Test Cost Reduction for SOCs Using Virtual TAMs and Lagrange Multipliers*

Anuja Sehgal[1]
as@ee.duke.edu

Vikram Iyengar[2]
vikrami@us.ibm.com

Mark D. Krasniewski[1]
mdk3@ee.duke.edu

Krishnendu Chakrabarty[1]
krish@ee.duke.edu

[1]Electrical & Computer Engineering, Duke University, Durham, NC 27708, USA
[2]IBM Microelectronics, Essex Jct, VT 05452, USA

ABSTRACT

Recent advances in tester technology have led to automatic test equipment (ATE) that can operate at up to several hundred MHz. However, system-on-chip (SOC) scan chains typically run at lower frequencies (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 testing time. We present a new 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 mechanisms (TAMs). We also present a new TAM optimization framework based on Lagrange multipliers. Experimental results are presented for three industrial circuits from the ITC'02 SOC test benchmarks.

Categories and Subject Descriptors

B.7.3 [**Integrated circuits**]: Reliability and testing

General Terms

Algorithms, Design

Keywords

Automatic test equipment (ATE), bandwidth matching, scan chains, system-on-chip (SOC), test access mechanism (TAM)

1. 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 [16]. 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 [13]. 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.

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.

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 constrain SOC test requirements to match current ATE capabilities include test data compression [10], response compaction [15], and reduced pin-count test [17]. All of 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 seem to predict that future investment in newer and expensive ATE is inevitable [6].

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. 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. Methods to increase the efficiency of ATE use include test scheduling, test access mechanism (TAM) optimization, and multi-site test. Test scheduling seeks to obtain an effective ordering of tests applied to the SOC to minimize testing time [2, 9]. TAM optimization is performed to improve test access to embedded cores in a modular test environment [4, 5, 7]. Finally, multi-site test seeks to test several copies of the SOC simultaneously on the ATE, thus reducing testing time across an entire production batch [16]. While these methods increase the efficiency of ATE use, they assume that the ATE always operates at core scan chain frequencies. Scan chains are typically run at frequencies lower than 50 MHz to reduce power consumption and avoid high-frequency scan design. However, recent advancements in tester design have led to ATE that can operate at up to several hundred MHz. The use of such high-frequency ATE channels at low scan chain frequencies severely under-utilizes ATE capability, resulting in an increase in testing time and time-to-market, thereby directly impacting implicit test cost.

*This research was supported in part by the National Science Foundation under grants CCR-9875324 and CCR-0204077.

DAC 2003, June 2–6, 2003, Anaheim, California, USA

In this paper, we present a new technique to reduce implicit test cost by matching ATE channel frequencies to core scan chain 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. 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 also propose a new method for virtual TAM optimization to improve test data transport from ATE channels to core I/Os. The new method based on Lagrange multipliers [12] exploits the monotonically non-increasing function of core testing time with TAM width to effectively partition the set of virtual TAM wires among the cores.

The rest of the paper is organized as follows. In Section 2, we introduce the concept of virtual TAMs. In Section 3, we discuss the use of Lagrange multipliers to TAM width partitioning. In Section 4, 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 5, we present experimental results for benchmark SOCs demonstrating the applicability of our methods. We conclude the paper in Section 6.

2. VIRTUAL TAMS

Recent advancements in ATE technology have led to a substantial increase in ATE channel frequencies. However, the frequency at which an embedded core can be tested is limited by its scan chain frequency, typically under 50 MHz. Core scan chain 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 [4, 5, 7], 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 [10]. The system TAMs are of two kinds: i) low-frequency TAMs driven by low-frequency ATE pins, and ii) high-frequency TAMs driven by high-frequency ATE pins. We apply bandwidth matching to the interface between high-frequency TAMs that interface with high-frequency ATE channels and low-frequency virtual TAMs that drive core scan chains; see Figure 1. Virtual TAMs are based on the following relationship between the TAM width and operating frequency of test data transport mechanisms:

$$W_{\mathrm{ATE}} \times f_{\mathrm{ATE}} = W_{\mathrm{TAM}} \times f_{\mathrm{TAM}} \quad (1)$$

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.

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 Equation (1) is satisfied. This is illustrated as follows; again see Figure 1. Given an SOC TAM of W_{ATE} pins (driven by the ATE), of which U pins are driven at the higher frequency f_{ATE} and $(W_{\mathrm{ATE}} - U)$ pins are driven at the lower scan frequency f_{TAM}, such that $f_{\mathrm{ATE}} = n \times f_{\mathrm{TAM}}$ using frequency division and bandwidth matching, the following relationship holds:

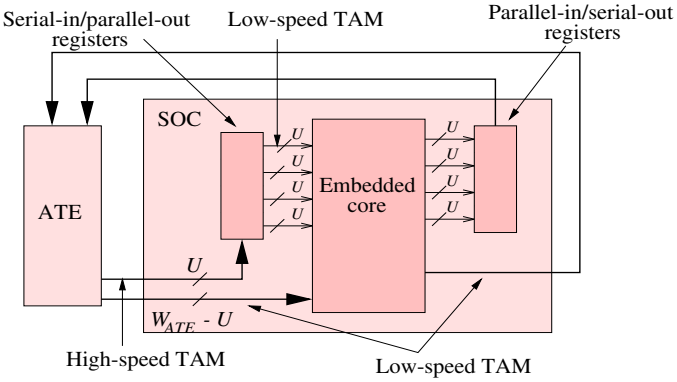

Figure 1: Virtual TAMs based on bandwidth matching.

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

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

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

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 a larger amount of test data is available to cores. Moreover, since the serial-in/parallel-out interfaces used for bandwidth matching are placed next to the cores, only the original W_{ATE} TAM wires are routed through the system. Thus, a large number of TAM wires can be obtained with low routing and hardware cost.

3. 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. Here, we first describe a simple TAM optimization problem, and 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 4, 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 non-increasing functions, as shown in [9]. We now solve the following optimization problem: determine the values of w_1, w_2, such that (i) $w_1 + w_2 = W$, and (ii) $\max\{T_1(w_1), T_2(w_2)\}$ is minimized, where W denotes the total virtual TAM width available.

We rephrase this problem as the minimization of a Lagrange cost function [12]. Let the Lagrange cost function $\mathcal{J}(w_1, w_2)$ be defined as

$$\mathcal{J}(w_1, w_2) = \max\{T_1(w_1), T_2(w_2)\} + \lambda(w_1 + w_2) \quad (4)$$

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 Equation (4) [12]. Thus, instead of minimizing $\max\{T_1(w_1), T_2(w_2)\}$, we solve Equation (4). 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 bound the value of W.

Case 1. Let us minimize the expression for $\mathcal{J}(w_1, w_2)$ in Equation (4) while setting λ to 0. If $\lambda = 0$, then $\mathcal{J}(w_1, w_2) \mid_{\lambda=0} = \max\{T_1(w_1), T_2(w_2)\}$. Hence, the penalty term $\lambda(w_1 + w_2)$ vanishes. Now, since both $T_1(w_1)$ and $T_2(w_2)$ are monotonically non-increasing, $\mathcal{J}(w_1, w_2) \mid_{\lambda=0}$ is minimized when both $w_1 \longrightarrow \infty$ 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 Equation (4), $\mathcal{J}(w_1, w_2) \approx \lambda(w_1 + w_2)$. The penalty term thus outweighs the min-max term in Equation (4). 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 by varying the value of the Lagrange multiplier λ, it is possible to minimize $\mathcal{J}$ (and equivalently, the SOC testing time cost function) for different values of W.

We next formalize the problem for the general case consisting of $B \geq 2$ TAMs and $N \geq 2$ cores. Recall that the core assignment to TAMs is pre-determined. 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 Equation (4) 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 $\mathcal{J}(w_1, ..., w_B)$ is minimized, where

$$\mathcal{J}(w_1, \ldots, w_B) = \max_j \left\{ \sum_i^N T_i(w_j) x_{ij} \right\} + \lambda \sum_{j=1}^{B} w_j. \qquad (5)$$

The expression $\max_j \left\{ \sum_i^N T_i(w_j) x_{ij} \right\}$ gives the maximum testing time over all TAMs. Equation (5) thus represents a B-dimensional optimization problem.

Iterative descent procedure. To solve Equation (5), we use an iterative descent procedure that 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.

An important property of the 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 $(n-1)^{th}$ 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 non-increasing function of n. Since a monotonically non-increasing function that is bounded from below is guaranteed to converge, the iterative procedure is also guaranteed to converge.

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. Further, 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 non-increasing functions. Let $\lambda = 1$. We wish to minimize $\mathcal{J}(w_1, w_2)$, where

$$\mathcal{J}(w_1, w_2) = \max\{10e^{-w_1}, 10e^{-2w_2}\} + (w_1 + w_2). \qquad (6)$$

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 $\lambda = 1$, therefore $J^{(0)} = 20.0005$.

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

$$\mathcal{J}'(w_1) = \max\{10e^{-w_1}, 2 \times 10^{-8}\} + w_1 + 10 \qquad (7)$$

Using the bisection search method [3], we find that the value $w_1 = 2$ minimizes the cost function in Equation (7). 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

$$\mathcal{J}'(w_2) = \max\{1.35, 10e^{-2w_2}\} + 2 + w_2 \qquad (8)$$

Here, bisection search [3] 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 is exponential in B.

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. For example, for the SOC benchmark circuit p22810, a λ value of 10,000 yields a TAM partition for a TAM width of 8. Since, λ varies inversely and monotonically with W, we use a bisection search over all possible values of λ to arrive at a solution for a given TAM width.

4. 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 $\mathcal{P}_{\text{NPAW}}$ formulated in [7]. Here, we first repeat the problem formulation from [7], and then present a method based on the Lagrange optimization procedure of Section 3 to solve $\mathcal{P}_{\text{NPAW}}$.

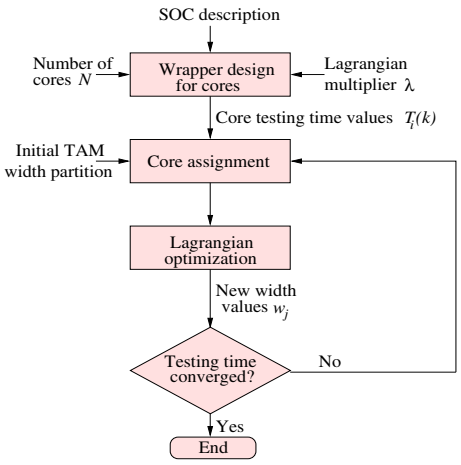

Figure 2: Procedure for core assignment and TAM optimization.

Problem $\mathcal{P}_{\rm NPAW}$: 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 $\mathcal{P}_{\rm NPAW}$ was shown to be $\mathcal{NP}$-hard in [7].

We use the method of alternating projections [12] to iterate between the Lagrange optimization procedure and a heuristic algorithm for core assignment [8], whose cost function is again 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 ad hoc core assignment). This width partition is then input to the core assignment algorithm [8], 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 to a fixed value.

Figure 2 illustrates the alternating procedure for core assignment and Lagrange width partition optimization. The wrapper design algorithm from [7] 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 [8] 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 3 clock cycles.

Recall from Equation (4) 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 [8] 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 Figure 2. Hence the testing time converges at a quicker rate than if the Lagrange procedure were run with no alternating core re-assignment step. The procedure in Figure 2 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 3 for the Lagrange procedure.

W	Number of TAMs: $B = 6$			
	$p_B(W)$	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

W	Number of TAMs: $B = 8$			
	$p_B(W)$	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

Table 1: Efficiency of Lagrange procedure for $B = 6$ and 8.

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 1, 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 $p_B(W) = \frac{W^{B-1}}{B!(B-1)!}$ [8]. Note that this expression is accurate only for larger values of W; hence we present results only for $W \geq 44$. In Table 1, we compare the efficiency of the Lagrange optimization algorithm with that of the *Partition_evaluate* algorithm proposed to solve Problem $\mathcal{P}_{\rm NPAW}$ in [8]. 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. 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 [8], 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* [8]. 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 [8] is in the order of minutes, whereas the proposed approach requires computation time in the order of a few seconds.

5. 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. Experimental results are presented for three benchmark SOCs from the *ITC'02 SOC Test Benchmarks* suite [14].

In Table 2, 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 $W_{\rm ATE}$. Therefore the *real* TAM width at the SOC boundary is $W_{\rm ATE}$. Of the $W_{\rm ATE}$ pins, there are U high-frequency pins and $(W_{\rm ATE} - U)$ low-frequency pins.

SOC	W_{ATE} [1]	$U = \frac{W_{\text{ATE}}}{2}$	W [2]	LB_T	T_{lgr} [3]	T_{virt} [4]	ΔT (%)	$U = \frac{W_{\text{ATE}}}{4}$	W [2]	LB_T	T_{virt}	ΔT [5] (%)
p22810	16	8	40	$167787^{\dagger}$	434922	194193	−55.3	4	28	$239665^{\dagger}$	285450	−34.4
	24	12	60	$111859^{\dagger}$	313607	153990	−50.9	6	42	$159797^{\dagger}$	190995	−39.1
	32	16	80	102965^{*}	245622	145417	−40.8	8	56	$119848^{\dagger}$	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	$544579^{\ddagger}$	1021510	544579	−46.7	4	28	$544579^{\ddagger}$	655144	−35.9
	24	12	60	$544579^{\ddagger}$	729864	544579	−25.4	6	42	$544579^{\ddagger}$	544579	−25.4
	32	16	80	$544579^{\ddagger}$	630934	544579	−13.7	8	56	$544579^{\ddagger}$	544579	−13.7
	40	20	100	$544579^{\ddagger}$	544579	544579	0	10	70	$544579^{\ddagger}$	544579	0
	48	24	120	$544579^{\ddagger}$	544579	544579	0	12	84	$544579^{\ddagger}$	544579	0
	56	28	140	$544579^{\ddagger}$	544579	544579	0	14	98	$544579^{\ddagger}$	544579	0
	64	32	160	$544579^{\ddagger}$	544579	544579	0	16	112	$544579^{\ddagger}$	544579	0
p93791	16	8	40	$698670^{\dagger}$	1775586	734085	−58.6	4	28	$998095^{\dagger}$	1132615	−36.2
	24	12	60	$465784^{\dagger}$	1198110	501163	−58.1	6	42	$665400^{\dagger}$	734085	−38.7
	32	16	80	$349341^{\dagger}$	936081	472388	−49.5	8	56	$499053^{\dagger}$	514825	−45.0
	40	20	100	$279475^{\dagger}$	734085	410483	−44.1	10	70	$399245^{\dagger}$	514825	−29.9
	48	24	120	$232898^{\dagger}$	599373	366888	−38.8	12	84	$332706^{\dagger}$	411860	−31.3
	56	28	140	$199628^{\dagger}$	514688	257173	−50.0	14	98	$285178^{\dagger}$	411205	−20.1
	64	32	160	$174676^{\dagger}$	472388	223598	−52.6	16	112	$249532^{\dagger}$	408683	−13.5

[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 [4]; *: Tighter lower bound obtained from [1]; ‡: same lower bound obtained from [1] and [4]

Table 2: Results on testing time (scan clock cycles) for TAM optimization using virtual TAMs.

SOC	TAM width W_{ATE}	LB_T [4]	ILP/enum [7]	*Partition_evaluate* [8]	GRP [9]	TR-Architect [4]	Proposed method
p22810	16	419466	462210	468011	489192	458068	**434922**
	24	279644	361571	313607	330016	**299718**	313607
	32	209734	312659	246332	245718	**222471**	245622
	40	167787	278359	232409	199558	**190995**	194193
	48	139823	278359	232409	173705	**160221**	164755
	56	119848	268472	153990	157159	**145417**	**145417**
	64	104868	260638	153990	142342	**133404**	133628
p34392	16	932790	**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	1746657	**1771720**	1786200	1932331	1791638	1775586
	24	1164442	1187990	1209420	1310841	**1185434**	1198110
	32	873334	**887751**	894342	988039	912233	936081
	40	698670	**698883**	741965	794027	718005	734085
	48	582227	**599373**	**599373**	669196	601450	**599373**
	56	499053	**514688**	**514688**	568436	528925	**514688**
	64	436673	460328	473997	517958	**455738**	472388

Table 3: Results on testing time (scan clock cycles) for TAM optimization using Lagrange multipliers (without virtual TAMs).

The U high-frequency pins are assumed to be capable of operating at a frequency of four times that of the $(W_{\rm ATE} - U)$ low-frequency pins, which operate at the lower scan chain frequency. Therefore, from Equation (3), the number of virtual TAM pins available to cores is given by $W = W_{\rm ATE} + 3U$. The value of $W_{\rm ATE}$ is varied from 16 to 64 for each benchmark SOC. For each SOC, we perform two sets of experiments, seting (i) $U = \frac{W_{\rm ATE}}{2}$, and (ii) $U = \frac{W_{\rm ATE}}{4}$. 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 [4, 7, 8, 9], where the entire TAM width of $W_{\rm ATE}$ was assumed to operate at the lower scan chain frequency, and only $W_{\rm ATE}$ 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 are also presented. These bounds are derived from the formulas presented in [1, 4]. The percentage decrease in the SOC testing time ΔT using virtual TAMs is presented for each value of $W_{\rm ATE}$ for the three benchmark SOCs. The value of ΔT is calculated as $\frac{T_{new} - T_{old}}{T_{old}} \times 100$.

For p22810, we obtain a decrease of as much as 47.7% 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 544579 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 [9]. Using virtual TAMs, it is possible to achieve the lower bound of 544579 cycles with $W_{\rm ATE} = 16$. 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 8 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 3, we compare our results with four recent TAM optimization approaches [4, 7, 8, 9]. In [7], the authors optimized a test bus architecture using a combination of integer linear programming (ILP) and exhaustive enumeration. The work in [7] was later improved in [8] to include a heuristic method for core assignment. This heuristic core assignment approach forms a part of the TAM optimization method presented in this paper. In [9], the authors presented a method to integrate TAM design and test scheduling using rectangle packing. Finally, in [4], the authors presented a heuristic algorithm TR-Architect for TestRail optimization. In Column 3 of Table 3, we also list the lower bound values on testing time for the benchmark SOCs calculated in [4]. Note that the testing times presented for the proposed Lagrange optimization approach in the last column of Table 3 do not assume virtual TAMs. This is to ensure a fair comparison with the approaches in [4, 7, 8, 9].

The results obtained for the proposed approach compare most closely to those of the *Partition_evaluate* algorithm [8], since the two methods use the same heuristic for core assignment. The CPU times taken by the method in [8] 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 3, 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 [9] and TR-Architect [4] algorithms appear to be the most efficient in terms of execution time taking at most 10 seconds to complete. The ILP/enumeration algorithm [7] takes prohibitively-large execution times (in the range of several minutes to hours), depending on the SOC complexity.

6. 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 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.

7. REFERENCES

[1] K. Chakrabarty. Optimal test access architectures for system-on-a-chip. *ACM Trans. Design Automation of Electronic Systems*, vol. 6, pp. 26–49, January 2001.

[2] 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, June 1997.

[3] T.H. Cormen, C.E. Leiserson and D.L. Rivest. *Introduction to Algorithms*, McGraw-Hill, New York, NY, 2001.

[4] S.K. Goel and E.J. Marinissen. Effective and efficient test architecture design for SOCs. *Proc. Int. Test Conf.,* pp. 529–538, 2002.

[5] Y. Huang et al. On concurrent test of core-based SOC design. *J. Electronic Testing: Theory and Applications*, vol. 18, pp. 401–414, Aug–Oct 2002.

[6] International Technology Roadmap for Semiconductors (ITRS). Silicon Industry Association (SIA). *http://public.itrs.net*, 2001.

[7] V. Iyengar, K. Chakrabarty and E. J. Marinissen. Test wrapper and test access mechanism co-optimization for system-on-chip. *J. Electronic Testing: Theory and Applications*, vol. 18, pp. 213–230, April 2002.

[8] V. Iyengar, K. Chakrabarty, and E. J. Marinissen. Efficient wrapper/TAM co-optimization for large SOCs. *Proc. Design Automation and Test in Europe Conf.*, pp. 491–498, 2002.

[9] V. Iyengar, K. Chakrabarty, and E.J. Marinissen. On using rectangle packing for SOC wrapper/TAM co-optimization. *Proc. VLSI Test Symp.*, pp. 253–258, 2002.

[10] A. Khoche et al. Test vector compression using EDA-ATE synergies. *Proc. VLSI Test Symp.*, pp. 97–102, 2002.

[11] A. Khoche. Test resource partitioning for scan architectures using bandwidth matching. *Digest of Int. Workshop on Test Resource Partitioning*, pp. 1.4-1–1.4-8, 2002.

[12] D.G. Luenberger. *Optimization by Vector Space Methods*, John Wiley and Sons, New York, NY, 1969.

[13] E.J. Marinissen and H. Vranken. On the role of DfT in IC - ATE matching. *Digest of Int. Workshop on Test Resource Partitioning*, 2001.

[14] E.J. Marinissen, V. Iyengar and K. Chakrabarty. A Set of Benchmarks for Modular Testing of SOCs. *Proc. Int. Test Conf.*, pp. 519–528, 2002.

[15] J. Rajski. DFT for high-quality low cost manufacturing test. *Proc. Asian Test Symp.*, pp. 3–8, 2001.

[16] E. Volkerink et al. Test economics for multi-site test with modern cost reduction techniques. *Proc. VLSI Test Symp.*, pp. 411-416, 2002.

[17] H. Vranken, T. Waayers, H. Fleury and D. Lelouvier. Enhanced reduced pin-count test for full scan design. *Proc. Int. Test Conf.*, pp. 738–747, 2001.