Analysis of the test data volume reduction benefit of modular SOC testing

Analysis of The Test Data Volume Reduction Benefit of Modular SOC Testing

Ozgur Sinanoglu
Kuwait University
Math & Computer Science Dept.
Safat, Kuwait
ozgur@sci.kuniv.edu.kw

Erik Jan Marinissen
NXP Semiconductors
Corporate Innovation & Technology
Eindhoven, The Netherlands
erik.jan.marinissen@nxp.com

ABSTRACT

Modular SOC testing offers numerous benefits that include test power reduction, ease of timing closure, and test re-use among many others. While all these benefits have been emphasized by researchers, the test time and data volume comparisons has been mostly constrained within the context of modular SOC testing only, by comparing the impact of various different modular SOC testing techniques to each other. In this paper, we provide a theoretical test data volume analysis that compares the monolithic test of a flattened design with the same design tested in a modular manner; we present numerous experiments that gauge the magnitude of this benefit. We show that the test data volume reduction delivered by modular SOC testing directly hinges on the test pattern count variation across different modules, and that this reduction can exceed 99% in the SOC benchmarks that we have experimented with.

1. INTRODUCTION

The increasing complexity of VLSI designs has been successfully ameliorated through a modular design approach, wherein the design is partitioned into smaller blocks. Such a 'divide-n-conquer' design style also enables the import of external design expertise through the integration of third-party design blocks, also known as cores.

Not only the design, but also the test approach for a large SOC can be modular; the various modules and cores are then tested as stand-alone units. Modular testing requires the test modules to be *wrapped* for controllability and observability purposes, and all test wrappers to be connected to SOC pins by one or more *test access mechanisms* (TAMs) [1].

Embedded non-logic blocks, such as memories and analog, require dedicated tests for reasons of test quality. Hard cores and encrypted cores, for which no implementation netlist is available to the SOC integrator, need to be tested by the test patterns as delivered by their core provider, and hence also require stand-alone testing. However, even for logic-only SOCs for which the entire netlist is available, modular testing has attractive benefits. Modular test development breaks a large monolithic SOC design down into more digestable chunks, which become tractable for ATPG and fault-simulation tools. Modular test enables test reuse, which is particularly important if subsequent SOC designs are based on a family concept. And, modular testing allows for careful scheduling of its various component tests, in order to reduce average or absolute test completion time, power consumption, IR drop, etc.

A benefit of modular testing that has received scant attention so far is the fact that it typically quite effectively reduces the test data volume, when compared to a monolithic test approach. In this paper, we provide a theoretical analysis of that reduction. We show that the wrappers that enable modular SOC testing require a modest investment in extra test data volume, which is largely off-set by savings in test data volume due to variations in ATPG pattern counts. Instead of loading all scan flip-flops with the SOC-wide maximum number of test patterns, as is the case in the monolithic approach, modular testing allows to apply only the required number of test patterns to each individual core. It is assumed in this analysis that a core that is not being tested is disconnected from the TAM, eliminating the need to shift though such a core while testing another core. As cores in an SOC tend to show quite a large variation in test pattern count, significant test data volume savings can be obtained through a modular test approach. We illustrate our theory by experimental results with SOCs based on ISCAS'89 benchmark cores [2] and SOCs from the ITC'02 benchmarks [3].

The remainder of the paper is organized as follows. In Section 2, we present an overview of prior work in modular SOC testing. Section 3 provides a conceptual analysis of test data volume for both monolithic testing and modular SOC testing, while in Section 4, we present the corresponding test data volume equations. In Section 5, we provide experimental results and a quantitative analysis. The paper is concluded in Section 6.

2. RELATED PRIOR WORK

Challenges in modular, core-based SOC testing are described in [4]. Zorian et al. [1] introduced a generic conceptual test access architecture enabling modular testing of SOCs, consisting of three elements per module-under-test: (1) a test pattern source and sink, either off-chip (ATE) or on-chip (BIST), (2) a test access mechanism (TAM), and (3) a wrapper. The wrapper can isolate the module from its surroundings and provides switching functionality between functional access to the module and test access through the TAM. Most approaches published since then rely on test wrappers and TAMs.

Wrapper design has been described in [5, 6] and interoperable wrappers have been standardized as IEEE Std. 1500 [7, 8, 9]. TAM types published include the test bus [10] and the TestRail [11]. TAM architectures range from the *all-cores-in-one-TAM* approach (such as the Multiplexing and Daisychain Archi-

978-3-9810801-3-1/DATE08 © 2008 EDAA

tectures [12]) to *each-core-in-a-private-TAM* (Distribution Architecture [12]), and hybrid combinations thereof [13]. The design of a TAM architecture and the set of feasible test schedules are strongly intertwined. There are many papers on SOC test scheduling, optimizing wrappers and TAMs [14, 13], taking into account fail probabilities [15, 16], and test power [17, 18]. A more detailed survey is provided in [19].

Recent work that is more related to the analysis we present herein consists of a SOC partitioning approach [20], a single TAM daisy-chain architecture [21] and a soft core isolation technique in [22]. In [20] and [21], the authors emphasize the benefit of modular testing on test time, and provide experimental data on SOCs with a particular TAM. In [22], it was noted that modular SOC testing leads to a small increase of the test data volume, due to the wrapper cells that require additional test data bits. In this paper, we focus our analysis on test data volume irrespective of any underlying TAM architecture. Exploring the underlying reasons for test data volume reduction benefit of modular SOC testing compared to monolithic testing, modeling the reduction via formulations, and experimentation to quantify this analysis are the key contributions of our paper.

3. CONCEPTUAL COMPARISON

Automatic Test Pattern Generation (ATPG) tools essentially work on a per-cone basis. A logic cone consists of all the combinational logic driving one flip-flop or circuit output. A logic cone is driven from one or multiple flip-flops, circuit inputs, or a combination thereof. Conventional (full) scan design makes (all) flip-flops both controllable and observable, as if they were regular primary inputs and outputs. Fault sensitization, propagation, and justification in ATPG tools are all done within the scope of one logic cone. A test pattern for the logic cone consists of the assignment of 0, 1, and X (don't care) bits to all the inputs of the cone, and a 0, 1, or Z (high-impedance) expected response bit at the cone's output.

A circuit normally consists of multiple logic cones. A test pattern for a single logic cone is therefore typically only a partial test pattern for the entire circuit, as it only defines stimulus and response bits for the inputs and output of that particular cone. The process of merging multiple partial, per-cone test patterns into one circuit-level test pattern is commonly referred to as *compaction*. ATPG theory distinguishes between 'static compaction', where compaction is done as a post-processing step, and 'dynamic compaction', in which compaction is integrated in the pattern generation itself. Partial test patterns can be merged into one test pattern, if their stimulus bits are non-conflicting. Two stimulus bits of different partial test patterns are non-conflicting if they are for different (pseudo) inputs, or if they are for the same (pseudo) input but have a non-conflicting value. Non-conflicting values are the same logic values, or different logic values one of which is X (don't care).

The number of partial test patterns required to test a logic cone is an outcome of the logic design process (and of the effectiveness of the ATPG tool used, although all ATPG tools in the market are very effective in this, as it is their kernel operation). The number of partial test patterns per cone depends on the width, depth, and exact structure of the logic cone. As logic design is under many constraints, the number of partial test patterns is typically not a number which is optimized for during the logic design phase, but rather an outcome that is simply accepted. Consequently, we observe quite a variation of partial test pattern counts for the various logic cones in a circuit.

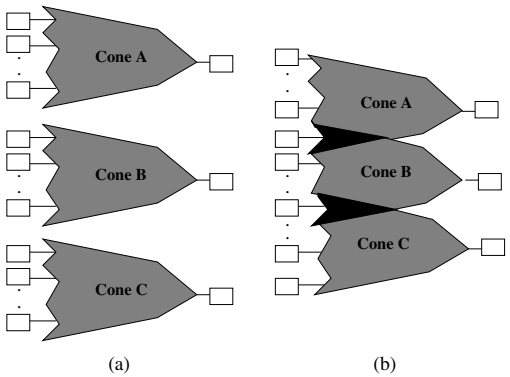

Figure 1: Cone structure of a design.

In the (rare) case of completely non-overlapping cones, partial test patterns for these cones can always be merged, as their stimulus bits come from disjunct (pseudo) inputs. The number of test patterns that need to be applied to the overall circuit is in that case the maximum of the numbers of partial test patterns for the constituting logic cones. This case is illustrated in Figure 1(a). The circuit consists of three non-overlapping logic cones, named Cone A, B, and C. Suppose Cones A, B, and C are driven by 20, 10, and 20 scan flip-flops, respectively. Also, suppose Cones A, B, and C require 200, 300, and 400 partial test patterns, respectively. Perfect pattern compaction results in 400 patterns for the overall circuit; 200 of these patterns test all the cones simultaneously, 100 patterns test Cones B and C, and 100 patterns test only Cone C. Regardless of which cone is targeted, each pattern consists of $20 + 10 + 20 = 50$ stimulus bits, resulting in a total stimulus volume of $400 \times 50 = 20,000$ bits.

In the (common) case that logic cones are partially overlapping, the partial test patterns per cone cannot always be merged, as some of their stimulus bits might be conflicting. Consequently, the number of test patterns for the overall circuit might grow larger than the maximum of the numbers of partial test patterns for the constituting logic cones. This situation is depicted in Figure 1(b), in which Cones A, B, and C have a pairwise overlap.

Summarizing, the test data volume for an arbitrary circuit is inflated with don't care dummy bits due to two reasons.

1. A hard-to-test logic cone, which requires many test patterns, dictates the overall number of test patterns. However, each test pattern is applied for all (pseudo) inputs and (pseudo) outputs. Consequently, easily testable cones, which require only a small number of test patterns, nevertheless get exercised by *all* test patterns, even though they have been tested out for a long time already.

2. Overlapping logic cones make that compaction techniques cannot merge all test patterns, and hence that we end up with more test patterns than the strict maximum over all cones.

In a monolithic test approach, all logic cones of a (large) SOC are considered in one ATPG run. Consequently, we typically see a very large variation in the number of partial test patterns per logic cone, while all pattern counts get topped-off to the maximum number of partial test patterns over all cones or higher.

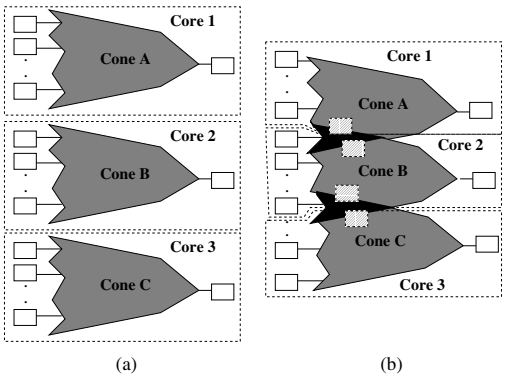

Figure 2: Design partitioned into cores.

In a modular, core-based test approach, the ATPG runs are per core. Each core is by definition only a fraction of the total SOC, and hence the variation in its partial test pattern counts between logic cones is always equal or smaller (and typically a lot less) then was the case for the entire SOC. In addition, there are fewer overlapping cones, as the overlaps at core boundaries are artificially removed by means of additional wrapper cells.

Ideally, every logic cone can be treated as a core to minimize the waste; this would not be a realistic approach, however, due to the area and data volume penalty imposed by wrapping these fine-grained cores. Only for the sake of illustration, however, let us consider the same example in Figure 1(a) partitioned into three cores as in Figure 2(a). In a modular SOC testing scheme, the test of Core 1 requires the shifting of Cone A partial patterns only into the scan cells in this cone; these partial patterns become the test patterns for Core 1. Similarly, Core 2 and Core 3 stimuli consist of as many bits as the number of scan cells in Cones B and C, respectively; 600 stimuli (200 for Core 1 and 400 for Core 3) consist of 20 bits each, and 300 (for Core 2) stimuli consist of 10 bits. The overall stimuli volume equals $600 \times 20 + 300 \times 10 = 15,000$ bits, leading to a reduction of test data volume of 25% over monolithic testing.

Next, we illustrate the case of overlapping logic cones in Figure 1(b) partitioned into isolated cores as in Figure 2(b). Dedicated wrapper registers inserted on the boundary of logic cones helps shatter the interaction of these cones, enabling their independent testing. The necessity to control and observe the isolation cells in addition to the core scan cells increase the the number of bits in core patterns however.

Test data volume reduction offered by modular SOC testing degrades due to the penalty imposed by the isolation cells. Whether this degradation offsets the benefit of modular SOC testing is the fundamental question that this paper aims at addressing.

In our analysis, we assume that cores are wrapped by using dedicated cells on each core I/O. While such an isolation scheme ensures full isolation, it is nevertheless a *pessimistic* approach in terms of test data volume. The utilization of functional registers along with dedicated cells may lead to reduced test data volume penalty. Another source of *pessimism* in our analysis is our assumption about the number of patterns in a monolithic design. We assume that an SOC tested as a monolithic entity (with isolation logic ripped out) requires as many patterns as the maximum number of core patterns, while in reality the overlapping of logic cones may result in the application of a much larger number of patterns.

We exclude the impact of the scan chain organization or the test access mechanism from our analysis. Prior research [23, 13] has shown that these factors contribute to test data volume penalty in the form of idle test bits. In this work, we assume perfectly balanced scan chains in both monolithic and modular testing; idle bits incurred by imbalanced chains may slightly push the results in the direction of either strategy, rendering it difficult to predict the exact impact a priori. In this paper, the comparative analysis focuses on useful (non-idle) test data bits only.

4. TEST DATA VOLUME FORMULATION

In monolithic design testing, the test data volume that includes both the stimulus and the response volumes should be computed by also accounting for the I/Os of the whole design. Thus, the test data volume in monolithic test can be formulated as:

$$TDV_{\text{mono}} = (I_{\text{chip}} + O_{\text{chip}} + 2B_{\text{chip}} + 2S_{\text{chip}}) \cdot T_{\text{mono}} \quad (1)$$

In the formulation above, I, O, B, and S denote the number of inputs, outputs, bidirectional ports, and scan cells, respectively. T_{mono} represents the number of test patterns for the monolithic, flat design. The number of bidirectional ports and the number of scan cells are multiplied by a factor of two, as each one of them necessitates the insertion of a stimulus bit and the observation of a response bit.

Based on our observation that we have presented in Section 3, we can state that:

$$T_{\text{mono}} \geq max_i\{T_{\text{ith cone}}\} \quad (2)$$

An *optimistic* test data volume for the monolithic design test can thus be formulated as:

$$TDV^{\text{opt}}_{\text{mono}} = (I_{\text{chip}} + O_{\text{chip}} + 2B_{\text{chip}} + 2S_{\text{chip}}) \cdot max_i\{T_{\text{ith cone}}\} \quad (3)$$

In our test data volume formulations in modular SOC testing, we also include the hierarchical cores; the reader may refer to [6] and [24] for details regarding the test of hierarchical cores. When a parent core is being tested, its wrapper is configured in InTest mode while the wrapper of the child cores are all configured into ExTest mode. Thus, the parent core inputs and the child core outputs need to be controlled, while the parent core outputs and the child core inputs need to be observed. The internal scan cells of the parent core should also be controlled and observed. Test data volume of modular SOC testing can then be formulated as:

$$TDV_{\text{modular}} = \sum_{P \in \text{Cores}} T_P \cdot (2S_P + ISOCOST_P) \quad (4)$$

$$\text{with } ISOCOST_P = I_P + O_P + 2B_P + \sum_{C \in \text{Child(P)}} (I_C + O_C + 2B_C), \quad (5)$$

where $ISOCOST$ for a core denotes the per pattern penalty incurred by the dedicated wrapper cells that surround both the parent core (denoted by P in the equations) and the child cores (denoted by C in the equations).

In Figure 3, the SOC p34932 from the ITC02 [3] benchmarks is sketched. This SOC consists of hierarchical cores; at the top-level four cores exist, three of which embed other cores. Referring back to Equation 5, ISOCOST for Core 2 equals the sum of I/Os of Core 2 itself and I/Os of the embedded Cores 3 through 9.

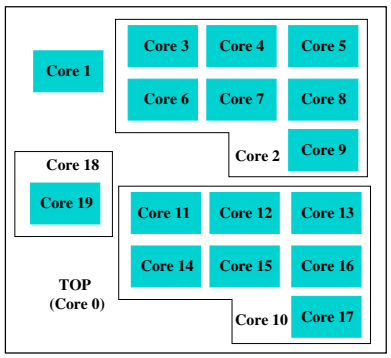

Figure 3: p34932 SOC from ITC02 benchmarks.

Next, we formulate TDV_{modular} by referring to T_{mono} as the base case, in order to explicitly state the penalty (due to isolation) and the benefit (due to varying pattern numbers) factors in modular testing. We utilize Equations 1 and 4 to derive the following equations:

$$TDV_{\text{modular}} = TDV_{\text{mono}} + TDV_{\text{penalty}} - TDV_{\text{benefit}} \quad (6)$$

$$\text{with } TDV_{\text{penalty}} = \sum_{A \in \text{Cores}} T_A \cdot ISOCOST_A \quad (7)$$

$$\text{and } TDV_{\text{benefit}} = \sum_{A \in \text{Cores}} (T_{\text{mono}} - T_A) \cdot 2S_A \quad (8)$$

In the equation above, $(T_{mono} - T_A)$ is guaranteed to be non-negative, as the number of monolithic test patterns is lower bounded by the number of patterns of each core as explained in the previous section.

5. EXPERIMENTAL RESULTS

In this section, we present experimental data in order to quantify the magnitude of test data volume reduction offered by modular SOC testing over monolithic, flattened testing.

5.1 Results on SOC1 and SOC2

We have constructed two SOCs by using ISCAS89 [2] benchmark circuits. The first SOC, namely SOC1, consists of five cores; these are s713, s953, and three instances of s1423 connected together as in Figure 4. We have generated all the test patterns, including the core test patterns and the monolithic flattened design patterns by using the same ATPG tool, ATALANTA [25], with identical parameters.

Table 1 provides information about the experiment conducted with SOC1. The number of inputs, outputs, scan cells, and test patterns are provided in Columns 2 through 5 for each SOC core, and the SOC top-level logic. By utilizing these parameters, the test data volume for each core and the SOC top-level logic is computed and reported in Column 6. The test data volume for the SOC is computed to be around 45K bits by adding up the individual core test data volumes. In the last two rows of the table, the test data volume is presented for the case wherein SOC1 is tested as a monolithic design with no isolation logic. In this case, 216 test patterns are generated by the ATPG tool for the monolithic design, leading to approximately 130K bits of test data volume. There are two conclusions to be drawn from this experiment. The first one is that Equation 2 holds; the number of test patterns for the monolithic design (216) is significantly more than the maximum number of patterns for the cores (85). Thus, optimistic test data volume for the monolithic design falls well below the actual one. The second conclusion is that modular SOC testing offers a test data volume reduction ratio of **2.87** ($= 129K/45K$) despite the additional bits to be shifted in and out of the wrapper cells; the associated test data volume penalty falls significantly below the test data volume benefit of modular testing, as can be observed right underneath the table. A pessimistically computed test data volume reduction ratio, however, happens to be 1.13 ($= 51K/45K$); the pessimism results in approximately 2.5x reduction from the actual ratio, which is 2.87.

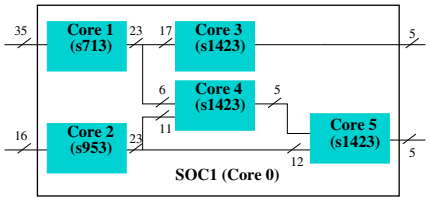

Figure 4: SOC1 constructed with ISCAS89 cores.

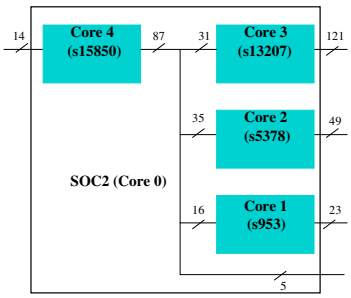

Figure 5: SOC2 constructed with ISCAS89 cores.

We have also created a second, larger SOC (SOC2) out of s953, s5378, s13207, and s15850 cores, as shown in Figure 5, and repeated the same experiment on this SOC. The results, which are provided in Table 2, are consistent with the ones in Table 1. The data again verifies the validity of Equation 2, as the number of test patterns for the monolithic design (945) is larger than the maximum number of core patterns (452). Modular SOC testing delivers a test data volume reduction ratio of **2.22** ($= 2.98M/1.34M$) for SOC2; the pessimistically computed ratio in this case is 1.06 ($1.43M/1.34M$), indicating a pessimism induced reduction factor of 2.1x.

5.2 Results on the ITC02 Benchmark SOCs

	I	*O*	*S*	*T*	*TDV*
Core 1 (s713)	35	23	19	52	4,992
Core 2 (s953)	16	23	29	85	8,245
Core 3-5 (s1423)	17	5	74	62	10,540
Core 0 = Top	51	10	0	2	326
SOC					45,183
Mono	51	10	270	216	129,816
Mono_opt	51	10	270	85	51,085

$TDV_{penalty}$ = 10,627 $TDV_{benefit}$ = 95,260

Table 1: Test data volume comparison for SOC1.

SOC	Cores	Norm. STDEV of Pattern Counts	TDV_{mono}^{opt}	$TDV_{penalty}$		$TDV_{benefit}$		$TDV_{modular}$	
d695	10	0.70	2,987,712	164,894	= +5.5%	1,935,953	= -64.8%	1,216,653	**= -59.3%**
h953	8	0.92	3,176,074	147,298	= +4.6%	1,121,480	= -35.3%	2,201,892	**= -30.7%**
f2126	4	0.68	11,812,624	400,418	= +3.4%	1,982,992	= -16.8%	10,230,050	**= -13.4%**
g1023	14	1.05	828,120	233,207	= +28.2%	479,124	= -57.9%	582,203	**= -29.7%**
g12710	4	0.18	34,140,348	16,223,802	= +47.5%	3,036,376	= -8.9%	47,327,774	**= +38.6%**
p22810	28	2.72	612,736,956	2,657,286	= +0.4%	601,177,672	= -98.1%	13,616,570	**= -97.7%**
p34392	19	1.29	522,738,000	4,991,278	= +9.5%	499,191,248	= -95.5%	28,538,030	**= -86.0%**
p93791	32	1.79	1,101,977,712	5,451,526	= +0.5%	1,060,719,663	= -96.3%	46,709,575	**= -95.8%**
t512505	31	0.93	459,196,200	4,293,188	= +0.9%	136,793,570	= -29.8%	326,695,818	**= -28.9%**
a586710	7	1.95	144,302,301,808	728,526,992	= +0.5%	144,080,555,088	= -99.8%	950,273,712	**= -99.3%**
Average					+10.1%		-60.3%		**-50.2%**

Table 4: Test data volume comparison for ITC02 SOC benchmarks.

To gauge the effectiveness of modular SOC testing on larger, industrial circuits, we have computed the test data volume for the ITC02 [3] benchmark SOCs. First, we illustrate this detailed computation for SOC p34932, which is shown in Figure 3.

Table 3 provides the test data volume computation for the SOC p34392. The first column denotes the core index, while the second column provides which other cores are embedded within the core, if the core is a hierarchical one. Columns 3 though 7 provide the number of inputs, outputs, bidirectional ports, scan cells, and test patterns for the core. The rightmost column denotes the test data volume for the cores; Equation 4 is utilized for this purpose. The rightmost entry of the final row provides the test data volume for this SOC tested in a modular manner.

We could only compute the *optimistic* test data volumes for the monolithic test, however, as the lack of netlist information for ITC02 SOCs prohibits ATPG execution and the computation of the actual number of test patterns for the flattened version of these SOCs. We have utilized our observation in Section 3, and the consequent Equation 3 for this purpose.

We have repeated the same computation for the other ITC02 benchmark SOCs, where only the core tests with TamUse=1 and ScanUse=1 have been considered. Table 4 provides these results; the first column denotes the benchmark SOC name, while the second column denotes the number of cores within the SOC. In the third column, we provide the normalized standard deviation of the core pattern counts for the SOC; it is computed by dividing the standard deviation by the average. The fourth column denotes the optimistic test data volume for the monolithic test of the flattened design, which is computed by utilizing Equation 3. The fifth column denotes the isolation penalty in bits (by Equation 7) and in percentage with respect to the monolithic test data volume given in Column 4, respectively. The sixth column provides similar data for the test data volume benefit (by Equation 8). Finally, Column 7 denotes the test data volume for modular SOC testing (by Equation 6), and the change in percentage compared to the test data volume of optimistic monolithic testing; a negative percentage value denotes a reduction in test data volume delivered by modular testing. It should be noted that the reduction delivered by modular testing can be much higher than those reported in the table, as the number of test patterns for the monolithic design can be much higher than the maximum number of test patterns among the cores. This has been verified by the data in Tables 1 and 2, wherein the pessimism factors have been calculated to be 2.5x and 2.1x, respectively. Despite this pessimism in the calculations, modular testing provides reduction in test data volume consistently for all the benchmarks except for g12710. This SOC consists of only four cores, each with approximately the same number of test patterns (852, 1314, 1223, 1223), resulting in an insignificant variation (0.18) in core pattern counts and thus in a small benefit value. Furthermore, as the total number of core I/Os exceeds the total number of scan cells in this SOC, a large penalty volume ensues. On the other extreme, the test data volume reduction is **99.3%** for the SOC a586710; in this SOC, a small core is tested with an extremely large number of patterns, resulting in a very large test data volume for the monolithic test, and thus a very large benefit obtained by modular testing.

Core	Embeds	I	O	B	S	T	TDV
0	1,2 and 18	32	27	114	0	27	39,069
1	-	15	94	0	806	210	361,410
2	3–9	165	263	0	8856	514	9,521,850
3	-	37	25	0	0	3108	192,696
4	-	38	25	0	0	6180	389,340
5	-	62	25	0	0	12336	1,073,232
6	-	11	8	0	0	1965	37,335
7	-	9	8	0	0	512	8,704
8	-	46	17	0	0	9930	625,590
9	-	41	33	0	0	228	16,872
10	11–17	129	207	0	4827	454	4,559,068
11	-	23	8	0	0	9285	287,835
12	-	7	4	0	0	173	1,903
13	-	12	16	0	0	2560	71,680
14	-	11	8	0	0	432	8,208
15	-	22	8	0	0	4440	133,200
16	-	7	7	0	0	128	1,792
17	-	15	4	0	0	786	14,934
18	19	175	212	0	6555	745	10,120,080
19	-	62	25	0	0	12336	1,073,232
SOC							28,538,030

Table 3: Test data volume computation for SOC p34392.

	I	O	S	T	TDV
Core 1 (s953)	16	23	29	85	8,245
Core 2 (s5378)	35	49	179	244	107,848
Core 3 (s13207)	31	121	669	452	673,480
Core 4 (s15850)	14	87	597	428	554,260
Core 0 = Top	14	198	0	2	752
SOC					1,344,585
Mono	14	198	1474	945	2,986,200
Mono_opt	14	198	1474	452	1,428,320

$TDV_{penalty}$ = 97,701 $TDV_{benefit}$ = 1,739,316

Table 2: Test data volume comparison for SOC2.

All the data presented in this section points to the test data volume reduction benefit of modular testing. We can see from Table 4 that the test data volume reduction of modular SOC testing is correlated to the normalized standard deviation of core pattern counts. The benchmark circuits g12710 and a586710 constitute two extremal points.

6. CONCLUSION

While modular SOC testing provides numerous benefits, such as test time, and power reduction, and addresses challenges faced in monolithic testing, such as timing closure, and ATPG tool capacity limitations, in this paper, we focus our attention on the test data volume reduction benefit of modular SOC testing. We present a comparative theoretical study backed up with quantitative results to gauge the test data volume reduction attained by modular testing over the monolithic test of a flattened design.

While core isolation inflates test data volume slightly due to the additional test data to be delivered to and collected from the isolating wrapper cells, such a mechanism helps break the inter-core dependencies, paving the way for the capability to test cores independently. This way, the degrading impact of pattern count variation is limited to within the core, minimizing the waste in test data volume.

To quantify the aforementioned reductions, we have conducted numerous experiments. We have not only created SOCs out of ISCAS89 benchmark circuits treated as cores, but also utilized ITC02 SOC benchmarks in our experiments as well. Despite the pessimism in our analysis, the results validate the test data volume reduction benefit of modular SOC testing over monolithic design testing.

REFERENCES

[1] Yervant Zorian, Erik Jan Marinissen and Sujit Dey, "Testing Embedded-Core Based System Chips", in *Proceedings IEEE International Test Conference (ITC)*, pp. 130–143, Washington, DC, USA, October 1998.

[2] Franc Brglez, David Bryan and Krzystof Kozminski, "Combinational Profiles of Sequential Benchmark Circuits", *Proceedings International Symposium on Circuits and Systems (ISCAS)*, vol. 14, n. 2, pp. 1929–1934, May 1989.

[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)*, pp. 519–528, Baltimore, MD, USA, October 2002.

[4] Erik Jan Marinissen and Yervant Zorian, "Challenges in Testing Core-Based System ICs", *IEEE Communications Magazine*, vol. 37, n. 6, pp. 104–109, June 1999.

[5] 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)*, pp. 770–777, Atlantic City, NJ, USA, October 2000.

[6] Anuja Sehgal, Sandeep Kumar Goel, Erik Jan Marinissen and Krishnendu Chakrabarty, "IEEE P1500-Compliant Test Wrapper Design for Hierarchical Cores", in *Proceedings IEEE International Test Conference (ITC)*, pp. 1203–1212, Charlotte, NC, USA, October 2004.

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

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

[9] Francisco DaSilva, editor, *IEEE Std 1500™-2005, IEEE Standard Testability Method for Embedded Core-based Integrated Circuits*, IEEE, New York, NY, USA, August 2005.

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

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

[12] Joep Aerts and Erik Jan Marinissen, "Scan Chain Design for Test Time Reduction in Core-Based ICs", in *Proceedings IEEE International Test Conference (ITC)*, pp. 448–457, Washington, DC, USA, October 1998.

[13] Sandeep Kumar Goel and Erik Jan Marinissen, "SOC Test Architecture Design for Efficient Utilization of Test Bandwidth", *ACM Transactions on Design Automation of Electronic Systems*, vol. 8, n. 4, pp. 399–429, October 2003.

[14] Vikram Iyengar, Krishnendu Chakrabarty and Erik Jan Marinissen, "Efficient Wrapper/TAM Co-Optimization for Large SOCs", in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 491–498, Paris, France, March 2002.

[15] Erik Larsson, "Integrating Core Selection in the SOC Test Solution Design-Flow", in *Proceedings IEEE International Test Conference (ITC)*, pp. 1349–1358, Charlotte, NC, USA, October 2004.

[16] Urban Ingelsson, Sandeep Kumar Goel, Erik Larsson and Erik Jan Marinissen, "Test Scheduling for Modular SOCs in an Abort-on-Fail Environment", in *Proceedings IEEE European Test Symposium (ETS)*, pp. 8–13, Tallinn, Estonia, May 2005.

[17] Vikram Iyengar and Krishnendu Chakrabarty, "Precedence-Based, Preemptive, and Power-Constrained Test Scheduling for System-on-a-Chip", in *Proceedings IEEE VLSI Test Symposium (VTS)*, pp. 368–374, Marina del Rey, CA, USA, May 2001.

[18] Erik Larsson and Zebo Peng, "Test Scheduling and Scan-Chain Division Under Power Constraint", in *Proceedings IEEE Asian Test Symposium (ATS)*, pp. 259–264, Kyoto, Japan, November 2001.

[19] Vikram Iyengar, Krishnendu Chakrabarty and Erik Jan Marinissen, "Recent Advances in Test Planning for Modular Testing of Core-Based SOCs", in *Proceedings IEEE Asian Test Symposium (ATS)*, pp. 320–325, Tamuning, Guam, USA, November 2002.

[20] Anuja Sehgal, Jeff Fitzgerald and Jeff Rearick, "Test Cost Reduction for the AMD™ Athlon Processor using Test Partitioning", in *Proceedings IEEE International Test Conference (ITC)*, Santa Clara, CA, USA, October 2007.

[21] Tom Waayers, Richard Morren and Roberto Grandi, "Definition of a Robust Modular SOC Test Architecture; Resurrection of the Single TAM Daisy-chain", in *Proceedings IEEE International Test Conference (ITC)*, page Digital Object Identifier 10.1109/TEST.2005.1584022, Austin, Texas, USA, October 2005.

[22] Ozgur Sinanoglu and Tsvetomir Petrov, "A Non-Intrusive Isolation Approach for Soft Cores", in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 27–32, Nice, France, April 2007.

[23] Erik Jan Marinissen and Sandeep Kumar Goel, "Analysis of Test Bandwidth Utilization in Test Bus and TestRail Architectures for SOCs", in *Proceedings IEEE Design and Diagnostics of Electronic Circuits and Systems Workshop (DDECS)*, pp. 52–60, Brno, Czech Republic, April 2002.

[24] Anuja Sehgal, Sandeep Kumar Goel, Erik Jan Marinissen and Krishnendu Chakrabarty, "Hierarchy-Aware and Area-Efficient Test Infrastructure Design for Core-Based System Chips", in *Proceedings Design, Automation, and Test in Europe (DATE)*, pp. 285–290, Munich, Germany, March 2006.

[25] H. K. Lee D. S. Ha, "Technical Report: On the Generation of Test Patterns for Combinational Circuits", Department of Electrical Eng., Virginia Polytechnic Institute and State University, December 1993.