PAPER

Design and Optimization of Transparency-Based TAM for SoC Test

Tomokazu YONEDA[†a)], ***Member***, **Akiko SHUTO**[††*], ***Nonmember***, **Hideyuki ICHIHARA**[††b)], **Tomoo INOUE**[††c)], ***Members***, ***and*** **Hideo FUJIWARA**[†d)], ***Fellow***

SUMMARY We present a graph model and an ILP model for TAM design for transparency-based SoC testing. The proposed method is an extension of a previous work proposed by Chakrabarty with respect to the following three points: (1) constraint relaxation by considering test data flow for each core separately, (2) optimization of the cost for transparency as well as the cost for additional interconnect area simultaneously and (3) consideration of additional bypass paths. Therefore, the proposed ILP model can represent various problems including the same problem as the previous work and produce better results. Experimental results show the effectiveness and flexibility of the proposed method compared to the previous work.

key words: *SoC test, design for testability, TAM design, transparency, ILP*

1. Introduction

SoCs are increasingly designed and tested in a modular fashion. In order to perform modular test, each embedded core should be isolated from its surrounding circuitry. Zorian et al. introduced a generic test architecture that permits modular test for SoCs [1]. It consists of the following three components: 1) test pattern source and test response sink, 2) test access mechanism (TAM), and 3) wrapper. The TAM propagates test patterns for a core from test pattern source to the core, and furthermore propagates the responses from the core to test pattern sink. The wrapper provides an interface between TAM and core and it is standardized as IEEE Std. 1500 [2].

A number of approaches have been proposed for wrapper and TAM design including test scheduling problem [3]–[10]. These approaches use the infrastructure dedicated to test such as *TestBus* [11], [12] and *TESTRAIL* [5] as TAMs, and most of them use the SoC test time as minimization criterion. However, the routing overhead of TAMs is another important cost in SoC testing. Therefore, a number of TAM approaches which are not dedicated to test have also been proposed.

The TAM architectures which are not dedicated to test can be roughly classified into three types: 1) the method re-using functional buses [13]–[15], 2) the methods re-using functional networks [16], [17] and 3) the methods based on transparency [18]–[23]. The methods re-using functional buses and networks assume that cores are accessible by using the buses and the networks while the transparency-based methods deal with SoCs without such direct accessible connections.

Transparency-based methods can be further classified into two types in terms of their properties of transparency: 1) single-cycle throughput transparency [20]–[23] and 2) multi-cycle throughput transparency [18], [19]. Single-cycle throughput transparency has two main advantages compared to multi-cycle throughput transparency: 1) short test application time and 2) ability to preserve timing information for test sequences. In [20], a single-cycle throughput transparency-based TAM design method was proposed to minimize the area overhead of additional gate counts. The authors extended [20] so that it can handle test time and area overhead co-optimization problem in [21] and power constraints in [22]. [23] also proposed a single-cycle throughput transparency-based method to minimize the overhead of additional interconnect area. However, previous work [20]–[23] considered core-level transparency design problem and system-level TAM design problem separately. In other words, the authors first tackled only the design for transparency problem without considering the system level connectivity information. After that, the authors worked on the optimization problem to minimize system-level cost for TAM design without considering the cost of making cores transparent.

In this paper, we extend the method proposed in [23] which is based on integer linear programming (ILP) formulation so that not only the system-level cost for TAM design but also the core-level cost for making cores transparent can be simultaneously taken into consideration during the optimization process. Moreover, we also relax the constraints by considering test data flows for each core under test separately and extend it to be able to handle the case where cores cannot be made transparent by the embedded multiplexers in the behavioral models proposed in [23] due to IP protection. In order to deal with the core-level cost for transparency as well as the system-level cost for additional interconnects, we propose a new graph model whose vertices are input/output ports of cores while the vertices in [23] are cores themselves. And, the edges in the proposed graph are interconnects and transparent paths while

Manuscript received October 2, 2009.
Manuscript revised January 19, 2010.

[†]The authors are with Nara Institute of Science and Technology (NAIST), Ikoma-shi, 630–0192 Japan.
[††]The authors are with Hiroshima City University, Hiroshima-shi, 731–3194 Japan.
[*]Presently, with Sony Semiconductor Kyushu Co., Ltd.
a) E-mail: yoneda@is.naist.jp
b) E-mail: ichihara@hiroshima-cu.ac.jp
c) E-mail: tomoo@hiroshima-cu.ac.jp
d) E-mail: fujiwara@is.naist.jp

DOI: 10.1587/transinf.E93.D.1549

the edges in [23] are interconnects only. Moreover, in [23], only the sufficient conditions for transparency-based TAM design are considered to minimize the amount of computation necessary to solve the ILP model, and the constraints are local in the sense that transparent access to any core is provided by exactly one justification path from system inputs and exactly one propagation path to system outputs independently of core under test. In contrast with [23], we do a global analysis to relax the conditions for transparency-based TAM design by using more variables and more constraints in the proposed ILP formulation. We can reduce the cost by considering the test requirements for each core under test independently and determining unique justification and propagation paths for it. Though we increase the number of vertices and edges in the graph modeling and the number of variables and constraints in the ILP formulation, the proposed ILP model can be solved in reasonable time for large SoCs. Furthermore, the proposed ILP model can represent various problems including the same problem as [23] by setting constant parameters appropriately. Experimental results show the effectiveness and flexibility of the proposed method.

The rest of this paper is organized as follows. We discuss the related work proposed in [23] in Sect. 2. Section 3 shows the proposed TAM design method for transparency-based SoC test including graph modeling and ILP modeling. Experimental results are discussed in Sect. 4. Finally, Sect. 5 concludes this paper.

2. Related Work

In [23], single-cycle transparency is achieved by embedding multiplexers in the behavioral models described using a hardware description language. An example is shown in Fig. 1. An additional control input T is used to switch a core from the normal mode ($T = 0$) to transparent mode ($T = 1$). An additional 2-bit output port Y is added to ensure complete transparency. In general, a core may be required to expand its input/output ports for transparent access of other cores, not just for itself. Therefore, [23] proposed a method to minimize the overhead of additional ports and associated interconnect area. The method analyzes SoC testing requirements and formulates it as an ILP problem as follows.

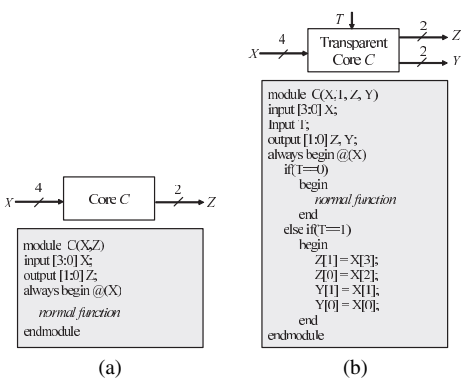


Fig. 1 Embedding a multiplexer in the behavioral description. (a) Original core. (b) Core with embedded multiplexers.

The method first constructs a weighted directed system graph whose vertices are the cores, fanout points of functional interconnects, a primary input and a primary output and whose edges are functional interconnects. All the primary inputs (outputs) are considered as a single vertex in the graph. The weight of an edge (C_i, C_j) denotes the total width of the buses connecting C_i to C_j. An example SoC S_1 and the corresponding system graph are shown in Fig. 2. F_1 corresponds to the fanout point of the interconnect between C_3 and C_4, and F_2 corresponds to the fanout point of the interconnect between C_4 and Z_1, respectively. Next, it breaks all cycles in the system graph by solving the minimum feedback vertex set problem. The acyclic SoC and the corresponding system graph G are shown in Fig. 3. In this example, the cycles can be broken by removing the edge between F_2 and C_2. This implies that the 3-bit input to C_2 must be multiplexed to a primary input. The 3-bit input to C_2 represents the edge between PI and C_2 in Fig. 3.

Then, it constructs the same weighted directed graph G^* as G except for the weights of edges. The new edge weights in G^* are determined by analyzing the justification and propagation requirements of each core using test graph and solving the ILP problem defined as follows.

A test graph G_j for core C_j is a subgraph of G^* which contains a vertex C_i if either of the two conditions holds: i) C_i lies on a directed path from the source vertex to C_j, or ii) C_i lies on a directed path from C_j to the sink vertex. Similarly, an edge (C_i, C_k) belongs to G_j if it either lies on a path from the source vertex to C_j or on a path from C_j to the sink vertex. Figure 4 (a) shows the test graph for C_3. For each test graph G_j, it imposes a set of constraints on edge weights for the edges in G^* as follows.

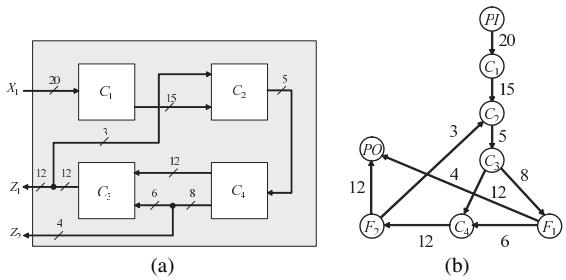


Fig. 2 An example SoC S_1 and its system graph.

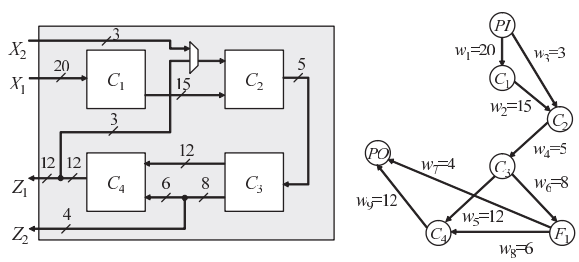
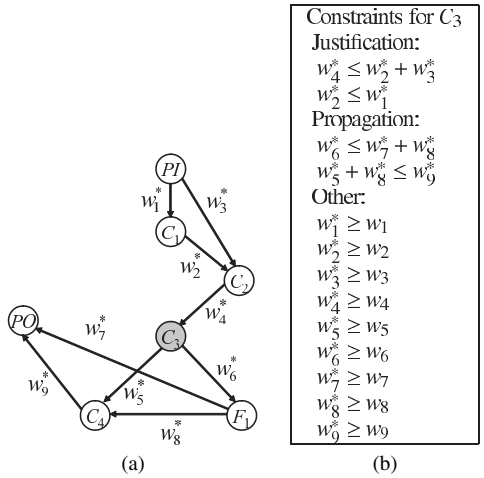


Fig. 3 Acyclic SoC and its system graph G for the example SoC S_1.



(a) (b)

Fig. 4 (a) Test graph G_3. (b) Constraints on the edge weights for C_3 in [23].

1. justification constraints: If C_i lies on a path from the source vertex to C_j in G_j, then the sum of the weights of the edges directed away from C_i in G_j must not exceed the sum of the weights of the edges incident on C_i.
2. propagation constraints: If C_i lies on a path from C_j to the sink vertex in G_j, then the sum of the weights of the edges incident on C_i in G_j must not exceed the sum of the weights of the edges directed away from C_i.

Moreover, for each edge, the weight of the edge in G must not exceed the weight of the edge in G^* (i.e., $w_k^* \geq w_k$). The constraints on the edge weights for C_3 are shown in Fig. 4 (b). The total increase in the interconnect for S_1 is given by $Cost = \sum_i (w_i^* - w_i)$ where w_i^*s are variables whose values are to be determined and w_is are known constants. Therefore, the problem to minimize interconnect area can be expressed an ILP model where the objective is to minimize $Cost$ subject to the constraints on the edge weights.

3. TAM Design for Transparency-Based SoC Test

3.1 Problem Formulation

In this section, we present a graph modeling and ILP modeling method for TAM design. The proposed method is an extension of [23] with respect to the following three points: (1) constraint relaxation by considering test data flow for each core separately, (2) optimization of the cost for transparency as well as the cost for additional interconnect area simultaneously and (3) consideration of additional bypass paths. Before describing the details of graph and ILP modeling, we explain the effectiveness of the above-mentioned extensions separately, and present the transparency-based TAM design problem.

(**Constraint Relaxation**) In [23], only the sufficient conditions are considered to minimize the amount of computation necessary to ensure cores' transparency, and the constraints are local in the sense that transparent access to any core is provided by exactly one justification path from system inputs and exactly one propagation path to system outputs independently of core under test.

However, if additional computation for global analysis is permitted, we can relax the constraints by considering the test requirements for each core under test independently and determining unique justification and propagation paths for it.

For example, in [23], for each edge, the weight of the edge in G must not exceed the weight of the edge in G^*, i.e., $w_i^* \geq w_i$. In Fig. 4, we need $w_6^* \leq w_7^* + w_8^*$ and $w_5^* + w_8^* \leq w_9^*$ as propagation constraints for C_3 and the constraint $w_i^* \geq w_i$ upon every edge. Consequently, the cost to satisfy the constraints is 6 ($w_5^* = 12, w_6^* = 8, w_7^* = 4, w_8^* = 6, w_9^* = 18$). However, the constraint $w_i^* \geq w_i$ is not necessary for every edge when we consider the justification and propagation constraints for a core. When we consider the test requirements for core C_j, the constraint $w_i^* \geq w_i$ is necessary only for the edge that is incident on C_j or directed away from C_j. In this example, we need only the following three constraints: (1) $w_4^* \geq w_4$, (2) $w_5^* \geq w_5$ and (3) $w_6^* \geq w_6$, and the propagation constraints for C_3 can be satisfied with cost of 4. ($w_5^* = 12, w_6^* = 8, w_7^* = 4, w_8^* = 4, w_9^* = 16$). Therefore, we can relax the constraints by considering the test requirements for each core separately using different variables.

(**Cost for Transparency**) During the optimization of transparency-based TAM design, we have to consider two types of cost. One is the area overhead for making cores transparent and the other is the routing overhead for additional interconnect. The former cost can be determined within each core while the latter cost depends on the global layout information, and therefore it is difficult to compare these costs. The method in [23] considers that the area overhead for making cores transparent is negligible and tries to minimize only the number of additional interconnects. However, we can get better solution if we know the relative costs between the above two types of cost and both costs can be taken into account simultaneously during the optimization.

Figure 5 is a simple example. If we consider only the cost for additional interconnects, the propagation constraints for C_1 can be satisfied without additional interconnect by using the dotted line shown in Fig. 5 (a). Though the cost of additional interconnect area is 0, we actually have to pay the cost for making C_2 6-bit transparent. Suppose that the relative cost for 1-bit transparency to that for 1-bit additional interconnect is 1 (both costs are 1), then the total cost for the solution shown in Fig. 5 is 6. On the other hand, if we consider both two costs simultaneously, then we can get better solution shown in Fig. 5 (b) where the total cost is 2. Moreover, if the interconnect from the fanout to primary output Z_2 is very short and the cost for 1-bit transparency in C_2 is higher than the cost for 1-bit increase on Z_2, then we can get another solution shown in Fig. 5 (c). From this example, we can say that it is important and effective to consider the cost

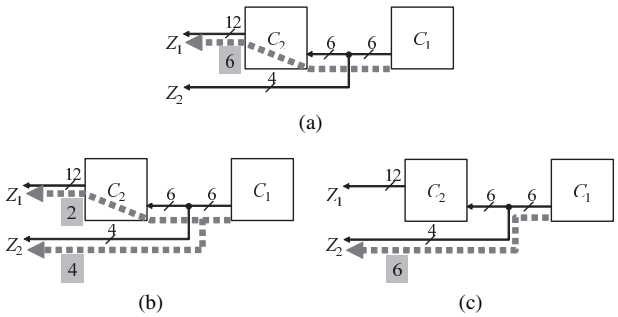


Fig. 5 Propagation paths in various costs for transparency. (a) Cost for making C_2 1-bit transparent is 0. (b) Cost for making C_2 1-bit transparent is 1. (c) Cost for making C_2 1-bit transparent is 2 and cost for 1-bit additional interconnect on Z_2 is 1.

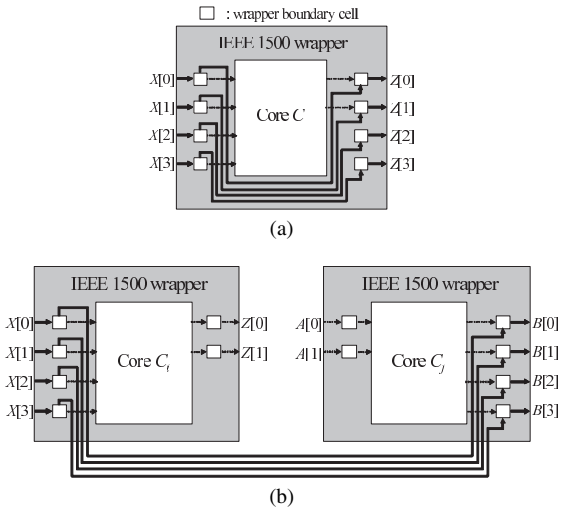


Fig. 6 (a) Single-core bypass path by IEEE 1500 wrapper. (b) Multi-core bypass path by IEEE 1500 wrapper.

of making cores transparent as well as the cost of additional interconnect area simultaneously during optimization.

(**Additional Bypass Path**) In the SoC design strategies, the behavioral models described using a hardware description language are not always available due to IP protection and so on. Even if it is available, it may happen that the total cost (including area and layout etc.) of making cores transparent by embedding multiplexers [23] is higher than that of bypass paths added outside of the cores since the embedded cores usually have IEEE 1500 wrappers and they can be used to configure the bypass paths. Figure 6 (a) shows an example of the bypass path using IEEE 1500 wrapper for the core used in Fig. 1 (a). In this example, a 4-bit bypass path from input wrapper boundary cells to output wrapper boundary cells is added outside of the core. Similarly, we can consider the multi-core bypass path (bypass path from a core input to another core output) by using IEEE 1500 wrappers as shown in Fig. 6 (b). In this paper, we use the term "single/multi-core bypass path" for the path implemented by IEEE 1500 wrappers while the term "transparent path" denotes the path implemented by embedded multiplexers. In transparency-based TAM design, we should select paths used as TAM accordingly to the the cost of the single/multi-core bypass paths and transparent paths. For example, it may happen that the cost of a set of single-core bypass paths is higher than that of a multi-core bypass path due to the layout related issue and so on. Therefore, considering the transparent paths as well as bypass paths is important and effective for cost optimization.

Before describing the proposed graph modeling and ILP modeling, we formally present the transparency-based TAM design problem P_{TTAM} as follows.

Definition 1: *P_{TTAM}: Given a set of cores, a set of interconnects, a set of multi-core bypass paths, and for each core a set of possible transparent paths and single-core bypass paths. Furthermore, given a cost for 1-bit increase of each interconnect, multi-core bypass path, transparent path and single-core bypass path. Determine a transparency-based TAM such that the total cost including the core-level cost for transparency and the system-level cost for additional interconnects is minimized.*

Deriving the cost for each transparent and bypass path and preparing effective multi-core bypass paths are important tasks. However, as shown in the above problem formulation, we assume that all the information is given to us, and how to derive the costs and how to prepare the effective multi-core bypass paths are out of scope of this paper.

3.2 Graph Modeling

In this section, we introduce an extended system graph for TAM design such that the above three points can be taken into consideration during optimization. First of all, we break all cycles in the system by solving the minimum feedback vertex set problem in similar way to [23] shown in Sect. 2. Next, we construct an extended weighted directed acyclic system graph $G = (V, E, w)$ as follows.

- $V = V_{port} \cup V_{fan} \cup \{v_{PI}\} \cup \{v_{PO}\}$ where
  V_{port}: the set of all input and output ports of cores,
  V_{fan}: the set of all fanout points,
  v_{PI}: the vertex corresponds to the primary inputs of the system, and
  v_{PO}: the vertex corresponds to the primary outputs of the system.
- $E = E_f \cup E_t \cup E_s \cup E_m$ where
  E_f: the set of all functional interconnections,
  E_t: the set of all transparent paths,
  E_s: the set of all single-core bypass paths, and
  E_m: the set of all multi-core bypass paths.
- w: $E \rightarrow Z$

If $e \in E_f$, then $w(e)$ denotes the width of the functional interconnect. Otherwise, $w(e)$ is equal to 0. The extended acyclic system graph G corresponds to Fig. 3 is shown in Fig. 7. In Fig. 7, we consider two multi-core bypass paths which correspond to e_{24} and e_{25}, respectively.

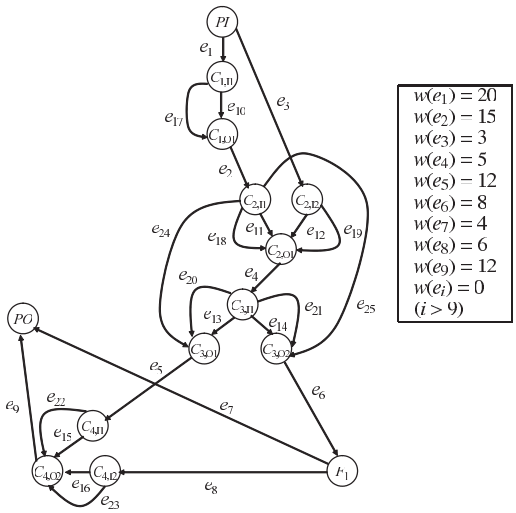


Fig. 7 Extended system graph G.

The edges correspond to functional interconnects and transparency paths (i.e., $e \in E_f \cup E_t$) are shown as straight lines, and the edges correspond to bypass paths (i.e., $e \in E_s \cup E_m$) are shown as curved lines.

Then, we construct the same weighted directed graph $G^* = (V, E, w^*)$ as G except for the weights of edges. The new edge weights in G^* can be determined by solving the ILP problem formulated in the next section.

3.3 ILP Formulation

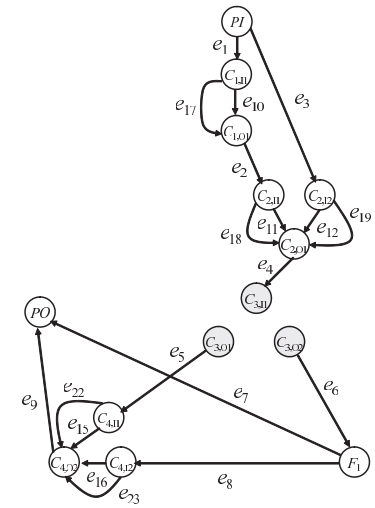
In this section, we present an ILP model to determine the new edge weights in G^* such that the total cost including the cost for transparency as well as the cost for additional interconnects is minimized.

The new edge weights in G^* can be determined by generating test graph for each core. For core C_j, the test graph $G_j = (V_j, E_j, w_j^*) \subseteq G^*$ contains vertices and edges reachable to(from) the input(output) ports of C_j. Figure 8 (a) shows the test graph for C_3. For each edge $e \in E_j$, $w_j^*(e)$ represents a test data flow on e to test C_j, and $w_j^*(e)$ and the new weight $w^*(e)$ in G^* are determined by solving the following ILP model. The main idea to avoid unnecessary constraints explained in Sect. 3.1 is to impose a set of constraints for each core under test using a different set of integer variables from the other cores.

Integer Variables:
$w_j^*(e)$: test data flow on e to test C_j
$w^*(e)$: new weight on e (final bit-width of interconnect e)

Constraints:
For each core C_j,

1. justification constraints: for each vertex $v \in V_j$ reachable to the input ports of C_j, the sum of the test data flows of the edges directed away from v must not exceed the sum of the test data flows of the edges incident on v shown as follows.



(a)

Constraints for C_3
Justification:
$w_3^*(e_4) \le w_3^*(e_{18}) + w_3^*(e_{11}) + w_3^*(e_{12}) + w_3^*(e_{19})$
$w_3^*(e_{18}) + w_3^*(e_{11}) \le w_3^*(e_2)$
$w_3^*(e_{12}) + w_3^*(e_{19}) \le w_3^*(e_3)$
$w_3^*(e_2) \le w_3^*(e_{17}) + w_3^*(e_{10})$
$w_3^*(e_{17}) + w_3^*(e_{10}) \le w_3^*(e_1)$
Propagation:
$w_3^*(e_5) \le w_3^*(e_{22}) + w_3^*(e_{15})$
$w_3^*(e_6) \le w_3^*(e_7) + w_3^*(e_8)$
$w_3^*(e_8) \le w_3^*(e_{16}) + w_3^*(e_{23})$
$w_3^*(e_{22}) + w_3^*(e_{15}) + w_3^*(e_{16}) + w_3^*(e_{23}) \le w_3^*(e_9)$
Test Data:
$w_3^*(e_4) \ge w(e_4)$
$w_3^*(e_5) \ge w(e_5)$
$w_3^*(e_6) \ge w(e_6)$
$w_3^*(e_i) \ge w(e_i) \times a$ for $i \ne 4, 5, 6$

(b)

Fig. 8 (a) Test graph G_3. (b) Constraints on the edge weights for C_3 in the proposed method.

$$\sum_{e \in E_v^{in}} w_j^*(e) \ge \sum_{e \in E_v^{out}} w_j^*(e) \quad \text{for} \quad \forall v \in V_j^{in} \tag{1}$$

where V_j^{in} is the set of vertices reachable to the input ports of C_j, E_v^{in} is the set of edges incident on v and E_v^{out} the set of edges directed away from v, respectively.

2. propagation constraints: for each vertex $v \in V_j$ reachable from the output ports of C_j, the sum of the test data flows of the edges incident on v must not exceed the sum of the test data flows of the edges directed away from v as follows.

$$\sum_{e \in E_v^{out}} w_j^*(e) \ge \sum_{e \in E_v^{in}} w_j^*(e) \quad \text{for} \quad \forall v \in V_j^{out} \tag{2}$$

where V_j^{out} is the set of vertices reachable from the output ports of C_j.

3. test data constraints: if $e \in E_j$ is the edge incident on the input ports of C_j or the edge directed away from the

output ports of C_j,

$$w_j^*(e) \geq w(e). \tag{3}$$

Otherwise,

$$w_j^*(e) \geq w(e) \cdot a \tag{4}$$

where a is a binary constant.

If we set the binary constant a to 1, we can represent the same constraints as [23] where every edge $e \in E_j$ is constrained by $w_j^*(e) \geq w(e)$. On the other hand, if we set a to 0, we can relax the constraints by considering the test data flows where only the edges $e \in E_j$ incident on the input ports of a core C_j under test and directed away from the output ports of C_j are constrained by $w_j^*(e) \geq w(e)$. The constraints on the test data flows for C_3 are shown in Fig. 8 (b).

Objective:

$$\text{Minimize: } Cost \tag{5}$$

$Cost$ is defined by the following three equations.

$$w^*(e) = \max_j \left(w_j^*(e)\right) \quad \text{for} \quad \forall e \in E \tag{6}$$

$$w^{max}(e) = \max\left(w^*(e), w(e)\right) \quad \text{for} \quad \forall e \in E \tag{7}$$

$$Cost = \sum_{e \in E} c(e) \cdot (w^{max}(e) - w(e)) \tag{8}$$

where $c(e)$ is a known constant for each e and denotes the cost for 1-bit increase of e.

The advantage of the proposed ILP model is that we can represent various situations including the same problem as [23] by setting $c(e)$. For example, the proposed ILP model can represent the same problem as [23] where the cost of transparency is 0 and we cannot consider the bypass paths by setting $c(e)$ as follows.

- $c(e) = 1$ if $e \in E_f$,
- $c(e) = 0$ if $e \in E_t$,
- $c(e) = \infty$ if $e \in E_s \cup E_m$

We can also consider the case where both the cost of additional interconnections and transparency are taken into account by setting $c(e) \neq 0$ for $e \in E_t$. An example setting for this case is shown as follows.

- $c(e) = 1$ if $e \in E_f$,
- $c(e) = 1$ if $e \in E_t$,
- $c(e) = \infty$ if $e \in E_s \cup E_m$

Moreover, by setting $c(e)$ as follows, we can consider the case where transparency cannot be achieved due to IP protection and only bypass paths are allowed to add.

- $c(e) = 1$ if $e \in E_f$,
- $c(e) = \infty$ if $e \in E_t$,
- $c(e) \neq \infty$ if $e \in E_s \cup E_m$

Of course, we can set $c(e)$ to different value for every $e \in E$ depending on the given SoC.

The number of variables for the ILP model are given by $|E| \cdot (N + 2)$ where N denotes the total number of cores ($N \cdot |E|$ for w_j^*, $|E|$ for w^* and $|E|$ for w^{max}). The number of constraints are given by $N \cdot (|V_{port} \cup V_{fan}| + |E|) + |E| \cdot (N + 2)$ where $|V_{port} \cup V_{fan}| + |E|$ for justification, propagation and test data flow constraints for each core, and $N \cdot |E|$ and $2|E|$ for linearizing the maximum functions in (6) and (7) of the proposed ILP model, respectively.

4. Experimental Results

In this section, we present experimental results for four SoCs. Fig. 2 (a) and Fig. 9 shows S_1 and S_2 we handcrafted, respectively. We used two benchmark SoCs $d695$ and $p93791$ from ITC'02 SOC Test Benchmarks [24]. As the original benchmark SoCs do not have any data on the connectivity between cores, we handcrafted the connectivity between cores for the SoCs. Figure 10 shows the handcrafted interconnects for $d695$ after breaking all the feedback loops. We denote the SoCs with the handcrafted interconnects as $d695^*$ and $p93791^*$. Furthermore, we consider two multi-core bypass paths for S_1 and S_2, and three multi-core bypass paths for $d695^*$ and $p93791^*$. The multi-core bypass paths for S_1 are shown in Fig. 7. The multi-core bypass paths for S_2 and $d695^*$ are shown as curved and dashed lines in Figs. 9 and 10, respectively. Table 1 summarizes the

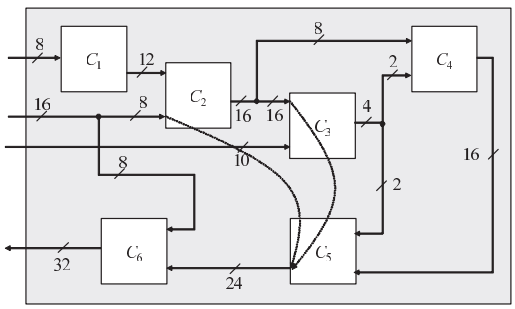


Fig. 9 An example SoC S_2.

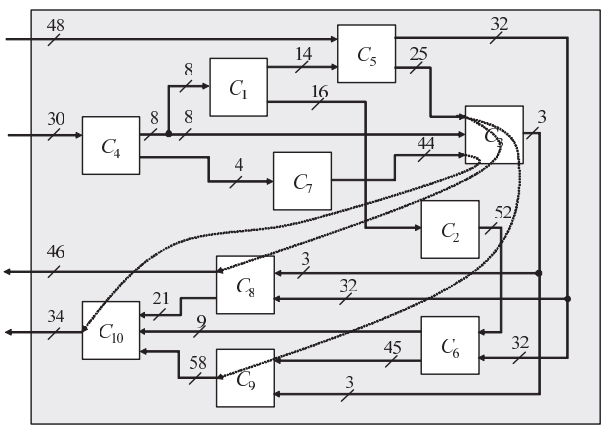


Fig. 10 SoC $d695^*$.

characteristics for the four SoCs. Column 2 and 3 denote the number of cores and interconnects. Column 4 and 5 denote the number of vertices and edges in the proposed graph modeling. Column 6 and 7 denote the number of variables and constraints in the ILP model we generated, respectively.

4.1 Comparison with the Previous Transparency-Based Approach [23]

First, in order to evaluate the effectiveness of the proposed method compared to [23], we made experiments for the following four cases for each SoC.

Case1: previous approach
Same optimization problem as [23] where (1) bypass paths are not allowed and (2) the objective is to minimize only the cost of additional interconnect area.

Table 1 Characteristics for four SoCs.

SoC	core	interconnect	graph model vertex	graph model edge	ILP model var	ILP model const
S_1	4	9	14	25	150	298
S_2	6	15	22	39	312	666
$d695^*$	10	23	38	79	948	2098
$p93791^*$	32	53	102	208	2098	16730

Case2: Case1 + constraint relaxation
Same optimization problem as Case1 except that the constraints are relaxed (i.e., $a = 0$ in Case2 while $a = 1$ in Case1).

Case3: Case2 + two-level optimization
Same optimization problem as Case2 except that the cost of making cores transparent is taken into consideration to minimize the total cost.

Case4: Case3 + multi-core bypass path
Same optimization problem as Case3 except that the multi-core bypass paths is taken into consideration.

Furthermore, we used two different cost settings for every case: (1) $c(e) = 1$ and (2) $c(e) = random$. $c(e) = 1$ means that we used the same cost "1" for every edge, and $c(e) = random$ means that we used an uniform random integer in the range of 1 to 8 for each edge.

Table 2 shows the results for the four SoCs, and We used the *lp_solve* package from Eindhoven University of Technology [25] on a PC with a Pentium IV 3.2 GHz processor and 2 GB memory for all the experiments. In Table 2, column 4, 5 and 6 denote the core-level cost, system-level cost and total cost, respectively. Column 7 represents the reduction ratio relative to Case1. Column 8 denotes the CPU time spent to solve the ILP problems. We can observe the

Table 2 Results for four SoCs.

		Case	Cost core-level	Cost system-level	Cost total	Reduction (%) (relative to Case1)	CPU (s)
S_1	$c(e) = 1$	1	73	23	96		0.02
		2	63	17	80	16.67	0.00
		3	63	17	80	16.67	0.02
		4	37	17	54	43.75	0.02
	$c(e) = random$	1	368	100	468		0.02
		2	328	82	410	12.39	0.02
		3	328	82	410	12.39	0.02
		4	224	70	294	37.18	0.03
S_2	$c(e) = 1$	1	116	44	160		0.02
		2	104	22	126	21.25	0.03
		3	100	22	122	23.75	0.02
		4	90	16	106	33.75	0.03
	$c(e) = random$	1	410	44	454		0.03
		2	388	22	410	9.69	0.02
		3	372	22	394	13.22	0.03
		4	341	28	369	18.72	0.03
$d695^*$	$c(e) = 1$	1	838	635	1473		0.11
		2	707	289	996	32.38	0.11
		3	655	293	948	35.64	0.13
		4	540	288	828	43.79	0.13
	$c(e) = random$	1	3412	3370	6782		0.13
		2	2958	1572	4530	33.21	0.13
		3	2772	1572	4344	35.95	0.11
		4	2262	1292	3554	47.60	0.13
$p93791^*$	$c(e) = 1$	1	3337	1500	4837		6.77
		2	2660	574	3234	33.14	5.41
		3	2463	574	3037	37.21	5.47
		4	2240	578	2818	41.74	5.38
	$c(e) = random$	1	12891	6761	19652		5.98
		2	10463	2885	13348	32.08	5.48
		3	9960	2904	12864	34.54	4.14
		4	8949	2481	11430	41.84	4.92

Table 3 Dedicated TAM design results for *d*695 in [3] and the number of added TAM wires.

TAM width (bits)	number of TAMs	width partition	core assignment	test time (cycles)	added TAM wires w_{TAM} (bits)
8	2	4 + 4	(1, 1, 1, 1, 1, 2, 1, 1, 2, 1)	83580	48
12	3	3 + 4 + 5	(1, 1, 2, 2, 3, 2, 3, 3, 1, 1)	56329	51
16	3	3 + 5 + 8	(2, 1, 1, 1, 3, 2, 1, 2, 3, 3)	42568	67

effect of relaxing the constraints by comparing Case2 with Case1. We can reduce the total cost up to 33%. By considering both the core-level cost for transparency and the system-level cost for additional interconnect area simultaneously (Case3), we can further reduce the total cost. Furthermore, we can reduce the total cost up to 47.6% by taking the multi-core bypass paths into consideration (Case4). In these results, we did not show the case where cores cannot be made transparent and bypass paths are allowed to add. However, the same results can be obtained for every case in terms of the total cost if we exchange $c(e)$ for $e \in E_t$ with $c(e)$ for $e \in E_s$. Regarding the CPU time, we can solve the proposed ILP model less than 7 seconds even for $p93791^*$ which is the biggest SoC in terms of the number of cores in ITC'02 SOC Test Benchmarks.

4.2 Comparison with Dedicated TAM Approach

Next, we evaluate the effectiveness of the proposed method compared to dedicated TAMs such as *TestBus* [11], [12] and *TESTRAIL* [5] with IEEE 1500 wrapper in terms of the cost and test time. In order to make a fair comparison between the proposed transparency-based TAM and the dedicated TAMs, we assume that every core has IEEE 1500 wrapper and the core-level costs for both approaches are the same, and then compare the test time under the same system-level cost constraint. For the above comparison, we need to explain (1) how to evaluate the system-level cost for the dedicated TAMs, and (2) how to evaluate the test time for the proposed transparency-based TAM.

(**system-level cost for dedicated TAM**) In order to evaluate the system-level cost of dedicated TAM approaches, we need the detailed results of TAM designs. Since [3] provided the detailed information about dedicated TAM designs and core assignments for *d*695, we have decided to compare the results with them. Table 3 shows the results provided in [3] and Fig. 11 shows an example of TAM design when we use 12-bit TAM. From the information, we calculate the number of added TAM wires w_{TAM} as follows and put them in the last column in Table 3

$$w_{TAM} = \sum_{b \in B} w_b \cdot (n_b + 1) \tag{9}$$

where B denotes the set of test buses, and w_b and n_b denote the bit-width of test bus b and the number of cores assigned to b, respectively. Here, we considered that each core has an input bus from either PI or previous core and an output bus to either PO or next core on the same bus, and those buses are individual. For example, we have three test buses with widths 3, 4 and 5 in Fig. 11. We considered that there are 5, 4 and 4 individual bus lines on each test bus, and the total number of added TAM wires is 51.

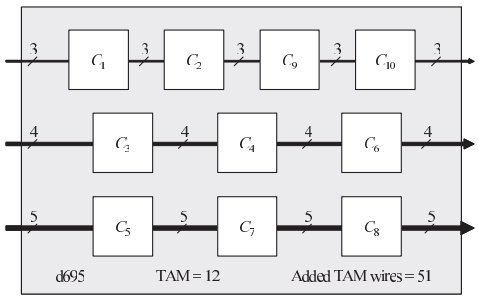


Fig. 11 Dedicated test bus design with 12-bit TAM for *d*695.

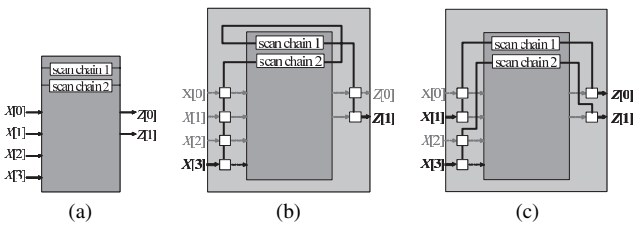


Fig. 12 Wrapper for transparency-based SoC test. (a) Original core. (b) Wrapper for *INTEST* with 1-bit access. (c) Wrapper for *INTEST* with 2-bit access.

(**test time for the proposed TAM**) In order to evaluate the test time for the proposed TAM, the following two points must be taken into consideration.

1. Only the serial test is possible (i.e., cores can be tested one by one) since the objective of the proposed transparency-based TAM design is to minimize the overall cost, not to minimize the test time.
2. We don't need to provide the complete transparent access for each core under test since we assumed that every core has IEEE 1500 wrapper.

Regarding the point (2), in Sect. 4.1, we showed the results where we provided the complete transparent access for every functional port using test data constraints (Eq. (4)) in the proposed ILP model. For example, for C_2 in S_2 shown in Fig. 9, we provided 20-bit and 16-bit transparent access for justification and propagation, respectively. However, we don't need to provide the complete transparent access when we assume IEEE 1500 wrapper for each core under test. We can select any bit-width of transparent access to test the core in the similar way to the dedicated TAM approaches. Fig. 12 (b) and (c) show examples of the wrapper configurations for *INTEST* with 1-bit and 2-bit transparent access, respectively. In order to evaluate the test time, we assume

Table 5 Comparison with dedicated TAM approach [3] for $d695$.

scenario	dedicated TAM [3] system-level cost	dedicated TAM [3] test time	proposed (Case2) system-level cost	proposed (Case2) test time	test time reduction (%)
1. $c(e_{tam}) = c(e_f) = 1$	48	83580	46	66706	20.19
	51	56329	51	51642	8.32
	67	42568	63	45453	−6.78
2. $c(e_{tam}) = 2 \times c(e_f) = 2$	96	83580	93	45453	45.62
	102	56329	99	45453	19.31
	134	42568	129	45268	−6.34

Table 4 Cost and test time for $d695^*$.

transparent access for each core (bits)	system-level cost ($c(e_f) = 1$)	test time (cycles)
1	0	659700
2	0	330344
3	0	224874
4	1	167050
5	3	135304
6	5	115734
7	8	102265
8	11	89388
9	16	81684
10	21	77094
11	26	71002
12	31	68663
13	36	67198
14	41	66779
15	46	66706
16	51	51642
17	57	49474
18	63	47707
19	69	46354
20	75	45695
21	81	45527
22	87	45526
23	93	45453
24	99	45453
25	105	45342
26	111	45269
27	117	45268
28	123	45268
29	129	45268
30	135	45195
31	142	45195
32	149	37687

that each core under test requires the same width of transparent access. Then, we design the transparency-based TAM by Case 2 with $c(e) = 1$ for $e \in E_f$ in Sect. 4.1 to minimize the system-level cost, and calculate the test time by summing up the test time of each core since only the serial test is possible. Table 4 shows the system-level cost and the test time for each bit-width of transparent access for $d695^*$.

Finally, we compare the test time of the proposed TAM and the dedicated TAM [3] in Table 5. In this comparison, we considered the following two scenarios, and used the system-level cost of the dedicated TAM shown in Column 2 as the constraint for the proposed TAM.

Scenario1: The cost for functional interconnects and the cost for dedicated test buses are the same ($c(e_f) = c(e_{tam}) = 1$ for $e_f \in E_f$ and every test bus line e_{tam}).

Scenario2: The cost for dedicated test buses is two times as high as that for functional interconnects ($c(e_{tam}) = 2 \times c(e_f) = 2$).

This is because our proposed approach increases the bit-width of functional interconnects while [3] adds the dedicated test buses, and floor-plan for SoCs is usually optimized for its functional operation. Therefore, the cost for functional interconnects might be smaller than that for dedicated test buses.

In Table 5, Columns 3 and 4 denote the system-level cost and the corresponding test time of [3] for each TAM width in the above two scenarios. Columns 5 and 6 denote the system-level cost and the corresponding test time for the proposed method when the value in Column 3 is used as the system-level cost constraint. First of all, remind that our proposed method does not consider test scheduling problem and the test time is calculated assuming the serial test schedule while the dedicated TAMs in [3] were designed to minimize test time. Even though the proposed method does not consider the test time optimization, it can reduce the test time up to 20% and 45% in Scenario1 and Scenario2, respectively, for the case with 8-bit TAM. Even in case of 16-bit TAM, the propose method incurs only 6 to 7% test time overhead. This shows the potential of the proposed method for test time optimization, and by considering the test scheduling problem in our proposed test framework, we will further reduce the test time while keeping the advantage of low cost TAM design.

Furthermore, the dedicated TAM approach always requires additional system-level cost, and the minimum system-level cost is $N + 1$ where N is the number of cores in the SoC (i.e., all cores are assigned to 1-bit single TAM and tested serially). For $d695^*$, the minimum cost is 11. On the other hand, the proposed method can produce solutions with a cost less than $N + 1$. For $d695^*$ shown in Table 4, the proposed method can provide a TAM design with a cost less than 11, and even without paying the system-level cost. This is another advantage of the proposed method compared to the dedicated TAM approach.

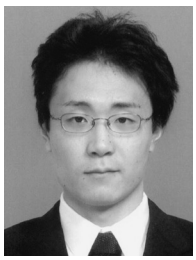
5. Conclusion

We proposed a graph model and an ILP model for TAM design for transparency-based SoC testing. The proposed method is an extension of [23] so that not only the cost for system-level interconnect area but also the cost for transparency can be simultaneously taken into consideration during optimization process. We also extended it to be able

to handle the case where cores cannot be made transparent due to IP protection. Moreover, we relaxed the constraints by considering the test data flows in the proposed ILP formulation. Therefore, the proposed ILP model is flexible in the sense that it can represent various problems including [23] by setting constant parameters. The experimental results demonstrated the flexibility and effectiveness of the proposed method compared to the previous approaches.

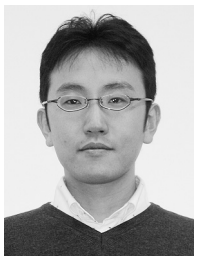
References

[1] Y. Zorian, E.J. Marinissen, and S. Dey, "Testing embedded-core based system chips," Proc. International Test Conference, pp.130–143, Oct. 1998.
[2] IEEE Computer Society, "IEEE standard testability method for embedded core-based integrated circuits," IEEE Std 1500-2005, 2005.
[3] 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, no.2, pp.213–230, April 2002.
[4] W. Zou, S.R. Reddy, I. Pomeranz, and Y. Huang, "SOC test scheduling using simulated annealing," Proc. VLSI Test Symposium, pp.325–329, May 2003.
[5] S.K. Goel and E.J. Marinissen, "Effective and efficient test architecture design for SOC," Proc. International Test Conference, pp.529–538, Oct. 2002.
[6] Y. Huang, W.T. Cheng, C.C. Tsai, N. Mukherjee, O. Samman, Y. Zaidan, and S.M. Reddy, "Resource allocation and test scheduling for concurrent test of core-based SOC design," Proc. Asian Test Symposium, pp.265–270, Nov. 2001.
[7] V. Iyengar, K. Chakrabarty, and E.J. Marinissen, "On using rectangle packing for SOC wrapper/TAM co-optimization," Proc. VLSI Test Symposium, pp.253–258, April 2002.
[8] Y. Huang, N. Mukherjee, S. Reddy, C. Tsai, W.T. Cheng, O. Samman, P. Reuter, and Y. Zaidan, "Optimal core wrapper width selection and SOC test scheduling based on 3-dimensional bin packing algorithm," Proc. International Test Conference, pp.74–82, Oct. 2002.
[9] Y. Xia, M.C. Jeske, B. Wang, and M. Jeske, "Using distributed rectangle bin-packing approach for core-based SoC test scheduling with power constraints," Proc. International Conference on Computer-Aided Design, pp.100–105, Nov. 2003.
[10] E. Larsson, K. Arvidsson, H. Fujiwara, and Z. Peng, "Efficient test solutions for core-based designs," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.23, no.5, pp.758–775, May 2004.
[11] T. Ono, K. Wakui, H. Hikima, Y. Nakamura, and M. Yoshida, "Integrated and automated design-for-testability implementation for cell-based ICs," Proc. Asian Test Symposium, pp.122–125, Nov. 1997.
[12] P. Varma and S. Bhatia, "A structured test re-use methodology for core-based system chips," Proc. International Test Conference, pp.294–302, Oct. 1998.
[13] C.A. Papachristou, F. Martin, and M. Nourani, "Microprocessor based testing for core-based system on chip," Proc. Design Automation Conference, pp.586–591, June 1999.
[14] P. Harrod, "Testing reusable IP - a case study," Proc. International Test Conference, pp.493–498, Sept. 1999.
[15] J.R. Huang, M.K. Iyer, and K.T. Cheng, "A self-test methodology for IP cores in bus-based programmable SoCs," Proc. VLSI Test Symposium, pp.198–203, April 2001.
[16] E. Cota, M. Kreutz, C.A. Zeferino, L. Carro, M. Lubaszewski, and A. Susin, "The impact of NoC reuse on the testing of core-based systems," Proc. VLSI Test Symposium, pp.128–133, April 2003.
[17] C. Liu, Z. Link, and D. Pradhan, "Reuse-based test access and integrated test scheduling for network-on-chip," Proc. Design, Automation, and Test in Europe, pp.303–308, March 2006.
[18] M. Nourani and C.A. Papachristou, "Structural fault testing of embedded cores using pipelining," J. Electronic Testing: Theory and Applications, vol.15, no.1/2, pp.129–144, Aug.–Oct. 1999.
[19] S. Ravi, G. Lakshminarayana, and N.K. Jha, "Testing of core-based systems-on-a-chip," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.20, no.3, pp.426–439, March 2001.
[20] T. Yoneda and H. Fujiwara, "Design for consecutive testability of system-on-a-chip with built-in self testable cores," J. Electronic Testing: Theory and Applications Special Issue on Plug-and-Play Test Automation for System-on-a-Chip, vol.18, no.4/5, pp.487–501, Aug. 2002.
[21] T. Yoneda, T. Uchiyama, and H. Fujiwara, "Area and time co-optimization for system-on-a-chip based on consecutive testability," Proc. International Test Conference, pp.415–422, Sept. 2003.
[22] T. Yoneda, H. Takakuwa, and H. Fujiwara, "Power-constrained area and time co-optimization for SoCs based on consecutive testability," Proc. Asian Test Symposium, pp.150–155, Dec. 2005.
[23] K. Chakrabarty, "A synthesis-for-transparency approach for hierarchical and system-on-a-chip test," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.11, no.2, pp.167–179, April 2003.
[24] E.J. Marinissen, V. Iyengar, and K. Chakrabarty, "A set of benchmarks for modular testing of SOCs," Proc. International Test Conference, pp.519–528, Oct. 2002.
[25] *lp_solve* version 5.5 http://lpsolve.sourceforge.net/5.5/



Tomokazu Yoneda received the B.E. degree in information systems engineering from Osaka University, Osaka, Japan, in 1998, and M.E. and Ph.D. degree in information science from Nara Institute of Science and Technology, Nara, Japan, in 2001 and 2002, respectively. Presently he is an assistant professor in Graduate School of Information Science, Nara Institute of Science and Technology. His research interests are VLSI CAD, design for testability and SoC testing. He is a senior member of the IEEE.

Akiko Shuto received her Bachelor of Information Engineering degree from Hiroshima City University in 2006. She is currently with Sony Semiconductor Kyushu Co., Ltd.



Hideyuki Ichihara received his M.E. and Ph.D. degrees from Osaka University in 1997, 1999, respectively. He was a research scholar of University of Iowa, U.S.A. from February to July in1999. Since December 1999, he had been an assistant professor of Hiroshima City University, and he is currently an associate professor of the university. He received IEICE Best Paper Award 2004 and Workshop on RTL and High Level Testing 2004 Best Paper Award. His research interests are VLSI testing and design for testability. He is a member of the IEEE Computer Society.

Tomoo Inoue is a professor of Graduate School of Information Sciences, Hiroshima City University. His research interests include test generation and high-level synthesis and design for testability and dependability, as well as design and test of reconfigurable devices such as field-programmable gate arrays. He received the BE, ME and PhD degrees from Meiji University, Kawasaki, Japan, in 1998, 1990 and 1997, respectively. From 1990 to 1992, he was with Matsushita Electric Industrial Co., Ltd. From 1993 to 1999, he was an assistant professor of Graduate School of Information Science, Nara Institute of Science and Technology. In 1999, he joined School of Information Sciences, Hiroshima City University as an associate professor. Tomoo Inoue received WRTLT (Workshop on RTL and High Level Testing) 2004 Best Paper Award. He is a member of the IEEE Computer Society and IPSJ (Information Processing Society of Japan).

Hideo Fujiwara received the B.E., M.E., and Ph.D. degrees in electronic engineering from Osaka University, Osaka, Japan, in 1969, 1971, and 1974, respectively. He was with Osaka University from 1974 to 1985 and Meiji University from 1985 to 1993, and joined Nara Institute of Science and Technology in 1993. Presently he is a Professor at the Graduate School of Information Science, Nara Institute of Science and Technology, Nara, Japan. His research interests are logic design, digital systems design and test, VLSI CAD and fault tolerant computing, including high-level/logic synthesis for testability, test synthesis, design for testability, built-in self-test, test pattern generation, parallel processing, and computational complexity. He is the author of Logic Testing and Design for Testability (MIT Press, 1985). He received the IECE Young Engineer Award in 1977, IEEE Computer Society Certificate of Appreciation Awards in 1991, 2000 and 2001, Okawa Prize for Publication in 1994, IEEE Computer Society Meritorious Service Awards in 1996 and 2005, IEEE Computer Society Continuing Service Award in 2005, and IEEE Computer Society Outstanding Contribution Award in 2001. He served as an Editor of the IEEE Trans. Computers (1998–2002), Journal of Electronic Testing: Theory and Application (1989–2004), Journal of Circuits, Systems and Computers (1989–2004), VLSI Design: An Application Journal of Custom-Chip Design, Simulation, and Testing (1992–2005), and several guest editors of special issues of IEICE Transactions of Information and Systems. He is currently an advisory member of IEICE Trans. Information and Systems. Dr. Fujiwara is a fellow of the IEEE, a Golden Core member of the IEEE Computer Society, and a fellow of the IPSJ (the Information Processing Society of Japan).