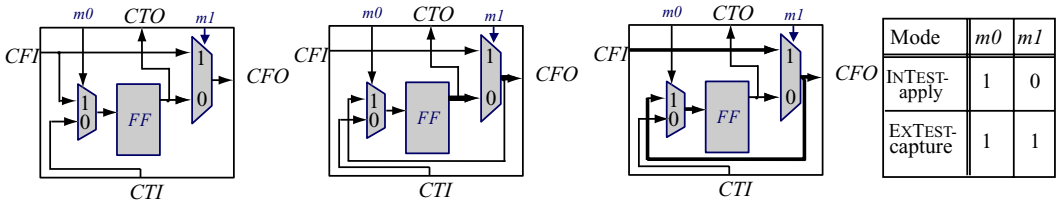
Most prior work on wrapper/TAM optimization for SOCs has assumed a non-hierarchical test infrastructure [2, 6, 7, 10, 11]. In comparison, only a limited amount of work has been done on wrapper design and TAM optimization for hierarchical SOCs. Recently in [8], the issue of wrapper design for hierarchical cores has been addressed, and in [9, 12, 13], wrapper design and TAM optimization techniques for hierarchical cores have been explored.

2.1 Wrapper Design Issues

To understand the differences between the testing of non-hierarchical and hierarchical SOCs, it is important to understand the functionality of some elements of the IEEE P1500 Wrapper architecture [11]. The main difference between testing non-hierarchical and hierarchical SOCs arises due to the functionality of the wrapper cells that buffer the functional inputs and outputs of the core. The input wrapper cell shifts and applies test stimuli in the INTEST mode, and captures and shifts test responses in the EXTEST mode. The operation of the output wrapper cells is the reverse of the input wrapper cells in the two modes.

[1]This research was supported in part by the National Science Foundation under grants CCR-9875324 and CCR-0204077. This work was carried out whle Anuja Sehgal was a PhD student at Duke University.

3-9810801-0-6/DATE06 © 2006 EDAA



Mode	*m0*	*m1*
INTEST-apply	1	0
EXTEST-capture	1	1

Figure 1. (a) "Default" IEEE P1500 input wrapper cell; (b) Modified IEEE P1500 wrapper cell in INTEST-apply cycle; (c) Modified IEEE P1500 wrapper cell in EXTEST-capture cycle; (d) Control signals for the multiplexers in the two modes.

Figure 1(a) shows a basic "default" IEEE P1500 wrapper cell; *CFI* and *CFO* are the functional input and output of the wrapper cell respectively, and *CTI* and *CTO* are the test data inputs and outputs respectively. During the INTEST mode, the path from *CTI* to *CFO* through the flip-flop and the two multiplexers is used, and during EXTEST mode, the path from *CFI* to the flip-flop is used to capture the test response from the chip interconnect. Consequently, the path from *CFI* to *CFO* through the upper selection of the Mux when $m1 = 1$ remains untested. This is a drawback of this wrapper cell, which is highly undesirable since the untestable path is used in functional mode. Hence an alternative P1500-compliant wrapper cell configuration, as shown in Figure 1(b), is more commonly used in practice, e.g., at Philips. The paths used during INTEST-apply and EXTEST-capture cycles are highlighted in Figure 1(b) and (c) respectively. Both paths of the second multiplexer are exercised in the test mode, thereby allowing better testability of the cell. Also, during INTEST-apply cycle, the feedback path causes the test data bit of the previous cycle to be fed back into the flip-flop. This is an added advantage of this cell, since it can prevent data corruption due to clock skew in back to back wrapper cells. However, in this wrapper cell configuration, the feedback path also implies that the apply cycle of the INTEST mode cannot coincide with the capture cycle of the EXTEST mode. The multiplexer control signal values for the two modes are conflicting, as shown in Figure 1(d).

In this paper, and in [8], the use of the wrapper cell configuration of Figure 1(b) is assumed. For a hierarchical core, suppose that the parent core and the child cores are tested on different TAM partitions. When the parent core is in the INTEST mode, the embedded child cores have to be in EXTEST mode to ensure complete internal testing of the parent core [8, 9]. Thus, the parent and the child INTEST testing cannot be done concurrently, since the child cores cannot be configured simultaneously in INTEST and EXTEST mode. Due to this added constraint, the wrapper/TAM optimization techniques that are used for non-hierarchical SOCs cannot be directly used for hierarchical SOCs that rely on the wrapper cell configuration shown in Figure 1(b). Nevertheless, the design in Figure 1(b) is still the preferred wrapper cell due to its better testability.

2.2 Wrapper/TAM Optimization

In [12], a wrapper/TAM optimization approach is presented, which uses an existing hierarchy-oblivious TAM optimization approach iteratively to solve the problem of TAM optimization for hierarchical SOCs. However, in this approach, the constraint of child and parent INTEST testing being time multiplexed is ignored. In [13], a TAM design technique for hierarchical SOCs is presented in which the hierarchical cores are assumed to be hard wrapped cores. This approach requires large area overhead due to the added registers for bandwidth matching, and it also requires synchronization of the clock signals.

In [9], modified input and output wrapper cells are used to allow concurrent INTEST testing of the parent and child cores. An extra flip-flop and a multiplexer are used in this configuration. Therefore, the wrapper cells used in this approach have larger area compared to the conventional IEEE P1500 wrapper cells; thus they may not always be feasible.

In [8], the problem of designing an IEEE P1500-compliant wrapper for hierarchical cores has been addressed. The IEEE P1500-compliant wrapper for hierarchical cores has a new set of terminals, called the *CTAM* terminals, which are used to access the child cores embedded in a parent core. The wrapper for the parent core operates in two complementary test modes. In the INTEST$_P$ mode, internal testing of the parent core is carried out. Test data is scanned through the parent core scan chains, the parent core wrapper cells, and the child core wrapper cells. The child cores have to be in the EXTEST mode to ensure complete core-internal testing of the parent core. Hence, in the INTEST$_P$ mode, the available TAM wires are distributed between the parent core scan chains and wrapper cells, as well as the child core TAM architecture. This was ignored in the prior work on TAM optimization for hierarchical cores in [12]. In the INTEST$_C$ mode, child core internal testing is carried out; all child cores are in INTEST mode. In this mode, the available TAM width at the wrapper interface can be utilized for child core testing alone. Since the two modes operate independently, they are time-multiplexed using multiplexers.

The wrapper design approach presented in [8] is suitable only for hierarchical cores with a hard TAM architecture. Moreover, the wrapper design approach of [8] has not been used for TAM optimization. In this paper, we solve the wrapper and TAM optimization problem in conjunction for two different design scenarios and evaluate the approach using the metrics of area impact and test application time.

3 DfT Techniques for Test Length Reduction

In this section, we discuss the impact of using (a) scan chain bypass, (b) core bypass, and (c) scan chain reordering, on the test time for a hierarchical core. We also discuss the impact of these DfT techniques on chip area. Scan chain bypass and core bypass have been studied in prior work [14], but they have not been discussed in the context of hierarchical cores.

Scan chain bypass : The test time $T_i(w)$ for a core C_i is defi ned as:$T_i(w) = (1 + \max(si_i, so_i)) \times p_i + \min(si_i, so_i)$, where si_i and so_i are the maximum scan-in and scan-out length of core C_i for a TAM width of w, and p_i is the number of test patterns. Typically, it is assumed that for any given core, the test data volume in the INTEST mode is much greater than the test data volume in the EXTEST mode. Thus, the child core architecture, which consists of the child cores' wrapper and the embedded TAM, is optimized for the INTEST mode only. During INTEST, the test stimuli are scanned into the input wrapper cells and scan chains, and the test responses are scanned out of the scan chains and the output wrapper cells. Hence, the scan chains are sandwiched between the input and output wrapper cells. This ensures smaller scan-in and scan-out lengths during INTEST, compared to the scan chains being ordered before or after the wrapper cells. However, in the EXTEST mode, the scan chains of the child cores add to the scan-in and scan-out length even though they do not participate in the test. This additional contributer to the scan length can be eliminated by using a scan chain bypass. A fl ip-fl op or register is not required in the bypass path if the scan-chains are local to the same module. Typically, a scan chain bypass has an area overhead of one multiplexer. Thus, this trade-off between scan length reduction and increased DfT overhead has to be carefully evaluated. In the case of a hierarchical core, the scan length reduction in child core EXTEST mode pays off only if the scan chain for the child core architecture is long enough to be the maximum scan-in and scan-out length of the parent core.

Core bypass: The core bypass feature allows the bypass of an entire core in one clock cycle by means of a bypass register or fl ip-fl op across the core.. It has been shown that this feature can reduce the test time signifi cantly in a TestRail architecture [2]. In [8], it is shown that core bypasses can reduce the test time of hierarchical cores as well. If the TAM width at the parent core's interface is less than the number of CTAMs, test length is likely to be reduced if one or more CTAM chains of the child core architecture are daisy-chained. This results in an increase in the scan-in and scan-out lengths of all the cores. A core bypass can reduce the impact of daisy-chaining on the scan lengths of the cores. Scan chain bypasses can also reduce the scan-in and scan-out lengths in this example; however, core bypasses are more effective in the INTEST$_C$ mode since they eliminate the wrapper cells in the scan path as well. Scan chain bypasses are more useful for the INTEST$_P$ mode, because the scan chains are excluded from the scan path without excluding wrapper cells, which are essential for the INTEST$_P$ mode.

Scan chain reordering: This is another technique that can be explored to reduce the CTAM scan length of the child cores during the INTEST$_P$ mode. Compared to the INTEST mode, the roles of the input and output wrapper cells in the EXTEST mode are reversed. Ideally, in the EXTEST mode, the output wrapper cells should precede the input wrapper cells to minimize the scan-in and scan-out lengths. By using additional hardware, it is possible to make the wrapper reconfi gurable in the EXTEST mode, such that the order of the wrapper cells is reversed. This technique requires three two-input multiplexers for each scan path of the core. The benefi ts of using this type of design optimization, as in the case of scan chain and core bypasses, depends on the child core TAM architecture available. The improvements in the CTAM lengths depends on the difference in the number of input and output wrapper cells of the cores that interface directly with the CTAM terminals.

4 Design Scenario I

In this scenario, the parent core provider implements the wrapper and TAM architecture for the child cores before core delivery. The system integrator designs the overall wrapper/TAM architecture for the hierarchical SOC. The information provided for the hierarchical core by the core provider includes information about the test infrastructure for the child cores. This information is used for wrapper design of the hierarchical core in the INTEST$_P$ and INTEST$_C$ modes.

The problem of wrapper/TAM optimization for this scenario is stated as follows.

Problem P_I: Given (i) a hierarchical SOC with N top-level cores, where one or more top-level core may be a parent core with embedded child cores, (ii) the wrapper and TAM architecture for each embedded child core, (iii) the test set parameters for each top-level core C_i, $1 \leq i \leq N$, and (iv) the SOC-level TAM width W, determine:
1. The number of TAM partitions and the partition widths;
2. Wrapper design for each top-level core C_i, $1 \leq i \leq N$;
3. Assignment of top-level cores to TAM partitions,
such that the testing time for the SOC is minimized. □

The test set parameters for each core C_i, $1 \leq i \leq N$, includes the number of primary inputs fi_i, primary outputs fo_i, bidirectional I/Os b_i, test patterns p_i, scan chains k_i, and the different scan chain lengths. The cores are assumed to have hard scan chains, i.e, the number and length of scan chains are fi xed. IfC_i is a hierarchical (parent) core, the test data for it also includes the test data and test infrastructure parameters for its child cores. For any given parent core C_i, the test data and test infrastructure for its embedded child core includes the following: (i) the set $\mathcal{M}_i$ of CTAM chains; (ii) the set $\mathcal{CC}_i$ of embedded child cores; (iii) for each child core $c \in \mathcal{CC}_i$, the number of test patterns p_c, the total scan length, scan-in length, and scan-out length on CTAM chain k, $1 \leq k \leq |\mathcal{M}_i|$, denoted by $sl_{c,k}$, $si_{c,k}$, and $so_{c,k}$, respectively. If core C_i is not hierarchical, $\mathcal{M}_i = \emptyset$ and $\mathcal{CC}_i = \emptyset$.

If all cores in the SOC are non-hierarchical, i.e., $\mathcal{M}_i = \emptyset$, $1 \leq i \leq N$, P_I reduces to Problem $\mathcal{P}_{\text{NPAW}}$, the wrapper/TAM optimization problem for non-hierarchical SOCs from [6]. Since

$\mathcal{P}_{\text{NPAW}}$ was shown to be $\mathcal{NP}$-hard in [6], P_I is also $\mathcal{NP}$-hard. Hence, we resort to effi cient heuristics to solve this problem.

In the IEEE P1500-compliant wrapper design technique presented in [8], wrapper design allows a hierarchical core to interface with a TAM partition of any width. Thus, using this wrapper design technique, hierarchical cores can be integrated into an existing wrapper/TAM optimization technique for non-hierarchical SOCs. We implement wrapper/TAM optimization for this scenario using the TR-ARCHITECT tool [2].

The TR-ARCHITECT tool includes four main procedures, namely: (1) CREATESTARTSOLUTION; (2) OPTIMIZE-BOTTOMUP; (3) OPTIMIZE-TOPDOWN; (4) RESHUFFLE. In CREATESTARTSOLUTION, an initial TAM partition is created by assigning to each core, at least one TAM wire based on its test data volume. The other procedures optimize the initial solution. The procedure OPTIMIZE-BOTTOMUP attempts to minimize the test time for a given test architecture by merging the TAM partition with the shortest test time with another TAM partition, such that the current test time of the SOC is not exceeded. The wires that are thus freed up can be used for overall test time reduction. In OPTIMIZE-TOPDOWN, fi rst, the TAM partition with the largest test time is merged in an iterative manner with another TAM partition. If the fi rst step does not reduce the test time any more, two TAM partitions that are not the partitions with the largest test time are merged to free up wires, provided the test time does not increase. The freed up wires are then used to reduce the overall SOC time further. The procedure RESHUFFLE minimizes the test time for a given test architecture by moving individual cores assigned to the TAM partition with the largest test time to another TAM partition, such that the current time of the SOC is not exceeded. The test times for the cores, for any given TAM width, is obtained using procedure WRAPPERDESIGN. In our work, the wrapper design technique from [8] is used for hierarchical cores and WRAPPERDESIGN is used for non-hierarchical cores.

For our experimental setup, we use TR-ARCHITECT to create a TAM architecture for the child cores embedded in the parent cores in the ITC'02 SOC benchmarks. We assume an internal CTAM width for the hard TAM architectures of the child cores, and use TR-ARCHITECT to design the wrapper and TAM architectures for the child cores. The TAM architecture is designed to minimize the overall test length of the child cores. From the results thus obtained, it is possible to determine the scan-in, scan-out and scan-lengths of the cores on each CTAM chain. This information is suffi cient for the wrapper design procedure used in the overall design fl ow of the hierarchical SOC. We present experimental results for two of the ITC'02 benchmark SOCs in Section 7.

5 Design Scenario II

In this scenario, since the embedded child core test infrastructure is soft, the system integrator can design the wrapper and TAM architecture of the child cores in accordance with the TAM width available at the parent core interface. Thus, if w TAM wires are available at the parent core level, the child core architecture can be designed to have up to w CTAM terminals. Thus, the CTAM terminals can be directly connected to the available TAM width, and external daisy chaining of the CTAM chains is not required. In this case, the problem of designing the wrapper and TAM architecture for the child cores corresponds to that of designing the wrapper and TAM architecture for an SOC; the TAM architecture is designed for an SOC-level TAM width of w. However, wrapper design for the parent core and the overall wrapper/TAM optimization problems for the hierarchical SOC are still different from the corresponding problem for non-hierarchical cores .

The problem statement for this scenario is as follows.

Problem P_{II}: Given (i) a hierarchical SOC $\mathcal{S}$ with a set $\mathcal{C}$ of N cores, i.e., $\mid \mathcal{C} \mid = N$, (ii) the test parameters and the parent core PC_i for each core C_i, where $1 \leq i \leq N$, $C_i \in \mathcal{C}$, and $PC_i \in \{\mathcal{S}\} \cup \mathcal{C} \backslash \{C_i\}$, and (iii) the total SOC-level TAM width W, determine:

1. The number of TAM partitions and the partition widths;
2. Wrapper design for each core C_i;
3. Assignment of cores to TAM partitions,

such that the total testing time for the SOC is minimized. □

The test parameters for each core C_i, $1 \leq i \leq N$ includes the number of primary inputs fi_i, primary outputs fo_i, bidirectional I/Os b_i, test patterns p_i, scan chains k_i, and the different scan chain lengths. The cores are assumed to have hard scan chains, i.e., the number and lengths of scan chains are fi xed. The hierarchy tree is determined from the parent module information given for each core. The parent for the top-level modules is the SOC itself. If the parent module for every core in an SOC is the SOC itself, i.e., $PC_i = \mathcal{S}$, $1 \leq i \leq N$, then P_{II} for that SOC reduces to Problem $\mathcal{P}_{\text{NPAW}}$ from [6]. Thus, by the method of restriction, Problem P_{II} is also an $\mathcal{NP}$-hard problem.

We extend TR-Architect in two main ways to solve P_{II}: (a) we pre-process the SOC specifi cations, such that only the top-level cores are presented to the TR-ARCHITECT tool as described in [2], and (b) we replace WRAPPERDESIGN with another procedure called HIERWRAPPER. The HIERWRAPPER procedure makes use of TR-ARCHITECT, WRAPPERDESIGN, and WRAPP, where WRAPP is the wrapper design technique for INTEST$_P$ mode, as presented in [8].

6 Lower Bounds on Test Time

For each hierarchical core C_i, a lower bound on test time LB_i can be defi ned as$LB_i = LBC_i + LBP_i$, where LBC_i is the lower bound on test time in the INTEST$_C$ mode, and LBP_i is the lower bound on test time in the INTEST$_P$ mode. For the case of hard embedded child cores in hierarchical cores, LBC_i corresponds to the test time for the hard child core TAM architecture in clock cycles. For soft embedded child core architectures in hierarchical cores, the lower bound on test time LBC_i for Core C_i can be determined using a lower bound expression defi ned for non-hierarchical SOCs, as presented in [2]. Thus, $LBC_i = \min\{LB_T^1, LB_T^2\}$, where LB_T^1 is an architecture-independent lower bound based on the maximum test time of

the wrapped cores, and LB_T^2 is a lower bound based on the test data volume of the SOC.

The formulation for LBP_i is based on the test data volume for the INTEST$_P$ mode. For the case of soft embedded child core architectures, the scan-in and scan-out lengths of the parent core used for calculating test data volume are defined as

$$si_i = nff_i + fi_i + b_i + \sum_{j=1}^{nc_i}(nff_j + fi_j + fo_j + b_j) \quad (6.1)$$

$$so_i = nff_i + fo_i + b_i + \sum_{j=1}^{nc_i}(nff_j + fi_j + fo_j + b_j) \quad (6.2)$$

where the various parameters in (1) and (2) are defined as follows:

1. si_i is the scan-in length for core C_i;
2. so_i is the scan-out length for core C_i;
3. nff_i is the number of flip-flops for core C_i;
4. fi_i is the number of functional inputs for core C_i;
5. fo_i is the number of functional outputs for core C_i;
6. b_i is the number of bidirectional terminals for core C_i;
7. nc_i is the number of child cores in core C_i.

If child core scan chain bypasses are enabled, we disregard the number of flip-flops in the child cores in (1) and (2). For a hard implementation of the child core architecture, the scan-in and scan-out lengths of the parent core are defined as $si_i = FF_i + fi_i + b_i + Tsi$, and $si_i = FF_i + fi_i + b_i + Tso$, where Tsi and Tso are the maximum scan-in and scan-out lengths of the child core architecture, respectively, in the EXTEST mode of the child core wrappers.

The test data volume of Core C_i is defined as $v_i = \max\{si_i, so_i\} \cdot p_i + \min\{si_i, so_i\}$, where p_i is the number of test patterns. Thus, a lower bound (in clock cycles) on the test time of parent core C_i for a SOC level TAM width of W is defined as

$$LBP_i = \lceil v_i/W \rceil + p_i. \quad (6.3)$$

Using Equation (6.3), a lower bound on test time can be calculated for each parent core for both design scenarios. The sum of the lower bounds for all the individual cores in the SOC provides a lower bound on the test time for the SOC. However, if the SOC has multiple TAM partitions, the TAM partition with the maximum test time can have only one core with the minimum number of test patterns. Hence, only the minimum number of test patterns among all cores is used in the lower bound expression for SOC test time. Also, we assume that the scan-out of the responses of the last test pattern of a core is not overlapped with the scan-in of the stimuli of the first test pattern of another core on the same level of hierarchy. Thus, a lower bound LB_1^H for hierarchical cores can be defined as

$$LB_1^H = \lceil \sum_{i=1}^{N} v_i/W \rceil + \min_{1 \le i \le N}\{p_i\} + \sum_{i=1}^{N} LBC_i \quad (6.4)$$

We also obtain another lower bound LB_2^H from [15] as follows: $LB_2^H = \max_{1 \le i \le N}\{T_i(W)\}$, where $T_i(W)$ is the test time for core C_i at core-level TAM width of W. The bound LB_2^H is more accurate for larger TAM widths ($W > 32$). However, for smaller values of W, LB_1^H is tighter. Hence, the overall lower bound is determined as $LB^H = \max\{LB_1^H, LB_2^H\}$.

p22810		p34392	
Module name	# CTAMs	Module name	# CTAMs
Module 1	15	Module 2	20
Module 5	16	Module 10	20
		Module 18	16

Table 1. Number of CTAMs specified for the hierarchical modules in p22810 and p34392.

p22810				
W	LB_T^H	T_0	T_{byp}	$\Delta T(\%)$
16	426459	711615	544763	− 23.44
24	288652	486576	386464	− 20.57
32	239205	325479	291539	− 10.42
40	239205	239205	239205	0
48	239205	239205	239205	0
56	239205	239205	239205	0
64	239205	239205	239205	0
p34392				
W	LB_T^H	T_0	T_{byp}	$\Delta T(\%)$
16	1139550	2080643	1484442	− 28.65
24	644179	1433969	976347	− 31.91
32	606261	827486	792544	−4.22
40	606261	776537	606261	−21.92
48	606261	657210	606261	−7.77
56	606261	606261	606261	0
64	606261	606261	606261	0

LB_H^T: Lower bound on test time in clock cycles; T_0: Test application time (in clock cycles) without core bypass;

T_{byp} : Test application time(in clock cycles) with core bypass; ΔT: $\frac{T_{byp} - T_0}{T_0} \times 100$.

Table 2. Experimental results for Design Scenario I.

7 Experimental Results

We first present experimental results for p22810 and p34392 for Design Scenario I. For the experimental setup, the TAM optimization design tool TR-ARCHITECT is used to create a hard implementation of the child core architectures. The child core architecture for the two hierarchical cores in p22810 is the same as that presented in [8]. The CTAM widths used for the various hierarchical cores in p22810 and p34392 are shown in Table 1. We present results on test times obtained for different values of TAM width for the two benchmark SOCs in Table 2. The test times are presented for two cases: (a) when child core architectures in hierarchical cores are equipped with core bypasses, and (b) when core bypasses are not implemented. Lower bounds on test time are also listed in Table 2.

From Table 2, we observe that the lower bounds on test time do not decrease any further for TAM widths greater than 32. This is due to the presence of a bottleneck core that dominates the SOC test time. In SOC p22810, hierarchical Core 1 dominates the test time due to long scan lengths in its child core architecture. As a result, the test time reaches the lower bound value of 239205 clock cycles for $W > 32$. Similarly, hierarchical Core 18 dominates the test time for p34392 for $W > 32$. Note that the test time of any hierarchical core cannot decrease below the test time of it's child core architecture, once the CTAM width of the child core architecture is equal to the parent-core level TAM width. In both SOCs, for $W > 32$, the CTAM width of the child core architectures is matched with the available TAM width at the

W	[9]		Proposed Approach			ΔT	ΔNG
	T_o	NG_o	LB_T^H	T_n	NG_n	%	%
p22810							
16	466667	61481	435526	530778	55721	13.73	−9.36
24	309641	61481	294697	343942	55730	11.07	−9.35
32	229899	61481	230445	288273	55736	25.39	−9.34
40	191978	61481	192408	263624	55739	37.31	−9.33
48	157226	61481	153867	251299	55760	59.83	−9.30
56	145417	61021	136051	238974	55790	64.33	−8.57
64	133405	61481	127614	238974	55817	79.13	−9.21
p34392							
16	1019766	32435	965292	1154719	26789	13.23	−17.40
24	702852	29540	651838	774221	26813	10.15	−9.23
32	584524	32435	581588	606261	26834	3.71	−17.26
40	544579	29478	581588	593924	26852	9.06	−8.90
48	544579	30237	581588	581588	26879	6.79	−11.10
56	544579	30237	581588	581588	26900	6.79	−11.03
64	544579	30237	569252	581588	26900	6.79	−11.03
p93791							
16	1792354	95379	1770054	1854566	87179	3.47	−18.59
24	1211510	106880	1186843	1272220	87323	5.01	−18.29
32	917246	105422	892069	940318	87311	2.51	−17.17
40	730713	104342	714705	4765715	87359	4.79	−16.27
48	610037	106496	598571	640488	87281	4.99	−18.04
56	528407	107875	478150	551849	87380	4.43	−18.99
64	458600	106496	418382	473726	87455	3.29	−17.87

$\Delta T = \frac{T_n - T_o}{T_o} \times 100$; $\Delta NG = \frac{NG_n - NG_o}{NG_o} \times 100$;

Table 3. Comparison of the test time and area overhead of the proposed approach with the approach from [9].

parent core level for the bottleneck core. Table 2 also shows the test time results with and without core bypasses. As expected, for many values of W, the test time for the TAM architecture with core bypasses is less than that for the TAM architecture without core bypasses. The reduction in test time with the use of core bypasses decreases with an increase in W. This happens because the daisy chaining of the child core CTAM lengths typically reduces with an increase in TAM width. The core bypass implementation requires 46 and 96 additional 2-to-1 multiplexers in p22810 and p34392, respectively. If N_c is the set of child cores in an SOC, and w_i is the TAM width available to each child core C_i $i \in N_c$, the number of extra multiplexers required for the implementation of core bypasses is given by $\sum_{i=1}^{|N_c|} w_i$. Since the child core architecture is fixed, this number is independent of the SOC-level TAM width W. Thus, core bypasses can reduce the test time of an SOC significantly for smaller TAM width values at the expense of a small increase in chip area.

Next, we present results for Design Scenario II. In Table 3, we compare the test application time and chip area of the proposed approach to that of the method presented in [9]. In [9], modified wrapper cells are used, which allow parallel INTEST testing of the child cores and the parent cores, at the expense of an extra flip-flop and an extra multiplexer in each wrapper cell. The chip area is quantified in terms of the total number of NAND2 gates used in the wrapper cells of the SOC. It is assumed that a 2-to-1 multiplexer and a flip-flop is equivalent in area to 3 and 7 NAND2 gates, respectively. The derived lower bounds on test time are also presented for the proposed approach in Table 3.

In Table 3, the increase in test time for the proposed approach is accompanied by a decrease in the chip area compared to [9] for all the SOCs. In SOC p22810, the test time does not decrease as much with an increase in W due to hierarchical Core 1. Although, the SOC-level TAM width increases, the increase in the TAM width of Core 1 is not sufficient to reduce the test time of its child core architecture significantly. As a result, Core 1 dominates the test time of p22810. In p34392, the lower bounds on test time are reached for several TAM width values due to the bottleneck Core 18 that dominates the SOC test time. In this SOC, the reduction in chip area is significant compared to the increase in test time for all TAM width values. From these results, we conclude that for most SOCs, the gains in area are much higher compared to [9], and these gains can be achieved at the expense of a small increase in the overall SOC test application time.

8 Conclusion

We have addressed the problem of test infrastructure design for hierarchical SOCs. We have shown how a hierarchy-aware test planning method can be used for TAM optimization and test scheduling for hierarchical SOCs in two practical design scenarios. We have derived lower bounds on test time and presented experimental results for four ITC'02 SOC test benchmarks. We have shown that, in the first design scenario, core bypasses can reduce the test time of hierarchical SOCs significantly with a small increase in chip area. We have also shown that for the second design scenario, the wrapper area is much less compared to [9], but there is only a small increase in test application time.

References

[1] Y. Zorian et al. Testing Embedded-Core Based System Chips. In *Proc. ITC*, pp. 130–143, 1998.

[2] S. K. Goel and E. J. Marinissen. SOC Test Architecture Design for Efficient Utilization of Test Bandwidth. *ACM Trans. Design Automation of Electronic Systems*, 8(4):399–429, October 2003.

[3] S. Dutta et al. VIPER: A Multiprocessor SOC for Advanced Set-Top Box and Digital TV Systems. *IEEE Design & Test of Computers*, 18(5):21–31, Sep-Oct 2001.

[4] S. K. Goel et al. Test Infrastructure Design for the Nexperia$^{\text{TM}}$ Home Platform PNX8550 System Chip. In *Proc. DATE Designers Forum*, pp. 108–113, 2004.

[5] E. J. Marinissen et al. A Set of Benchmarks for Modular Testing of SOCs. In *Proc. ITC*, pp. 519–528, 2002.

[6] V. Iyengar et al. Co-Optimization of Test Wrapper and Test Access Architecture for Embedded Cores. *JETTA*, 18(2):213–230, April 2002.

[7] E. Larsson and Z. Peng. An Integrated Framework for the Design and Optimization of SOC Test Solutions. *JETTA*, 18(4/5):385–400, August 2002.

[8] A. Sehgal et al. IEEE P1500-Compliant Test Wrapper Design for Hierarchical Cores. In *Proc. ITC*, pp. 1203–1212, 2004.

[9] S. K. Goel. A Novel Wrapper Cell Design for Efficient Testing of Hierarchical Cores in System Chips. In *Digest of Papers of ETS*, pp. 147–152, 2004.

[10] E. J. Marinissen et al. A Structured And Scalable Mechanism for Test Access to Embedded Reusable Cores. In *Proc. ITC*, pp. 284–293, 1998.

[11] F. DaSilva at al. Overview of the IEEE P1500 Standard. In *Proc. ITC*, pp. 988–997, 2003.

[12] V. Iyengar at al. Design and Optimization of Multi-level TAM Architectures for Hierarchical SOCs. In *Proc. VTS*, pp. 299–304, 2003.

[13] Q. Xu and N. Nicolici. Time/Area Tradeoffs in Testing Hierarchical SOCs with Hard Mega-Cores. In *Proc. ITC*, pp. 1196–1202, 2004.

[14] E. J. Marinissen et al. Wrapper Design for Embedded Core Test. In *Proc. ITC*, pp. 911–920, 2000.

[15] K. Chakrabarty. Optimal Test Access Architectures for System-on-a-Chip. *ACM TODAES*, 6(1):26–49, January 2001.