SOC Test Planning Using Virtual Test Access Architectures

Anuja Sehgal, *Member, IEEE*, Vikram Iyengar, *Member, IEEE*, and Krishnendu Chakrabarty, *Senior Member, IEEE*

***Abstract*—Recent advances in tester technology have led to automatic test equipment (ATE) that can operate at up to gigahertz speeds. However, system-on-chip (SOC) scan chains are typically run at lower frequencies, e.g., 10–50 MHz. The use of high-speed ATE channels to drive slower scan chains leads to an underutilization of resources, thereby resulting in an increase in SOC testing time. We present a new test planning technique to reduce the testing time and test cost by matching high-speed ATE channels to slower scan chains using the concept of virtual test access architectures. We also present a new test access mechanism (TAM) optimization framework based on Lagrange multipliers and analyze the impact of virtual TAMs on the overall SOC test power consumption for one of the ITC'02 benchmarks. Experimental results for TAM optimization based on Lagrange multipliers and virtual TAMs are presented for three industrial circuits from the set of ITC'02 SOC test benchmarks.**

***Index Terms*—Lagrange multipliers, system-on-chip (SOC) testing, test access mechanisms (TAMs), test scheduling, testing time, virtual TAMs.**

I. Introduction

THE widespread use of embedded cores in system-on-chip (SOC) design has led to higher chip densities and shorter design cycle times. However, the growing demand for automatic test equipment (ATE) resources during manufacturing test of SOCs has led to a sharp increase in test cost [28]. Test cost for large SOCs can be viewed as consisting of:

1) **Explicit test cost** (Cost of investing in a new ATE, also known as *Capital Expenditure*): Complex cores often require expensive ATE resources such as high-frequency channels, high pin counts, large memory depths as well as special features for analog and RF cores [21]. As a result, older-generation ATE are often inadequate and large investments in new ATE must be made.
2) **Implicit test cost**: Large SOCs require long test sequences to guarantee high levels of fault coverage for embedded cores. This has led to an increase in testing time during which the SOC sits on an expensive ATE, thereby preventing other SOCs from being tested. This in turn leads to increased time-to-market and decreased profitability.

Manuscript received November 30, 2003; revised March 11, 2004. This work was supported in part by the National Science Foundation under Grants CCR-9875324 and CCR-0204077. This paper was presented in part at the IEEE/ACM Design Automation Conference, Anaheim, CA, June 2003.

A. Sehgal and K. Chakrabarty are with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA (e-mail: as@ee.duke.edu).

V. Iyengar is with IBM Microelectronics, Essex Junction, VT 05452 USA.

Digital Object Identifier 10.1109/TVLSI.2004.834228

As a result of rising costs, test is increasingly being viewed as a major bottleneck in SOC design and manufacturing; it is therefore important to reduce both explicit and implicit test cost through SOC test planning.

The reduction of *explicit* test cost requires that an existing amortized-cost ATE be used instead of investing in a new, expensive ATE. Methods proposed to match SOC test requirements to current ATE capabilities include test data compression [15], response compaction [23], [25], and reduced pin-count test [29]. All these methods seek to ensure that the SOC test can be handled by the existing ATE. However, current growth trends in SOC functionality and test requirements predict that future investment in newer and expensive ATE is inevitable [10].

On the other hand, reduction of *implicit* test cost requires that once a new, expensive ATE has been purchased, its resources must be utilized as efficiently as possible. If an investment in an expensive ATE has already been made, e.g, for at-speed functional test or high-speed I/O test, it is more cost effective to reduce the implicit cost by utilizing the available resources, compared to using older generation ATEs that yield higher SOC test times. This mandates that SOC testing times must be minimized such that several SOCs can use the ATE in a short time, and that the high-frequency data channels and pin-count resources of the ATE are properly utilized by each SOC.

Modular testing of SOCs is being increasingly used to simplify test access, ease test application, and reduce testing time [30]. To facilitate modular test, an embedded core must be isolated from surrounding logic, and test access must be provided from the I/O pins of the SOC. Test wrappers are used to isolate the core, while a test access architecture, also referred to as a test access mechanism (TAM) is used to transport test patterns from a pattern source to a core-under-test, and test responses from a core-under-test to a response sink. A number of test access architectures have been proposed in the literature [30]. Test wrapper design and TAM optimization are important during system integration because they directly impact hardware overhead, testing time and tester data volume on the ATE. Fig. 1 illustrates SOC test access based on wrappers and TAMs.

Methods to increase the efficiency of ATE use include test scheduling, TAM optimization, and multi-site test. Test scheduling seeks to obtain an effective ordering of tests applied to the SOC to minimize testing time [4], [13]. TAM optimization is performed to improve test access to embedded cores in a modular test environment [6], [7], [11], [17], [18], [24]. Finally, multisite test seeks to test several copies of the SOC simultaneously on the ATE, thus, reducing testing time across an entire production batch [28]. While these test planning methods increase the

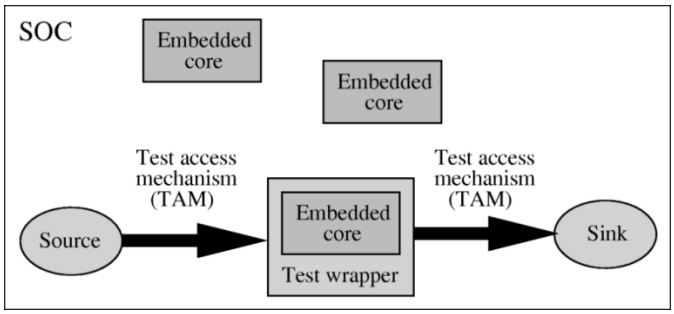


Fig. 1. SOC testing using test wrappers and TAMs [14].

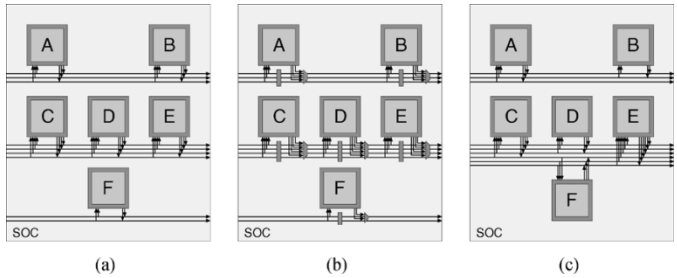


Fig. 2. Illustration of: (a) fixed-width Test Bus architecture; (b) fixed-width TestRail architecture; and (c) flexible-width Test Bus architecture [14].

efficiency of ATE-use, they assume that the ATE always operates at core scan clock frequencies. Scan chains are typically run at frequencies lower than 50 MHz to reduce power consumption and avoid time-consuming high-frequency scan design [26]. However, recent advancements in tester design have led to ATEs that can operate at up to several hundred megahertz [16]. The use of such high-frequency ATE channels at low scan clock frequencies severely under-utilizes ATE capability, resulting in an increase in testing time and time-to-market, thereby directly impacting implicit test cost.

The primary contribution of this paper lies in a new technique to reduce implicit test cost by matching ATE channel frequencies to core scan clock frequencies using virtual TAMs. A *virtual TAM* is an on-chip test data transport mechanism that does not directly correspond to a particular ATE channel. The number of virtual TAM wires is greater than the number of ATE channels; this is in contrast to conventional TAM wires that have a one-to-one correspondence with ATE channels. Virtual TAMs operate at scan chain frequencies; however, they interface with the higher-frequency ATE channels using bandwidth matching. Moreover, since the virtual TAM width is not limited by the ATE pin count, a larger number of TAM wires can be used on the SOC. This significantly increases the utilization of ATE capabilities and provides the SOC with a larger amount of test data in a shorter testing time. We study the impact of using virtual TAMs on SOC power consumption and present results for one of the ITC'02 benchmarks [22].

A secondary contribution of this paper lies in a new method for TAM optimization that leads to the efficient transport of test data from ATE channels to core I/Os. The new method based on Lagrange multipliers [19] exploits the monotonically nonincreasing function of core testing time with TAM width to effectively partition the set of virtual TAM wires into Test Buses that can be used to access embedded cores.

The rest of the paper is organized as follows. In Section II, we review various TAM optimization techniques. In Section III, we introduce the concept of virtual TAMs and derive a lower bound on the SOC testing time. In Section IV, we discuss the selection criteria for ATE parameters that impact the SOC test time. In Section V, we discuss the use of Lagrange multipliers for TAM width partitioning. In Section VI, we present the new TAM optimization flow using a combination of Lagrange multipliers for TAM width partitioning and a heuristic method for core assignment to TAMs. In Section VII, we present experimental results for benchmark SOCs demonstrating the applicability of our methods. In Section VIII, we examine the impact of virtual TAMs on SOC test power consumption. A larger number of TAM partitions, due to the use of virtual TAMs, can lead to greater test parallelism and higher test power. Hence, the reduction in testing time must be carefully balanced with the increase in test power. Section IX concludes the paper.

II. Review of TAM Optimization Techniques

Driven by the need to reduce test cost, researchers have presented a number of TAM optimization techniques in the literature. Dedicated and scalable TAMs such as *Test Bus* [27] and *TestRail* [20] architectures appear to be the most common. Fig. 2 illustrates the various TAM architectures.

In the Test Bus architecture shown in Fig. 2(a), only one core can be accessed at a time on each Test Bus [27]. As a result, cores connected to the same Test Bus can only be tested sequentially. This architecture allows for multiple Test Buses on an SOC that can operate independently. In the TestRail architecture [20] shown in Fig. 2(b), scan-testable cores connected to the same TestRail can be tested simultaneously as well as sequentially. A TestRail architecture allows for multiple TestRails on an SOC, which operate independently.

In most TAM architectures, the cores assigned to a TAM partition are connected to all the wires of that TAM partition. These designs are referred to as *fixed-width* TAMs. A generalization of this design is one in which the cores assigned to a TAM partition connect to (possibly different) subsets of the TAM wires [13]. The core/TAM connections are made at the granularity of TAM wires, instead of considering the entire TAM bundle as one inseparable entity. These access mechanisms are called *flexible-width* TAMs. Fig. 2(c) shows an example of a flexible-width Test Bus architecture.

In this paper, we focus on a Test Bus architecture in which the embedded cores have fixed-length scan chains. The TAM optimization problem is known to be $\mathcal{N}P$-hard [3]; hence, heuristic techniques are needed in practice. In this section, we review several TAM optimization methods that have been proposed recently and that are used as a basis for comparison in this paper.

In [11], the authors optimized a Test Bus architecture using a combination of integer linear programming (ILP) and exhaustive enumeration. Given an SOC with N cores, the optimization problem ($\mathcal{P}_{\mathrm{NPAW}}$) in [11] is formulated as follows: determine 1) the number B of TAM partitions for the SOC; 2) a partition of the total TAM width W among the TAMs; 3) an assignment

of the N cores to TAMs; and 4) a wrapper design for each core, such that SOC testing time is minimized.

The problem of wrapper design was solved using the *Design_wrapper* algorithm [11] based on Best Fit Decreasing heuristic. The core assignment problem was solved using an ILP model formulated as follows.

Consider an SOC consisting of N cores and B TAMs of widths $w_1, w_2, \ldots, w_B$. The time taken to test Core i assigned to TAM j is denoted by $T_i(w_j)$ clock cycles. Let x_{ij} be a binary variable, which is defined as follows:

$$x_{ij} = \begin{cases} 1, & \text{if Core } i \text{ is assigned to TAM } j \\ 0, & \text{otherwise.} \end{cases}$$

The time taken to test all cores on TAM j is given by $\sum_{i=1}^{N} T_i(w_j).x_{ij}$. Since all the TAMs can be used simultaneously for testing, the system testing time equals $\max_{1 \le j \le B} \sum_{i}^{N} T_i(w_j).x_{ij}$. The ILP model for core assignment can be formulated as follows.

Objective: minimize testing time $\mathcal{T}$, subject to

1) $\mathcal{T} \ge \sum_{i=1}^{N} T_i(w_j).x_{ij}, 1 \le j \le B$, i.e., $\mathcal{T}$ is the maximum testing time on any TAM;
2) $\sum_{j=1}^{B} x_{ij} = 1, 1 \le i \le N$, i.e., every core is assigned to exactly one TAM.

It was observed in [11] that the ILP model for core assignment can be solved in reasonable time for a small number of TAM partitions. Hence, a brute-force method referred to as the ILP/enumerate approach was used to exhaustively enumerate TAM partitions. A major drawback of this brute-force method is that it does not scale for $B > 3$. The execution time increased exponentially with B and thus the algorithm failed to yield efficient results for $B > 3$.

The optimization approach of [11] was extended in [12] to include a heuristic method for core assignment. The heuristic approach in [12] uses three steps for TAM optimization. In the first step, a heuristic algorithm called *Core_assign* is used for assigning cores to TAMs. In the second step, a procedure termed *Partition_evaluate* is used to enumerate and evaluate a large number of TAM partitions. A solution space pruning technique is used to limit the number of unique partitions evaluated, thereby ensuring feasible execution times for large problem instances. As a final optimization step, the ILP model is used only once to improve the solution. The heuristic core assignment procedure *Core_assign* from [12] forms a part of the TAM optimization method presented in this paper.

In [13], the authors presented a method to integrate TAM design and test scheduling using rectangle packing. They introduced the notion of flexible-width Test Buses and used a fork-and-merge Test Bus architecture. In this approach, the embedded cores in an SOC are allowed to be connected to any subset of the top-level TAM wires, thereby improving the utilization of the TAM wires. However, a drawback of this approach is that it can potentially increase the complexity of physical design due to more complicated routing of TAM wires.

Finally, in [6] the authors presented a heuristic algorithm for TestRail optimization, which forms the basis of the TR-Architect tool. This approach is effective for both the TestRail and Test Bus architecture. However, it lacks flexibility in one important aspect—the optimization flow in TR-Architect as presented in [6] does not allow the user to limit the number of TAM partitions. The number of TAM partitions is determined by the various procedures that constitute the optimization flow. In many practical scenarios, it might be necessary to limit the number of TAM partitions in Test Bus architectures due to power consumption constraints.

III. Virtual TAMs

Recent advancements in ATE technology have led to a substantial increase in ATE channel frequencies. In addition, the "test processor-per-pin" architecture, as in the Agilent 93000 tester [1], allows the ATE to drive channels with different speeds and reach data rates of up to 2.5 Gbps. However, the frequency at which an embedded core can be tested is limited by its scan clock frequency, typically under 50 MHz [16]. For example, [26] reports results on low-power scan testing for an industrial circuit with scan clock frequency of 40 MHz. Scan clock frequencies are kept low to meet SOC power constraints and to avoid the design costs of high-frequency scan. The TAMs designed to transport test data to core scan chains, e.g., in [6], [7], and [11] are, therefore, constrained to operate at frequencies far lower than ATE channel capabilities. This reduces the utilization of ATE resources and increases testing time, thereby, increasing the implicit test cost.

The mismatch between ATE capabilities and TAM operating frequencies can be reduced using virtual TAMs based on bandwidth matching [16]. The on-chip TAM wires are of two kinds: 1) low-frequency TAM wires driven by low-frequency ATE pins and 2) low-frequency TAM wires driven by high-frequency ATE pins.

We apply bandwidth matching at the interface between SOC-internal low-frequency TAMs and the set of ATE channels, which may include both high-frequency and low-frequency channels. Bandwidth matching leads to an increase in the number of TAM lines available within the SOC for core testing. The number of ATE channels, including low-frequency and high-frequency channels, used for SOC testing is less than the TAM width present internally for core testing after bandwidth matching is applied. Since the increase in the number of internal TAM lines is not visible at the ATE–SOC interface, we term the resulting internal TAM as the virtual TAM. The virtual TAM can adapt any of the TAM architectures described in Section II for core-internal testing. In this paper, we use a Test Bus TAM architecture.

Virtual TAMs are based on the following relationship between the TAM width and operating frequency of test data transport mechanisms

$$W_{\mathrm{ATE}} \times f_{\mathrm{ATE}} = W_{\mathrm{TAM}} \times f_{\mathrm{TAM}} \tag{1}$$

where W_{ATE} and W_{TAM} are the total ATE channel width and the total SOC TAM width, respectively, and f_{ATE} and f_{TAM} are the ATE channel and virtual TAM frequencies respectively. If bandwidth matching is not used, W_{TAM} equals W_{ATE}, and all the low-frequency and high-frequency ATE pins operate at the lower f_{TAM} frequency.

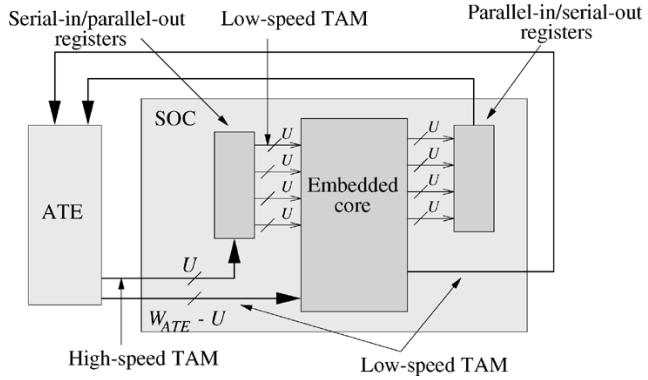


Fig. 3. Virtual TAMs based on bandwidth matching.

In order to minimize the the testing time by using the high-frequency ATE pins, yet not violating the scan frequency constraint of the cores, we increase the available TAM width and decrease the frequency of high-speed TAMs by the same factor n, such that (1) is satisfied. This is illustrated as follows.

Given an SOC TAM of $W_{\rm ATE}$ pins (driven by the ATE), of which U pins are driven at the higher frequency $f_{\rm ATE}$ and $(W_{\rm ATE}-U)$ pins are driven at the lower scan frequency $f_{\rm TAM}$, such that $f_{\rm ATE}=n\times f_{\rm TAM}$ using frequency division and bandwidth matching, the following relationship holds:

$$U\times f_{\rm ATE}+(W_{\rm ATE}-U)\times f_{\rm TAM} = U\times n\times f_{\rm TAM}+(W_{\rm ATE}-U)\times f_{\rm TAM}. \quad (2)$$

Therefore, the total number of pins available to the SOC for core testing, defined as the virtual TAM width, is given by

$$W=n\times U+(W_{\rm ATE}-U)=(n-1)U+W_{\rm ATE}. \quad (3)$$

Thus, every ATE pin operating at the higher frequency gives rise to $n-1$ virtual TAM pins. The virtual TAMs decrease testing time significantly since they provide a larger bandwidth of test data to the embedded cores.

Virtual TAMs are implemented on-chip using serial-in/parallel-out registers at the inputs of the embedded cores and parallel-in/serial-out registers at the outputs of the cores. As seen in Fig. 3, the low frequency data channels are connected directly to an embedded core and each high-frequency pin is fed through a 1-to-n ($n=4$ in this example) serial-in/parallel-out register. Similarly, parallel-in/serial-out registers used at the chip outputs are used to ensure that increased test parallelism is achieved without increasing the number of I/O channels. The increased parallelism reduces the overall SOC testing time. Moreover, since the serial-in/parallel-out interfaces used for implementation are placed next to the cores, only the original $W_{\rm ATE}$ TAM wires are routed through the system. Thus, a large number of TAM wires can be obtained with low routing and hardware cost.

IV. Selection of Values for the Parameters U and n

The testing time of an SOC is often dominated by the testing time of bottleneck embedded cores. The testing time for a bottleneck core reaches its minimum value at a particular TAM width $\widetilde{W}$, and it does not decrease any further for TAM widths greater than $\widetilde{W}$. In such situations, the SOC testing time also does not decrease any further for TAM widths greater than a particular TAM width, and it levels off at a corresponding lower bound value of $T^\star$ [3]. The presence of bottleneck cores in an SOC and the resulting lower bound on the testing time greatly influence the choice of U and n for a virtual TAM architecture. In this section, we discuss how appropriate values of U and n can be determined.

TABLE I
TAM Width $W^\star$ and Lower Bound on Testing Time $T^\star$ due to Bottleneck Cores for a Number of ITC'02 Benchmark SOCs

SOC	$W^\star$ (bits)	$T^\star$ (clock cycles)
u226	48	5333
d281	48	3926
g1023	40	14794
p34392	36	544579
t512505	36	5228420
h953	16	119357
f2126	16	335334
q12710	16	2222349

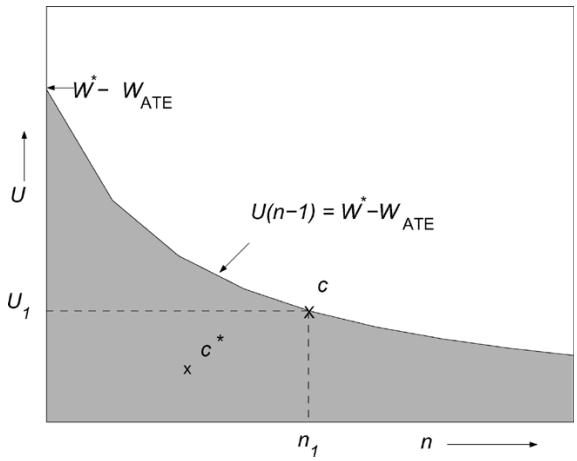


Fig. 4. The variation of U versus n for a given $W_{\rm ATE}$ when W is limited by $W^\star$ (figure not drawn to scale).

Let $W^\star$ denote the TAM width at which the SOC testing time reaches a lower bound due to a bottleneck core. Table I lists the values of $T^\star$ for several ITC'02 benchmark SOCs. For a given value of $W_{\rm ATE}$, values of U and n that result in a virtual TAM width greater than $W^\star$ lead to excess on-chip TAM wires without providing any reduction in testing time. Recall that $W=nU+(W_{\rm ATE}-U)$. To ensure that $W\le W^\star$, the following relationship must hold:

$$W^\star\ge nU+(W_{\rm ATE}-U)$$

which implies that

$$U(n-1)\le W^\star-W_{\rm ATE}. \quad (4)$$

Since $W^\star-W_{\rm ATE}$ is constant, the inequality (4) represents a rectangular hyperbola when U is plotted against n, and the desired values of U and n lie in the shaded region below the curve in Fig. 4. (This generic figure is not specific to an SOC.)

From Fig. 4, we see that if U and n are chosen such that the corresponding point $c^\star$ in the graph lies below the curve, hence

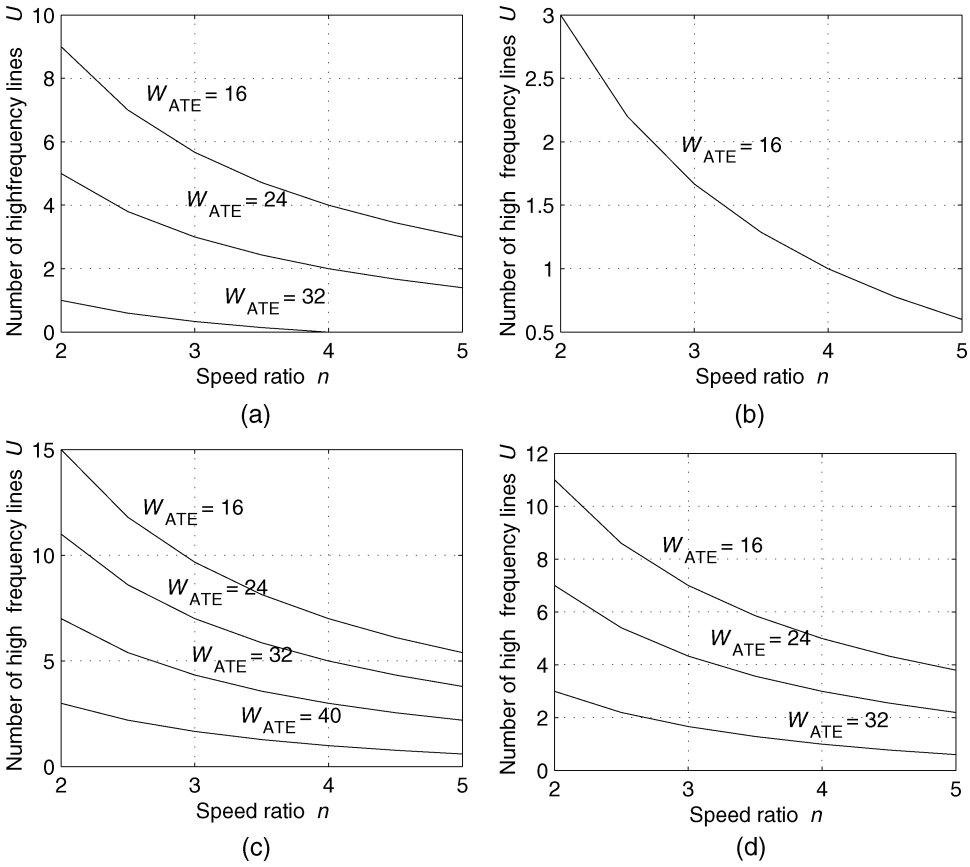


Fig. 5. (a) U versus n for SOCs p34392 and t512 505 ($W^\star = 36$), (b) U versus n for SOCs h953, f2126, and q12710 ($W^\star = 16$), (c) U versus n for SOCs U226 and d281 ($W^\star = 48$), and (d) U versus n for SOC g1023 ($W^\star = 40$).

the total virtual TAM width W is less than $W^\star$. In this way, the on-chip TAM wires are fully utilized, since every TAM wire contributes to the reduction of the overall test time of the SOC. However, since $c^\star$ lies below the curve, it yields a testing time that is greater than the lower bound. On the other hand, if the choice of U and n are made such that $U = U_1$, $n = n_1$, and the corresponding point c lies exactly on the curve, the testing time of the SOC equals the testing time of the bottleneck core, which is the lower bound on the testing time for the SOC. If n is constrained by ATE limitations, the inequality (4) yields the value of U that achieves the testing time lower bound. Similarly, if U is constrained, n can be chosen accordingly.

Fig. 5 shows the variation of U versus n when W is limited by $W^\star$ for the ITC'02 benchmark SOCs with bottleneck cores. It is seen that for a larger W_{ATE} value, the number of feasible values for U and n decrease. However, by using the relationship between U, n, W_{ATE}, and $W^\star$, we can choose a (U, n) pair such that the virtual TAM width is efficiently utilized.

A. Lower Bound on Testing Time

We next derive a lower bound on the testing time when W TAM wires are derived from W_{ATE} ATE channels via bandwidth matching. This lower bound is especially useful for SOCs that do not contain bottleneck cores. In our lower bound and in our test scheduling approach, we do not allow overlap of the scan-out operation for the last test pattern of a core with the scan-in operation for the first test pattern of the next core on the same TAM partition.[1] While this is feasible for a TestRail architecture as in [6], it is difficult to implement for a Test Bus architecture. Moreover, since the overlap can exist only for *one* test pattern for every core, the saving in testing time due to the overlap is less than 0.1% on average if the average number of test patterns per core is 1000, and this percentage is even lower for a larger number of patterns.

In order to derive the lower bound using a geometric argument, we use a rectangle representation for the core tests as in [13]. The testing times for a core in the SOC can be represented using a set of rectangles. A set $\mathbf{R}_i$ of rectangles for Core i ($1 \leq i \leq N$, where N is the number of cores in the SOC) is determined such that the height and width of each rectangle correspond to a TAM width and the corresponding test application time for the core, respectively. The TAM optimization problem can now be formulated in terms of rectangle packing as follows: Select one rectangle from each set $\mathbf{R}_i, 1 \leq i \leq N$, and pack the selected rectangles into a bin of fixed height, such that no two rectangles overlap, and the width to which the bin is filled is minimized.

The area of a bin, with the width representing total testing time T and the height representing total TAM width W, is given by $T \times W$. Each core yields a set of rectangles of different areas. Let $\mathcal{R}_i^{\min} \subseteq \mathbf{R}_i$ be the area of the minimum-area rectangle for

[1]This restriction should not be confused with the notion of overlap in scan testing in which the scan in of a test pattern for a core is overlapped with the scan out of the previous pattern for that core. Such an overlap is indeed permitted here.

Core i. Let the area of a rectangle representing Core i being tested at TAM width w given by $R_i(w) = T_i(w) \times w$, where $T_i(w)$ is the testing time of Core i on TAM width w. It follows that $\mathcal{R}_i{}^{\min} = \min_i\{R_i(w)\}, 1 \leq w \leq W$. We next show that the minimum-area rectangle for each core is a rectangle of height 1, i.e., $\mathcal{R}_i{}^{\min} = R_i(1)$.

A lower bound on the testing time $T_i(w)$ of Core i on a TAM partition of width w can be expressed as

$$T_i(w) \geq \left\lceil \frac{\max(ts_i, tr_i) \cdot p_i + \min(ts_i, tr_i)}{w} \right\rceil + p_i$$

where ts_i is the number of test bits to be scanned into core i, tr_i is the number of test bits to be scanned out of core i, and p_i is the number of test patterns to be applied to Core i. The numerator of the first term on the right-hand side of the above inequality represents the total test data volume to be applied to the core; it is independent of the number of TAM wires used to apply the test data. Hence for any core i, $R_i(w) = (\lceil v/w \rceil + p_i) \times w$ and $R_i(1) = (v + p_i)$, where v is the total test data volume for the core. Comparing the expressions for $R_i(w)$ and $R_i(1)$ for $w > 1$, we see that $(\lceil v/w \rceil) \times w \geq v$ and $p_i \times w > p_i$. Thus

$$R_i(w) > R_i(1)\ \forall w > 1.$$

We also note that the minimum-area rectangles for the cores might not fill the bin of area $T \times W$ perfectly owing to the variations in the sizes of the rectangles. As a result, there may be some unfilled space in the bin. Let us denote the total area of the unfilled space in the bin by Δ, where $\Delta \geq 0$. Now, we know that the total area of the bin cannot be less than the sum of the minimum-area rectangles of all the cores in the SOC and the sum of all the unfilled space in the bin. Thus

$$\begin{aligned} T \times W &\geq R_1(1) + R_2(1) + \cdots + R_N(1) + \Delta \\ &\geq R_1(1) + R_2(1) + \cdots + R_N(1) \end{aligned}$$

which implies that

$$T \geq \frac{\sum_{i=1}^{N} R_i(1)}{W} = \frac{\sum_{i=1}^{N} R_i(1)}{W_{\mathrm{ATE}} + (n-1)U}. \tag{5}$$

Let the lower bound obtained from (5) be denoted by LB_1. When compared to the lower bounds derived in [3] based on the notion of a bottleneck core, and as discussed earlier in this section, LB_1 is more accurate for TAM widths smaller than $W^\star$. However, for larger values of W ($W > W^\star$), the bounds derived in [3] are tighter. Hence, the overall lower bound LB_T is determined by taking the maximum of LB_1 and lower bound from [3]. Tables III and IV in Section VII show the lower bounds for various TAM widths.

V. Lagrange Multipliers

In this section, we introduce the proposed Lagrange framework for minimizing implicit SOC test cost. Implicit test cost is reflected in the SOC testing time, since testing time directly impacts the ATE time spent per SOC and contributes to test cost in real ($) terms. The SOC testing time is minimized by designing a virtual TAM architecture and optimizing the virtual TAM widths supplied to cores.

Lagrange multipliers can be applied to many constraint-driven optimization problems [19]. Consider the maximization problem

$$\text{Maximize}: \quad u\bar{X}, \ \text{subject to}\ a\bar{X} < b.$$

The problem of maximizing $u\bar{X}$, where $\bar{X}$ is a vector of variables and u, a, and b are constants, can be formulated as the minimization of cost function J defined as follows:

$$J = u\bar{X} - \lambda(a\bar{X} - b).$$

Here, λ is an integer, referred to as the Lagrange multiplier, whose value can be appropriately varied to minimize the cost function subject to the constraints.

We first describe a simple TAM optimization problem and discuss how it can be solved using Lagrange multiplier. We then formulate the general case. Consider an SOC with two TAMs ($B = 2$) and two cores ($N = 2$). Let B denote the number of TAMs and N denote the number of cores in the system. Let w_1 and w_2 be the widths of the two TAMs. We assume here that the core assignment to TAMs is determined *a priori*. (This constraint is relaxed in Section VI, where a method for integrated core assignment and TAM optimization is presented.) Core 1 is tested on TAM 1 and Core 2 is tested on TAM 2. Let the testing time of Core 1 on TAM 1 be denoted by $T_1(w_1)$, and the testing time of Core 2 on TAM 2 be denoted by $T_2(w_2)$. Note that $T_1(w_1)$ and $T_2(w_2)$ are both monotonically nonincreasing functions, as shown in [13]. We now solve the following optimization problem: determine the values of w_1, w_2, such that 1) $w_1 + w_2 = W$ and 2) $\max\{T_1(w_1), T_2(w_2)\}$ is minimized, where W denotes the total virtual TAM width available.

We rephrase this problem as the minimization of a Lagrange cost function [19]. Let the Lagrange cost function $\mathcal{J}(w_1, w_2)$ be defined as

$$\mathcal{J}(w_1, w_2) = \max\{T_1(w_1), T_2(w_2)\} + \lambda(w_1 + w_2) \tag{6}$$

where λ is referred as the Lagrange multiplier.

The theory of Lagrange multipliers shows that for every W, there exists a Lagrange multiplier λ such that the minimization of $\max\{T_1(w_1), T_2(w_2)\}$ is equivalent to the minimization of the right-hand expression in (6) [19]. Thus, instead of minimizing $\max\{T_1(w_1), T_2(w_2)\}$, we solve (6). Our goal is to devise an algorithm that determines the values of w_1 and w_2, such that $\mathcal{J}(w_1, w_2)$ is minimized for a given λ.

Next, we investigate the relationship between λ and W. We consider two corner cases to study the impact of λ on the value of W. Even though these abstract corner cases never arise in practice, they are presented here to highlight the relationship between λ and W.

Case 1) Let us minimize the expression for $\mathcal{J}(w_1, w_2)$ in (6) while setting λ to 0. If $\lambda = 0$, then $\mathcal{J}(w_1, w_2)\ |_{\lambda=0} = \max\{T_1(w_1), T_2(w_2)\}$. Hence, the penalty term $\lambda(w_1 + w_2)$ vanishes. Now, since both $T_1(w_1)$ and $T_2(w_2)$ are monotonically nonincreasing, $\mathcal{J}(w_1, w_2)\ |_{\lambda=0}$ is minimized when both

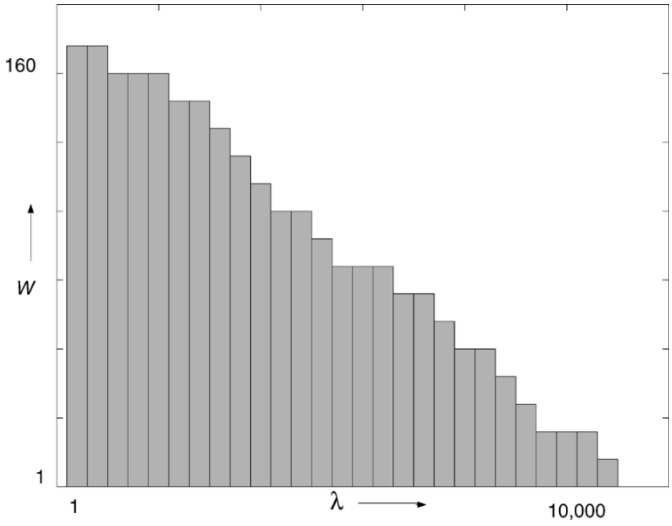


Fig. 6. Illustration of the relationship between TAM width W and the Lagrange multiplier λ in the minimization of cost function $\mathcal{J}$.

$w_1 \longrightarrow \infty$ and $w_2 \longrightarrow \infty$. Therefore, if λ is set to 0, $\mathcal{J}(w_1, w_2)$ is minimized by selecting a large value of W.

Case 2) Next, let us minimize the expression for $\mathcal{J}(w_1, w_2)$ while setting λ to a large value, i.e., $\lambda \longrightarrow \infty$. In this case, from (6), $\mathcal{J}(w_1, w_2) \approx \lambda(w_1 + w_2)$. The penalty term thus outweighs the max term in (6). Hence, to minimize $\mathcal{J}$ when λ is large, a small value of W must be chosen, i.e., $W \longrightarrow 0$.

From the above two cases, we note that if we vary the value of the Lagrange multiplier λ, an inversely proportional value of W in needed to minimize $\mathcal{J}$ (and equivalently, the SOC testing time cost function). The variation of W with λ is illustrated in Fig. 6.

Next, we formalize the problem for the general case of $B \geq 2$ TAMs and $N \geq 2$ cores. Recall that the core assignment to TAMs is predetermined. Let the constant $x_{ij} = 1$ ($1 \leq i \leq N, 1 \leq j \leq B$) denote that core i is assigned to TAM j, otherwise $x_{ij} = 0$. Generalizing (6) for N cores and B TAMs, we formulate the problem as follows. Determine the TAM widths $w_1, \ldots, w_B$, such that $\sum_{j=1}^{B} w_j = W$ and the cost function $\mathcal{J}(w_1, \ldots, w_B)$ is minimized, where

$$\mathcal{J}(w_1, \ldots, w_B) = \max_j \left\{ \sum_i^N T_i(w_j) x_{ij} \right\} + \lambda \sum_{j=1}^{B} w_j. \quad (7)$$

The expression $\max_j \left\{ \sum_i^N T_i(w_j) x_{ij} \right\}$ gives the maximum testing time over all TAM partitions. The penalty term is the product $\lambda \times W$, and λ varies inversely with W. For a given λ, a corresponding value of W is achieved that minimizes the cost function. It is possible to arrive at a desired value of W by using a bisection search over all possible values of λ. In our experiments, we have found that in order to find partitions for TAM widths varying from 8 to 160, the λ values need to vary from 10 000 to 1, as shown in Fig. 6. For example, for the SOC benchmark circuit p22810, $\lambda = 10000$ yields a TAM partition for a TAM width of $W = 8$.

A. Iterative Descent Procedure

Given a value of λ and the set of values for the x_{ij} variables, we solve (7) using an iterative descent procedure. This procedure optimizes the cost function $\mathcal{J}$ along each dimension j in a round-robin manner. Let $\mathbf{w}^{(0)} = \{w_i^{(0)} : 1 \leq i \leq B\}$ be the initial value of the solution vector, e.g., an arbitrary choice of equal TAM widths. In the first iteration, we keep the values of all w_i ($i \neq 1$) fixed at their initial values, i.e., $w_i^{(1)} = w_i^{(0)}$ for $i \neq 1$. We then optimize the cost function to determine the optimal value of w_1 for this constrained problem instance. Let w_1^* denote the optimal value of w_1. We set $w_1^{(1)} = w_1^*$. In the second iteration, keeping the values of all w_i ($i \neq 2$) constant, we optimize the cost function to determine the optimal value of w_2. In this manner, locally-optimal values for $w_1, \ldots, w_B$ are determined. The procedure then repeats to find the next value for w_1. The procedure cycles through each value of j, ending when the decrement in the cost function $\mathcal{J}$ goes below a given threshold ϵ. Also, it is reasonable to assume that TAM partitions in practice do not exceed a width of 64 [11]; thus all values of w_j are chosen such that $w_j \leq w_{\max}$, where $w_{\max} = 64$.

An important property of the above procedure is that the cost at the end of the n^{th} iteration is always less than or equal to the cost at the end of the $(n-1)^{\text{th}}$ iteration, i.e., $\mathcal{J}^{(n)} \leq \mathcal{J}^{(n-1)}$. We exploit this property to show that the procedure is guaranteed to converge. Note that $\mathcal{J}$ is bounded from below (a trivial lower bound is $\mathcal{J} \geq 0$). Also, from the property $\mathcal{J}^{(n)} \leq \mathcal{J}^{(n-1)}$, $\mathcal{J}^{(n)}$ is a monotonically nonincreasing function of n. Since a monotonically nonincreasing function that is bounded from below is guaranteed to converge, the iterative procedure is also guaranteed to converge.

B. Illustrative Example

We demonstrate the efficiency of the proposed method using a simple illustrative example. Let $N = 2$ and $B = 2$ as before. Let Core 1 be tested on TAM 1 and Core 2 on TAM 2. Furthermore, let $T_1(w_1) = 10e^{-w_1}$ and let $T_2(w_2) = 10e^{-2w_2}$. Note that both $T_1(w_1)$ and $T_2(w_2)$ are monotonically nonincreasing function of w_1 and w_2, respectively. Let $\lambda = 1$. We wish to minimize $\mathcal{J}(w_1, w_2)$, where

$$\mathcal{J}(w_1, w_2) = \max\{10e^{-w_1}, 10e^{-2w_2}\} + (w_1 + w_2). \quad (8)$$

Let the allowed values of w_1 and w_2 be constrained, such that $1 \leq w_1, w_2 \leq 10$. A brute-force solution would require the evaluation of $\mathcal{J}$ for all 100 possible combinations of w_1 and w_2. Such a brute-force search in this example gives $w_1^{opt} = 2$, $w_2^{opt} = 1$ and $J^{opt}(2,1) = 4.3534$. Next, we solve the problem using the proposed procedure. We initialize the TAM width vector to $w_1^{(0)} = w_2^{(0)} = 10$. Since $\lambda = 1$, $J^{(0)} = 20.0005$.

In the first iteration, we minimize $\mathcal{J}(w_1, w_2)$ varying only w_1, while keeping $w_2 = 10$. The constrained cost function can be expressed as

$$\mathcal{J}'(w_1) = \max\{10e^{-w_1}, 2 \times 10^{-8}\} + w_1 + 10. \quad (9)$$

Using the bisection search method [5], we find that the value $w_1 = 2$ minimizes the cost function in (9). Thus, $w_1^{(1)} = 2$, $w_2^{(1)} = 10$. After iteration 1, $J^{(1)} = 13.3534$. In Iteration 2, we set $w_1^{(2)}$ to 2, and minimize the cost function, while varying w_2. The new constrained cost function can thus be written as

$$\mathcal{J}'(w_2) = \max\{1.35, 10e^{-2w_2}\} + 2 + w_2. \qquad (10)$$

Here, bisection search [5] yields $w_2 = 1$, and the minimal value of the cost function $J^{(2)}$ equals 4.3534. Next, in Iteration 3, we fix w_2 to 1 and vary w_1. The solution obtained at the end of Iteration 3, remains unchanged. Thus, we have achieved the optimal values of w_1 and w_2. These are given by $w_1 = 2$, $w_2 = 1$. Recall that this solution is the same as the one we obtained earlier using brute-force search. However, we are able to find the optimal solution in only three iterations using the iterative descent procedure, as compared to 100 iterations using the brute-force search. Moreover, from the theory of Lagrange multipliers, the complexity of the proposed approach is linear in B, whereas that of the brute-force method is exponential in B.

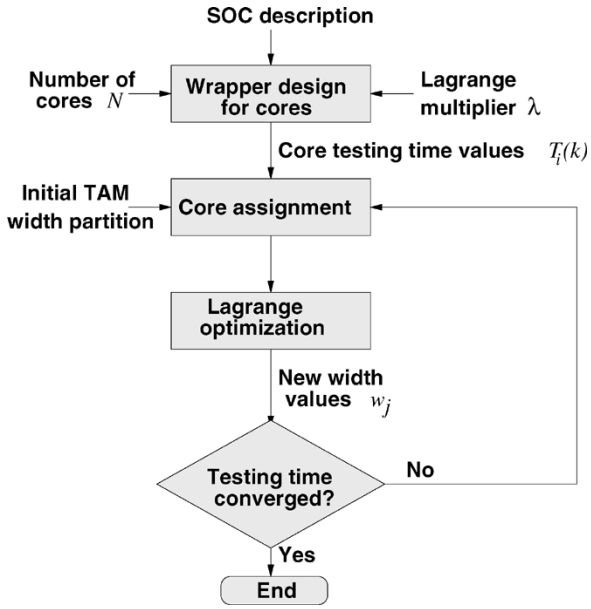


Fig. 7. Procedure for core assignment and TAM optimization.

VI. TAM Optimization and Core Assignment

In the previous section, we used Lagrange optimization to determine an optimal partition of TAM widths among cores when the core assignment to TAMs is known. In this section, we solve the more general problem of optimizing core assignments as well as TAM widths in conjunction. This problem is equivalent to the general TAM optimization problem $\mathcal{P}_{\mathrm{NPAW}}$ formulated in [11]. Here, we first repeat the problem formulation from [11], and then present a method based on the Lagrange optimization procedure of Section V to solve $\mathcal{P}_{\mathrm{NPAW}}$.

Problem $\mathcal{P}_{\mathrm{NPAW}}$: given an SOC having N cores and a total TAM width W, determine the number of TAMs, a partition of W among the TAMs, an assignment of cores to TAMs, and a wrapper design for each core, such that the total testing time is minimized.

Problem $\mathcal{P}_{\mathrm{NPAW}}$ was shown to be $\mathcal{NP}$-hard in [11].

We use the method of alternating projections [19] to iterate between the Lagrange optimization procedure and a heuristic algorithm for core assignment [12], where the cost function is the SOC testing time. First, the Lagrange optimization procedure is used to obtain a TAM width partition that minimizes the testing time for the SOC (based on an initial arbitrary core assignment). This width partition is then provided as input to the core assignment algorithm [12], and cores are re-assigned to TAMs. After this step, the new assignment is fed as input to the Lagrange optimization procedure and the process is repeated. The Lagrange optimization procedure and the core assignment algorithm are run alternately until the SOC testing time converges, i.e., the decrease in the cost function is less than a predefined threshold.

Fig. 7 illustrates the alternating procedure for core assignment and Lagrange width partition optimization. The wrapper design algorithm from [11] is used to optimize core wrappers for the SOC. From the wrapper design procedure, we obtain the testing time $T_i(k)$ of each core for TAM width k ($1 \leq k \leq w_{\max}$), where $w_{\max}$ is the upper limit on TAM width supplied to the wrapper design algorithm. The core testing times are then input to the core assignment algorithm [12] and cores are assigned to TAMs based on an initial ad hoc TAM width partition in which the width of each TAM is set to $w_{\max}$. After the core assignment is performed, the Lagrange optimization procedure determines the new expression for the cost function $\mathcal{J}$; a TAM partition that minimizes this cost function is obtained. The new TAM width partition is input to the core assignment algorithm and the process repeats until the testing time converges. Convergence is achieved when the decrement in the testing time is less than a threshold value ϵ. In our experiments, we set ϵ to three clock cycles.

Recall from (6) that the cost function for the Lagrange optimization problem is $\mathcal{J}(w_1, \ldots, w_B) = \max_j \left\{\sum_{i=1}^{N} T_i(w_j)x_{ij}\right\} + \lambda \sum_j^B w_j$. Now, the cost function (SOC testing time) for the core assignment algorithm of [12] used in the proposed method is given as: $\mathcal{T} = \max_j \left\{\sum_i^N T_i(w_j)x_{ij}\right\}$. It is, therefore, interesting to note that the cost function expressions for core assignment and TAM optimization are the same, since the values of λ and W remain constant during an execution of the procedure illustrated in Fig. 7. Hence, the testing time converges at a quicker rate than if the Lagrange procedure were run with no alternating core reassignment step. The procedure in Fig. 7 is once again an iterative descent procedure; each Lagrange and each core assignment iteration guarantees a decrease in the testing time. The proof of convergence for this procedure is, therefore, similar to that given in Section V for the Lagrange procedure.

In the absence of an analytical expression for the number of iterations required to arrive at a solution, we demonstrate the efficiency of the proposed procedure empirically. In Table II, we list the total number $p_B(W)$ of unique TAM partitions for a total TAM width of W and for B TAMs. The value of $p_B(W)$ is calculated using the expression $p_B(W) = W^{B-1}/B!(B-1)!$

TABLE II
EFFICIENCY OF THE LAGRANGE OPTIMIZATION PROCEDURE FOR $B = 6$ AND $B = 8$

	Number of TAMs: $B = 6$			
W	$p_B(W)$	P_{eval}	P_{lgr}	η
44	1909	46	18	0.009
48	2949	46	18	0.006
52	4401	65	18	0.004
56	6374	111	18	0.002
60	9000	278	18	0.002
64	12428	708	18	0.001

	Number of TAMs: $B = 8$			
W	$p_B(W)$	P_{eval}	P_{lgr}	η
44	1571	170	24	0.01
48	2889	48	24	0.008
52	5059	100	24	0.004
56	8499	110	24	0.002
60	13776	172	24	0.001
64	21643	256	24	0.001

[12]. Note that this expression is accurate only for larger values of W; hence, we present results only for $W \geq 44$. We compare the efficiency of the Lagrange optimization algorithm with that of the *Partition_evaluate* algorithm proposed to solve Problem $\mathcal{P}_{\mathrm{NPAW}}$ in [12]. The efficiency η is calculated as the ratio of the number of TAM partitions evaluated by the Lagrange optimization procedure to the total number of unique partitions, i.e., $\eta = P_{\mathrm{lgr}}/p_B(W)$. It can be seen that the number P_{lgr} of partitions evaluated by the Lagrange procedure is less than the number P_{eval} of partitions evaluated by *Partition_evaluate*. The value of P_{lgr} is constant over W, but increases super-linearly with B. Since both *Partition_evaluate* and the Lagrange procedure use the same algorithm for core assignment [12], the overall improvement in TAM optimization using the Lagrange procedure is based solely on the new TAM partitioning algorithm. Hence, the performance of the Lagrange procedure does not deteriorate with increasing W, which is not the case for *Partition_evaluate* [12]. This is especially critical when virtual TAMs are designed, since the total virtual TAM width for a high-performance ATE can be very high. For large TAM widths, the computation time in [12] is in the order of minutes, whereas the proposed approach requires computation time in the order of a few seconds.

VII. EXPERIMENTAL RESULTS

In this section, we present experimental results on core assignment and TAM optimization using virtual TAMs. We demonstrate that the SOC testing time and therefore implicit test cost can be significantly reduced using virtual TAMs. We present results for three large benchmark SOCs from the *ITC'02 SOC Test Benchmarks* suite [22].

In Table III, we present results on the testing times obtained for different values of TAM width using virtual TAMs. The testing time is measured in terms of the number of scan clock cycles. The total number of high-frequency and low-frequency ATE pins used for test is denoted by W_{ATE}. Therefore the *real* TAM width at the SOC boundary is W_{ATE}. Of the W_{ATE} pins, there are U high-frequency pins and $(W_{\mathrm{ATE}} - U)$ low-frequency pins. For p22810, we obtain a decrease of as much as 55.3% in testing time. In SOC p34392, one of the cores (Core 18) is a bottleneck core, as a result of which the testing time reaches the lower bound value of 544 579 clock cycles for all TAM widths larger than 32. This property of Core 18 for TAM widths larger than 32 in SOC p34392 was presented in [13]. Using virtual TAMs, it is possible to achieve the lower bound of 544 579 cycles with $W_{\mathrm{ATE}} = 16$. The testing time results for p93791 show an improvement of as much as 58.6% over the testing times obtained without using virtual TAMs, even if only eight pins out of 16 are running at the higher frequency. This represents a significant reduction in implicit test cost. The lower testing times and ATE pin-count requirements on the part of each SOC facilitate greater utilization of the ATE, and provide larger returns on the ATE investment.

In Table IV, we compare our results with four recent TAM optimization approaches [6], [11]–[13]. Note that the testing times presented for the proposed Lagrange optimization approach in the last column of Table IV do not assume virtual TAMs. This is to ensure a fair comparison with the approaches in [6], [11]–[13]. In Column 3 of Table IV, we also list the lower bound values on testing time for the benchmark SOCs calculated in [6].

The U high-frequency pins are assumed to be capable of operating at a frequency of four times that of the $(W_{\mathrm{ATE}} - U)$ low-frequency pins, which operate at the lower scan clock frequency. Therefore, from (3), the number of virtual TAM pins available to cores is given by $W = W_{\mathrm{ATE}} + 3U$. The value of W_{ATE} is varied from 16 to 64 for each benchmark SOC. For each SOC, we perform two sets of experiments, setting (i) $U = W_{\mathrm{ATE}}/2$, and (ii) $U = W_{\mathrm{ATE}}/4$. Testing time results are obtained for both these cases. By T_{lgr},we denote the testing time obtained by using Lagrange optimization, if no virtual TAMs are used. This follows the TAM design methods proposed in [6] and [11]–[13], where the entire TAM width of W_{ATE} was assumed to operate at the lower scan clock frequency, and only W_{ATE} TAM wires are partitioned among the cores. By T_{virt}, we denote the testing time obtained using Lagrange optimization and virtual TAMs. The lower bounds on testing time LB_T for the W virtual TAMs, as discussed in Section IV, are also presented. The percentage decrease in the SOC testing time ΔT using virtual TAMs is presented for each value of W_{ATE} for the three benchmark SOCs. The value of ΔT is calculated as $(T_{\mathrm{new}} - T_{\mathrm{old}})/T_{\mathrm{old}} \times 100$.

The results obtained for the proposed approach relate most closely to those of the *Partition_evaluate* algorithm [12], since the two methods use the same heuristic for core assignment. The CPU times taken by the method in [12] is in the range of a few hundred seconds at most, while the proposed Lagrange procedure is usually half of this. This is because, as shown in Section V, the Lagrange procedure is more efficient than the partitioning approach used in *Partition_evaluate*, therefore the CPU time taken by the Lagrange procedure is less than that required by *Partition_evaluate*. The rectangle packing [13] and TR-Architect [6] algorithms appear to be the most efficient in terms of execution time taking at most 10 s to complete. The ILP/enumeration algorithm [11] takes prohibitively-large execution times (in the range of several minutes to hours, depending on the SOC complexity).

TABLE III
RESULTS ON TESTING TIME (SCAN CLOCK CYCLES) FOR TAM OPTIMIZATION USING VIRTUAL TAMS

SOC	W_{ATE}[1]	$U = \frac{W_{\text{ATE}}}{2}$	W[2]	LB_T	T_{lgr}[3]	T_{virt}[4]	ΔT[5] (%)	$U = \frac{W_{\text{ATE}}}{4}$	W[2]	LB_T	T_{virt}	ΔT[5] (%)
p22810	16	8	40	168575†	434922	194193	−55.3	4	28	240828†	285450	−34.4
	24	12	60	112377†	313607	145417	−53.6	6	42	160547†	190995	−39.1
	32	16	80	102965*	245622	133628	−45.5	8	56	120405†	145417	−40.7
	40	20	100	102965*	194193	121393	−37.4	10	70	102965*	132025	−32.0
	48	24	120	102965*	164755	109555	−33.5	12	84	102965*	132025	−19.8
	56	28	140	102965*	145417	109555	−28.8	14	98	102965*	121393	−16.5
	64	32	160	102965*	133628	109555	−18.0	16	112	102965*	121393	−9.1
p34392	16	8	40	544579*	1021510	544579	−46.7	4	28	544579*	655144	−35.9
	24	12	60	544579*	729864	544579	−25.4	6	42	544579*	544579	−25.4
	32	16	80	544579*	630934	544579	−13.7	8	56	544579*	544579	−13.7
	40	20	100	544579*	544579	544579	0	10	70	544579*	544579	0
	48	24	120	544579*	544579	544579	0	12	84	544579*	544579	0
	56	28	140	544579*	544579	544579	0	14	98	544579*	544579	0
	64	32	160	544579*	544579	544579	0	16	112	544579*	544579	0
p93791	16	8	40	699740†	1775586	734085	−58.6	4	28	999635†	1045840	−41.0
	24	12	60	466486†	1198110	501163	−58.1	6	42	666415†	733567	−38.7
	32	16	80	349864†	936081	472388	−49.5	8	56	499809†	514825	−45.0
	40	20	100	279886†	734085	348567	−52.5	10	70	399845†	514825	−29.9
	48	24	120	233232†	599373	263404	−56.0	12	84	333198†	411860	−31.3
	56	28	140	199913†	514688	257173	−50.0	14	98	285601†	411205	−20.1
	64	32	160	174924†	472388	223598	−52.6	16	112	249898†	348567	−26.2

[1] W_{ATE}: ATE pin-count (*real TAM width*); [2] W: Virtual TAM width; [3] T_{lgr}: Testing time without virtual TAMs, using Lagrange multipliers; [4] T_{virt}: Testing time using virtual TAMs and Lagrange multipliers; [5] ΔT: Percentage change in testing time using virtual TAMs; †: Tighter lower bound obtained from Equation (5) in Section IV; *: Tighter lower bound obtained from [3].

TABLE IV
RESULTS ON TESTING TIME (SCAN CLOCK CYCLES) FOR TAM OPTIMIZATION USING LAGRANGE MULTIPLIERS (WITHOUT VIRTUAL TAMS)

SOC	TAM width W_{ATE}	LB_T	ILP/enum [11]	*Partition_evaluate* [12]	GRP [13]	TR-Architect [6]	Proposed method
p22810	16	421459†	462210	468011	489192	458068	**434922**
	24	280971†	361571	313607	330016	**299718**	313607
	32	210724†	312659	246332	245718	**222471**	245622
	40	168575†	278359	232409	199558	**190995**	194193
	48	140478†	278359	232409	173705	**160221**	164755
	56	120405†	268472	153990	157159	**145417**	**145417**
	64	105355†	260638	153990	142342	**133404**	133628
p34392	16	936872†	**998733**	1033210	1053491	1010821	1021510
	24	621093†	720858	882182	759427	**680411**	729864
	32	544579*	591027	663193	551778	**544579**	630934
	40	544579*	**544579**	**544579**	**544579**	**544579**	**544579**
	48	544579*	**544579**	**544579**	**544579**	**544579**	**544579**
	56	544579*	**544579**	**544579**	**544579**	**544579**	**544579**
	64	544579*	**544579**	**544579**	**544579**	**544579**	**544579**
p93791	16	1749376†	**1771720**	1786200	1932331	1791638	1775586
	24	1166242†	1187990	1209420	1310841	**1185434**	1198110
	32	829724†	**887751**	894342	988039	912233	936081
	40	699740†	**698883**	741965	794027	718005	734085
	48	583112†	**599373**	**599373**	669196	601450	**599373**
	56	499809†	**514688**	**514688**	568436	528925	**514688**
	64	437334†	460328	473997	517958	**455738**	472388

†: Tighter lower bound obtained from Equation (5) in Section IV; *: Tighter lower bound obtained from [3].

As explained in Section VI, the TAM optimization procedure using Lagrange multipliers requires an initial TAM width partition and an initial assignment of cores to the TAM partitions. This feature of the optimization procedure can potentially be used to improve the solution obtained using other techniques, e.g., the heuristic approach of [12] and TR-Architect [6]. The best solution, i.e., with the smallest SOC testing time, obtained using [6], [12] can be used as the starting point for the optimization based on Lagrange multipliers. While the proposed approach is not gauranteed to improve upon the solution in all cases, we found several problem instances in which a small reduction in testing time was achieved over both [12] and [6]. For example, for $W = 28$ and $W = 42$ in SOC p93791, a testing time improvement of 0.3% and 4%, respectively, was achieved over the TR-Architect solution and for $W = 56$ and $W = 64$ in SOC p22810, a testing time improvement of 5.5% and 13.2% was achieved, respectively, over the *Partition_evaluate* solution.

VIII. Effect of Virtual TAMs on Test Power Consumption

Dealing effectively with power consumption in SOCs is highly critical. Excessive power dissipation in an SOC can cause overheating, which can lead to irreversible damage to the chip. Therefore, along with efficient TAM design to minimize the testing time of the SOC, it is also important to investigate the effect of the TAM design on power consumption during scan testing.

The use of virtual TAMs leads to a reduction in the testing time of the SOC due to an increase in the chip-internal TAM width, which allows the cores to be tested on wider TAM partitions. In addition, virtual TAMs can potentially allow a larger number of TAM partitions, thereby providing higher parallelism in core testing. Both peak and average power increase with a larger number of TAM partitions due to greater test parallelism. In this section, we study the impact of virtual TAMs on test power dissipation for one of the ITC'02 benchmark circuits. We limit ourselves to a single benchmark circuit for the following reasons. Detailed information about the power consumption is only available for two benchmarks, namely d695 and d281. The d281 benchmark is a small SOC with six cores and its testing time does not decrease beyond a TAM width of 30, thus, the circuit does not show much variation in the power consumption with the use of virtual TAMs. Therefore, we only present results for d695.

In [9], the peak power consumption for the ISCAS-85 benchmark circuits is estimated based on the maximum switching activity. Since the ISCAS-85 circuits form the cores in SOC d695, we use the power data reported in [9] for estimating the test power for d695. These power estimates have also been used in [8] to evaluate power-constrained TAM optimization. Although the power estimates in [9] are for functional patterns, we use these values for scan test power as in [8] due to lack of any additional power data for these circuits. In [2], a Monte Carlo approach is used to estimate the average power consumption for the ISCAS-85 benchmark circuits; we use these power estimates to study the average power consumption in d695 when virtual TAMs are used.

The instantaneous power consumption $P_{\mathrm{SOC}}(t)$ of an SOC is given by the sum of the instantaneous power consumption values of the cores being tested in test cycle t. This can be expressed as: $P_{\mathrm{SOC}}(t) = \sum_{i=1}^{N} P_i(t)$, where N is the number of cores and $P_i(t)$ is the power consumption of Core i in test cycle t. The peak-power value for Core i is calculated as the maximum of the instantaneous power consumption values for Core i over all test cycles; this is given as: $P_i = \max_t P_i(t)$. Now, let y_{ik} be a binary variable that takes the value 1 if Core i is being tested in test cycle $t = k$, $0 \leq k \leq T$, where T is the total number of test cycles taken for the SOC; else $y_{ik} = 0$. The instantaneous power consumption for the SOC can be expressed as $P_{\mathrm{SOC}}(t) = \sum_{i=1}^{N} P_i.y_{it}$. Note that this is an over-estimate, since the instantaneous power value $P_i(t)$ for Core i is replaced by upper bound P_i; the power consumption for the SOC is likely to be lower in practice. The peak power P_{peak} of the SOC can now be expressed as

$$P_{\mathrm{peak}} = \max_t \sum_{i=1}^{N} P_i.y_{it},\ 0 \leq t \leq T.$$

TABLE V
Peak and Average Power Data for ISCAS-85 Cores in D695 [2], [9]

Core	Peak power (PSF)	Average power (mW)
c432	660	1.12
c499	602	2.05
c880	823	2.75
c1355	275	5.45
c1908	690	9.22
c2670	354	10.80
c3540	530	14.64
c5315	753	23.10
c6288	641	70.32
c7552	1144	37.52

If it takes T_i clock cycles to test core i and it takes T clock cycles to test the SOC, the average power consumption of the SOC is calculated as

$$P_{\mathrm{avg}} = \frac{\sum_{i=1}^{N} P_i \times T_i}{T}.$$

To study the relationship between testing time reduction and SOC test power consumption due to virtual TAMs, we tabulate the peak and average power consumption, and the testing time for d695 for different values of TAM width. In the absence of any previous work that characterizes test power of a core as a function of TAM width, we assume that the test power for a core does change appreciably with TAM width. With wider TAMs, the wrapper scan chains in a core are shorter, and there is less transition activity in them. Since a 0/1 or 1/0 transition traverses a shorter distance in the wrapper scan chains, it is expected a core consumes less power during scan shifting in this case. This depends, however, on the nature of the test patterns. The potential power reduction due to wider TAMs is also offset by the fact that more cores can be tested in parallel with more TAM partitions that may arise due to a wider top-level TAM, which in turn can lead to an increase in test power.

We compare the proposed Lagrange framework with [6], [12] and [13]. As is the case in this paper, test power was not explicitly considered, either in the objective function or as a constraint in [6], [12], [13]. Nevertheless, it is important to evaluate the impact of TAM optimization on test power. The peak power is measured as peak switching frequency (PSF) per node [8], [9] and the average power is measured in mW [2]. Table V shows the peak power and average power data for every core in d695.

We present results for various ATE TAM widths in Tables VI and VII. Of the W_{ATE} pins, there are U high-frequency pins and $(W_{\mathrm{ATE}} - U)$ low-frequency pins. As in Section VII, the U high-frequency pins are assumed to be capable of operating at a frequency four times that of the $(W_{\mathrm{ATE}} - U)$ low-frequency pins, which operate at the lower scan clock frequency. Therefore, from (3), the number of virtual TAM pins available to cores is given by $W = W_{\mathrm{ATE}} + 3U$. The value of W_{ATE} is varied from 16 to 64 and 25% of the ATE pins are assumed to be operating at the high frequency, i.e., $U = W_{\mathrm{ATE}}/4$.

In Table VI, we compare the proposed method with some recent TAM optimization approaches [6], [12], [13]. The testing

TABLE VI
RESULTS ON TESTING TIME AND PEAK AND AVERAGE POWER CONSUMPTION FOR TAM OPTIMIZATION USING VIRTUAL TAMS

1	2	3	4	5	6	7	8	9	10	11	12	13
Partition_Evaluate [12]												
W_{ATE}^1	T_o (cycles)	Number of partitions	$Ppeak_o$ (PSF)	$Pavg_o$ (mW)	W_{vir}	T_n (cycles)	Number of partitions	$Ppeak_n$ (PSF)	$Pavg_n$ (mW)	ΔT^2 (%)	ΔP^3_{peak} (%)	ΔP^4_{avg} (%)
16	42644	4	3288	79.38	28	24701	5	3582	93.79	−42.07	8.94	18.15
24	30032	3	2188	51.42	42	18564	5	3582	94.03	−38.18	63.71	82.86
32	22268	4	2941	72.69	56	13182	6	4112	144.34	−40.80	39.81	98.56
40	18448	3	2587	63.99	70	10432	6	4112	132.18	−43.45	58.94	106.56
48	15300	5	3789	91.10	84	9878	6	4112	108.47	−35.43	8.52	19.06
56	12941	5	3582	144.34	98	9878	6	4112	108.47	−23.66	14.79	−24.85
64	12941	6	4273	108.24	112	9878	6	4112	108.47	−23.66	−3.76	0.212
TR-Architect [6]												
W_{ATE}^1	T_o (cycles)	Number of partitions	$Ppeak_o$ (PSF)	$Pavg_o$ (mW)	W_{vir}	T_n (cycles)	Number of partitions	$Ppeak_n$ (PSF)	$Pavg_n$ (mW)	ΔT^2 (%)	ΔP^3_{peak} (%)	ΔP^4_{avg} (%)
16	44307	3	2587	59.08	28	24701	4	3187	99.21	−44.25	23.19	67.92
24	28576	3	2538	65.38	42	16872	4	3117	69.56	−40.95	22.81	6.39
32	21518	3	2587	53.85	56	12462	5	3764	69.72	−42.08	45.49	29.47
40	17677	3	2587	46.24	70	10216	5	3167	83.25	−42.20	22.41	80.03
48	14608	5	3534	86.27	84	9869	5	3764	80.50	−32.44	6.50	−6.68
56	12462	5	3764	69.72	98	9869	5	3764	80.50	−20.80	0	15.46
64	11033	6	4112	113.62	112	9869	5	3764	80.50	−10.55	−8.46	−29.14
P_{GRP} [13]												
W_{ATE}^1	T_o (cycles)	Number of partitions	$Ppeak_o$ (PSF)	$Pavg_o$ (mW)	W_{vir}	T_n (cycles)	Number of partitions	$Ppeak_n$ (PSF)	$Pavg_n$ (mW)	ΔT^2 (%)	ΔP^3_{peak} (%)	ΔP^4_{avg} (%)
16	43723	N/A	2883	45.57	28	25586	N/A	3671	55.40	−41.48	27.33	21.57
24	30317	N/A	3671	51.22	42	17300	N/A	3758	55.10	−42.93	2.36	7.57
32	23021	N/A	4085	54.33	56	13415	N/A	3818	76.51	−41.72	−6.51	40.82
40	18459	N/A	4215	45.01	70	10869	N/A	4954	66.03	−41.11	17.53	46.70
48	15698	N/A	4215	60.65	84	9869	N/A	4196	55.17	−37.13	−0.45	−9.03
56	13415	N/A	3818	58.86	98	9869	N/A	4085	50.35	−26.43	6.99	−14.40
64	11604	N/A	4699	67.81	112	9869	N/A	5301	50.35	−14.95	12.81	−25.74
Proposed Approach												
W_{ATE}^1	T_o (cycles)	Number of partitions	$Ppeak_o$ (PSF)	$Pavg_o$ (mW)	W_{vir}	T_n (cycles)	Number of partitions	$Ppeak_n$ (PSF)	$Pavg_n$ (mW)	ΔT^2 (%)	ΔP^3_{peak} (%)	ΔP^4_{avg} (%)
16	42644	4	3228	79.38	28	24701	6	4294	93.79	−42.07	33.02	18.15
24	29300	3	2475	51.42	42	18564	5	3582	94.03	−36.64	44.72	82.86
32	22257	6	4201	72.69	56	12192	6	4954	143.39	−45.22	17.92	97.26
40	18448	3	2587	63.99	70	10432	6	4112	132.18	−43.45	58.94	106.56
48	15300	5	3789	91.10	84	9869	6	5812	108.47	−35.49	53.39	19.06
56	12192	6	4954	143.39	98	9869	6	5812	108.47	−19.05	17.31	−24.35
64	11274	7	4896	108.24	112	9869	6	5812	108.47	−12.46	18.70	0.212

[1] W_{ATE}: ATE pin-count (*real TAM width*); [2] ΔT: Percentage change in testing time using virtual TAMs; [3] ΔP_{peak}: Percentage change in peak power using virtual TAMs; [4] ΔP_{avg}: Percentage change in average power using virtual TAMs; N/A: Not applicable (P_{GRP} uses a fork-and-merge TAM architecture that does not have unique TAM partitions).

times obtained with TR-Architect for $W_{\text{ATE}} > 32$ in Table VI have not been published in the literature; these results were made available to us by Erik Jan Marinissen and Sandeep Kumar Goel of Philips Research Laboratories. In Columns 2–4, we present the testing time results along with the peak and average power consumption using W_{ATE} pins with no virtual TAMs. We then present the same set of results with the use of virtual TAMs for all the four approaches in Columns 6–9. The percentage decrease in the SOC testing time ΔT using virtual TAMs is presented for each value of W_{ATE}. The value of ΔT is calculated as $(T_n - T_o)/T_o \times 100$, where T_o represents the testing time obtained without the use of virtual TAMs and T_n represents the testing time with the use of virtual TAMs. Similarly, the change in peak power and average power consumption of the SOC is calculated as $(\text{Ppeak}_n - \text{Ppeak}_o)/\text{Ppeak}_o \times 100$ and $(\text{Pavg}_n - \text{Pavg}_o)/\text{Pavg}_o \times 100$, respectively, where Ppeak_n and Pavg_n represent the peak and average power consumed with the use of virtual TAMs, and Ppeak_o and Pavg_o represent the peak and average power consumption in the absence of virtual TAMs. We also present the number of TAM partitions used to obtain the results. With an increase in the number of TAM partitions, more cores are tested in parallel, hence, the number of TAM partitions indicates the degree of parallelism in core testing.

The percentage increase in the peak and average power are reported in Columns 12 and 13 of Table VI. As expected, the use of virtual TAMs is usually accompanied with an increase in the peak and average power for scan testing. The increase in test power appears to vary directly with the increase in the number of TAM partitions. An increase in the number of TAM partitions implies an increase in the test parallelism. For example, in the proposed approach, the increase in average power is highest for the case ($W_{\text{ATE}} = 40$) for which the increase in the number of partitions is highest. However, it is clear that the test schedule also plays a role in the increase in power, since different approaches with the same number of partitions yield different results.

TABLE VII
RESULTS ON TESTING TIME AND POWER CONSUMPTION FOR A FIXED NUMBER OF TAM PARTITIONS OPTIMIZED USING LAGRANGE MULTIPLIERS

W_{ATE}	T (cycles)	Number of partitions	P_{peak} (PSF)	P_{avg} (mW)	W_{vir}	T_{virt} (cycles)	Number of partitions	P_{peak} (PSF)	P_{avg} (mW)	ΔT (%)	ΔP_{peak} (%)	ΔP_{avg} (%)
16	44924	2	1897	34.42	28	30390	2	1785	32.62	−32.35	−5.90	−5.22
	42674	3	2475	54.34		26869	3	2251	54.23	−37.03	−9.05	−0.202
	42644	4	3228	79.38		24775	4	2829	72.84	−41.90	−12.36	−8.23
	46741	5	3613	88.35		24701	5	3582	93.79	−47.15	−0.85	6.15
	48106	6	4273	104.36		24701	6	4294	121.76	−48.65	0.49	16.67
24	34962	2	1674	32.88	42	22686	2	1897	33.30	−35.11	13.32	1.27
	29300	3	2475	51.42		20861	3	2538	54.39	−30.64	−1.89	5.77
	29872	4	2829	73.53		18799	4	2941	75.00	−37.06	3.9	1.99
	30132	5	3582	102.12		18564	5	3582	94.03	−40.76	0	−7.92
	30132	6	3927	135.26		17663	6	4143	128.07	−41.38	5.50	−5.31
32	26162	2	1897	32.56	56	22717	2	1897	33.35	−13.16	0	2.42
	25630	3	2587	52.54		15510	3	2538	49.56	−39.48	−1.89	−5.67
	22268	4	3005	71.59		14444	4	3228	64.04	−35.13	7.44	−10.54
	21566	5	3671	97.72		13354	5	3216	78.11	−38.07	−12.39	−20.06
	22257	6	4201	72.69		12192	6	4954	143.39	−41.03	−2.11	17.24
40	22930	2	1897	33.24	70	18607	2	1897	29.74	−18.85	0	−10.52
	18448	3	2587	63.99		15209	3	2538	49.19	−17.55	−1.89	−23.12
	19322	4	2829	69.30		13363	4	3117	68.24	−30.84	10.18	−1.52
	18799	5	3582	97.40		13363	5	3582	82.92	−28.91	0	−14.86
	17663	6	4143	126.61		10434	6	4112	132.18	−40.92	−0.748	4.39

Nevertheless, in some cases the use of virtual TAMs leads to reduced test power, especially if the number of TAM partitions is either equal to or less than the number of TAM partitions when virtual TAMs are not used. In these cases, reduced testing time is also accompanied by reduced test power. However, it can be seen from Table VI that a reduction in average power is not always accompanied by a reduction in peak power, and vice versa. From the results reported in Table VI, the P_{GRP} approach [13] appears to be the most efficient from the average and peak power considerations.

Next, we examine the impact of virtual TAMs on test power when the number of TAM partitions is the same as for the case when virtual TAMs are not used. Table VII presents the testing time and the test power for four different values of W_{ATE} for both cases. For each value of W_{ATE}, we vary the number of TAM partitions from two to six. With an increase in the number of TAM partitions, the test time decreases, but the average and peak power increase in some cases. However, the percentage reduction in test time in these cases is much greater than the percentage increase in power consumption. We find that if we limit the number of TAM partitions due to power constraints, we can not only reduce test time using virtual TAMs, but we can also reduce peak and average power during scan testing in most cases. For example, if the number of TAM partitions is limited to four for an ATE TAM width of 16 bits, the reduction in test time is 41%, and it is accompanied by a reduction of 12% and 8% in peak and average power, respectively.

Out of the five TAM optimization approaches discussed in this paper, *ILP/enumerate*, *Partition_evaluate*, and the proposed approach allow the system integrator to specify the number of TAM partitions *a priori*. However, the rectangle packing approach in [13] uses a fork-and-merge TAM design that does not easily lend itself to a predetermined limit on the amount of test parallelism. In the published procedure for TR-Architect [6], the number of TAM partitions is determined by the underlying optimization procedures; therefore, it is also not as amenable to power-constrained TAM optimization.

IX. CONCLUSION

We have presented a new technique to reduce testing time and test cost for core-based SOCs by increasing test resource utilization. The proposed approach, which is based on the concept of virtual TAMs, allows high-speed ATE channels to drive slower scan chains at their maximum rated frequencies. We have shown that even though virtual TAMs operate at scan-chain speeds, they can be interfaced to high-speed ATE channels using bandwidth matching. In this way, the number of on-chip TAM wires is not limited by the number of available pins on the SOC; this allows better utilization of high-speed ATE channels and reduces testing time. We have investigated the effect of virtual TAMs on SOC test power. We have also presented a new TAM optimization framework based on Lagrange multipliers. Experimental results for three industrial SOCs from the ITC'02 SOC test benchmarks demonstrate the effectiveness of the proposed approach. As part of future work, we are investigating power-constrained virtual TAM optimization. We are studying the problem of determining optimum frequencies for the tester channels and for the embedded cores, such that the cores can be matched to tester channels under the constraints of ATE capabilities and test power.

ACKNOWLEDGMENT

The authors thank E. J. Marinissen and S. K. Goel of Philips Research Labs for making unpublished TR-Architect results available to us to allow comparison.

REFERENCES

[1] Semiconductor Test Equipment. Agilent Technologies. [Online]. Available: http://www.ate.agilent.com/ste/products/intelligent_test/SOC_test/

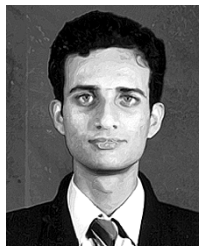
[2] R. Burch, F. Najm, P. Yang, and T. Trick, "A Monte Carlo approach for power estimation," *IEEE Trans. VLSI Syst.*, vol. 1, pp. 63–71, Mar. 1993.

[3] K. Chakrabarty, "Optimal test access architectures for system-on-a-chip," *ACM Trans. Design Automation Electron. Syst.*, vol. 6, pp. 26–49, Jan. 2001.

[4] R. M. Chou, K. K. Saluja, and V. D. Agrawal, "Scheduling tests for VLSI systems under power constraints," *IEEE Trans. VLSI Systems*, vol. 5, no. 2, pp. 175–185, June 1997.
[5] T. H. Cormen, C. E. Leiserson, and D. L. Rivest, *Introduction to Algorithms*. New York: McGraw-Hill, 2001.
[6] S. K. Goel and E. J. Marinissen, "Effective and efficient test architecture design for SOCs," in *Proc. Int. Test Conf.*, 2002, pp. 529–538.
[7] Y. Huang *et al.*, "On concurrent test of core-based SOC design," *J. Electron. Testing: Theory Applicat.*, vol. 18, pp. 401–414, Aug-Oct 2002.
[8] ——, "Optimal core wrapper width selection and SOC test scheduling based on 3-D bin packing algorithm," in *Proc. Int. Test Conf.*, 2002, pp. 74–82.
[9] M. S. Hsiao, E. M. Rudnick, and J. H. Patel, "Effects of delay models on peak power estimation of VLSI sequential circuits," in *Proc. IEEE Int. Conf.*, 1997, pp. 45–51.
[10] (2001) International Technology Roadmap for Semiconductors (ITRS). Silicon Industry Association (SIA). [Online]. Available: http://public.itrs.net
[11] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test wrapper and test access mechanism co-optimization for system-on-chip," *J. Electron. Testing: Theory Applicat.*, vol. 18, pp. 213–230, Apr. 2002.
[12] ——, "Efficient test access mechanism optimization for system-on-chip," *IEEE Trans. Computer-Aided Design*, vol. 22, pp. 635–643, May 2003.
[13] ——, "On using rectangle packing for SOC wrapper/TAM co-optimization," *IEEE Trans. Computers.*, vol. 52, pp. 1619–1632, Dec. 2003.
[14] ——, "Recent advances in TAM optimization, test scheduling, and test resource management for modular testing of core-based SOCs," in *Proc. IEEE Asian Test Symp.*, 2002, pp. 320–325.
[15] A. Khoche *et al.*, "Test vector compression using EDA-ATE synergies," in *Proc. VLSI Test Symp.*, 2002, pp. 97–102.
[16] A. Khoche, "Test resource partitioning for scan architectures using bandwidth matching," in *Dig. Int. Workshop Test Resource Partitioning*, 2002, pp. 1.4–1–1.4–8.
[17] S. Koranne, "A novel reconfigurable wrapper for testing of embedded core-based SOCs and its associated scheduling algorithm," *J. Electron. Testing: Theory Applicat.*, vol. 18, pp. 415–434, Aug.-Oct. 2002.
[18] E. Larsson and Z. Peng, "An integrated framework for design and optimization of SOC test solutions," *J. Electron. Testing: Theory Applicat.*, vol. 18, pp. 385–401, Aug.-Oct. 2002.
[19] D. G. Luenberger, *Optimization by Vector Space Methods*. New York: Wiley, 1969.
[20] E. J. Marinissen *et al.*, "A structured and scalable mechanism for test access to embedded reusable cores," in *Proc. Int. Test Conf.*, 1998, pp. 284–293.
[21] E. J. Marinissen and H. Vranken, "On the role of DFT in IC-ATE matching," in *Dig. Int. Workshop Test Resource Partitioning*, 2001.
[22] E. J. Marinissen, V. Iyengar, and K. Chakrabarty, "A set of benchmarks for modular testing of SOCs," in *Proc. Int. Test Conf.*, 2002, pp. 519–528.
[23] S. Mitra and K. S. Kim, "X-compact: An efficient response compaction technique for test cost reduction," in *Proc. Int. Test Conf.*, 2002, pp. 311–320.
[24] M. Nourani and C. Papachristou, "An ILP formulation to optimize test access mechanism in system-on-chip testing," in *Proc. Int. Test Conf.*, 2000, pp. 113–116.
[25] J. Rajski, "DFT for high-quality low cost manufacturing test," in *Proc. Asian Test Symp.*, 2001, pp. 3–8.
[26] J. Saxena, K. M. Butler, and L. Whetsel, "An analysis of power reduction techniques in scan testing," in *Proc. Int. Test Conf.*, 2001, pp. 670–677.
[27] P. Varma and S. Bhatia, "A structured test re-use methodology for core based system chips," in *Proc. Int. Test Conf.*, 1998, pp. 294–302.
[28] E. Volkerink *et al.*, "Test economics for multisite test with modern cost reduction techniques," in *Proc. VLSI Test Symp.*, 2002, pp. 411–416.
[29] H. Vranken, T. Waayers, H. Fleury, and D. Lelouvier, "Enhanced reduced pin-count test for full scan design," in *Proc. Int. Test Conf.*, 2001, pp. 738–747.
[30] Y. Zorian, E. J. Marinissen, and S. Dey, "Testing embedded-core-based system chips," *IEEE Computer*, vol. 32, pp. 52–60, June 1999.

Anuja Sehgal (S'03) received the B.E. degree in electronics engineering from the Ramrao Adik Institute of Technology, University of Mumbai, India, in 2001, and the M.S. degree in electrical and computer Eengineering from Duke University, Durham, NC, in 2003, where she is currently working toward the Ph.D. degree in the Department of Electrical and Computer Engineering.

Her research interests include test planning and test cost reduction for for digital, mixed-signal, and hierarchical system-on-chips.



Vikram Iyengar (S'94–M'04) received the B.E. degree in electrical engineering from the Birla Institute of Technology, India, in 1996, the M.S. degree in Computer Engineering from Boston University in 1998, and the Ph.D. degree in Computer Engineering from Duke University, Durham, NC, in 2002.

He is now an Advisory Engineer with the ASIC Test Methodology Group, IBM Microelectronics, Essex Junction, VT. He is coauthor of *Test Resource Partitioning for System-on-a-Chip* (Norwell, MA: Kluwer, 2002). His research interests lie in system-on-chip design and test. He has published over 30 journal articles and refereed conference papers in the area of VLSI design and test.

Dr. Iyengar is a recipient of the Boston University Electrical and Computer Engineering Department Chair Fellowship Award in 1996, an IBM Graduate Fellowship Award in 2001, and the EDAA Outstanding Dissertation Award in 2004. He is a member of ACM Sigda and Sigma Xi.

Krishnendu Chakrabarty (S'92–M'96–SM'00) received the B. Tech. degree from the Indian Institute of Technology, Kharagpur, in 1990, and the M.S.E. and Ph.D. degrees from the University of Michigan, Ann Arbor, all in computer science and engineering, in 1992 and 1995, respectively.

He is currently an Associate Professor of Electrical and Computer Engineering at Duke University, Durham, NC. From 2000 to 2002, he was also a Mercator Visiting Professor at the University of Potsdam, Potsdam, Germany. His current research projects include design and testing of system-on-chip integrated circuits, embedded real-time systems, distributed sensor networks, modeling, simulation and optimization of microfluidic systems, and microfluidics-based chip cooling. He is co-author of *Microelectrofluidic Systems: Modeling and Simulation* (Boca Raton, FL: CRC Press, 2002) and *Test Resource Partitioning for System-on-a-Chip* (Norwell, MA: Kluwer, 2002), and editor of *SOC (System-on-a-Chip) Testing for Plug and Play Test Automation* (Norwell, MA: Kluwer 2002). He has contributed chapters to a number of edited books, published over 170 papers in journals and refereed conference proceedings, and holds a U.S. patent in built-in self-test.

Dr Chakrabarty is a recipient of the National Science Foundation Early Faculty (CAREER) Award, the Office of Naval Research Young Investigator Award, and the Humboldt Research Fellowship from the Alexander von Humboldt Foundation, Germany. He is a recipient of a Best Paper Award at the 2001 Design, Automation, and Test in Europe (DATE) Conference. He is an Associate Editor of IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN, Editor of *Journal of Electronic Testing: Theory and Applications (JETTA)*, and a member of the editorial board for *Sensor Letters* and *Journal of Embedded Computing*. He has also served as an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: ANALOG AND DIGITAL SIGNAL PROCESSING. He serves as Vice Chair of Technical Activities in IEEE's Test Technology Technical Council, and is a member of the program committees of several IEEE conferences and workshops. He is a member of ACM, ACM SIGDA, and Sigma Xi.