IEEE 7th Workshop on RTL and High Level Testing (WRTLT'06), pp. 21-26, November, 2006.

An Optimal Test Bus Design for Transparency-based SoC Test

Tomokazu Yoneda†, Akiko Shuto‡*, Hideyuki Ichihara‡, Tomoo Inoue‡, and Hideo Fujiwara†

†Graduate School of Information Science, Nara Institute of Science and Technology
Kansai Science City, 630-0192, Japan
{yoneda, fujiwara}@is.naist.jp
‡Graduate School of Information Science, Hiroshima City University
3-4-1, Ozuka-higashi, Asaminami-ku, Hiroshima, 731-3194, Japan
{ichihara, tomoo}@im.hiroshima-cu.ac.jp

Abstract

We present a graph model and an ILP model for optimal test bus design for transparency-based SoC testing. The proposed method is an extension of [12] so that not only the system-level cost but also the core-level cost can be simultaneously taken into consideration during the optimization process. We also relax the constraints by considering test data flows and extend it to be able to handle the case where cores can not be made transparent due to IP protection. The proposed ILP model can represent various problems including the same problem as [12] and produce better results. Experimental results show the effectiveness and flexibility of the proposed method compared to [12].

keywords: *SoC, test access mechanism, transparency, ILP*

1 Introduction

The systems-on-chip (SoC) design strategies help us to reduce the time-to-market and design cost for new products significantly. On the other hand, testing of SoCs gives rise to difficult and time consuming problems due to the increasing design complexity[1]. The main difficulty of SoC testing is how to propagate test data from(to) the outside of the SoC to(from) cores which are deeply embedded inside the SoC.

A number of approaches have addressed wrapper design [2, 3, 4] which are IEEE 1500 [5] compliant. Similarly, several test access mechanism (TAM) architectures have been proposed [6, 7, 8, 9, 10, 11, 12]. TAM architecture can be roughly classified into two types:1)TAM dedicated to test including *TestBus*[6, 7] and *TESTRAIL*[8], and 2)TAM not dedicated to test including the method re-using functional buses[13, 14, 15], the methods re-using functional networks[16, 17] and the methods based on transparency[9, 10, 11, 12]. The methods re-using functional buses and networks assume that cores are accessible by using the buses and the networks while the transparency-based method deal with SoCs without such direct accessible connections. Transparency-based TAM can be further classified into two types:1)single-cycle throughput transparency[11, 12] and 2)multi-cycle throughput transparency[9, 10]. 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. The authors in [11, 12], however, considered core-level transparency design problem and system-level TAM design problem separately. In other words, the authors first tackled only with 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 [12] 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 and extend it to be able to handle the case where cores can not be made transparent due to IP protection and so on. The proposed method consists of an extended graph modeling and an ILP formulation. We can represent various problems including the same problem as [12] by setting constant parameters in the proposed ILP model 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 [12] in Section 2. Section 3 shows the proposed optimal test bus design method for transparency-based SoC test including graph modeling and ILP modeling. Experimental results are discussed in Section 4. Finally, Section 5 concludes this paper.

2 Related Work[12]

In [12], 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 ($T = 1$).

*She is currently with Sony Semiconductor Kyushu Co., Ltd.

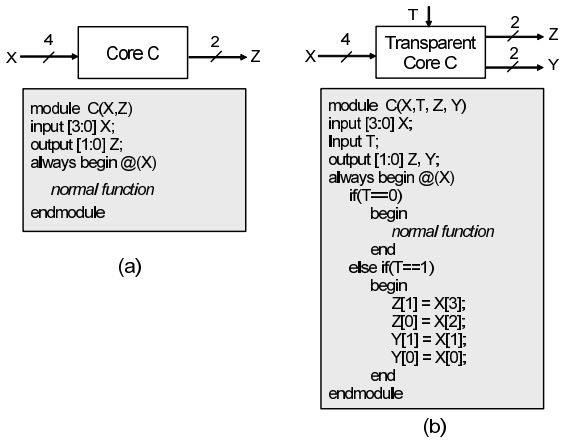

Figure 1. (a) Original circuit, (b) Circuit with embedded multiplexers

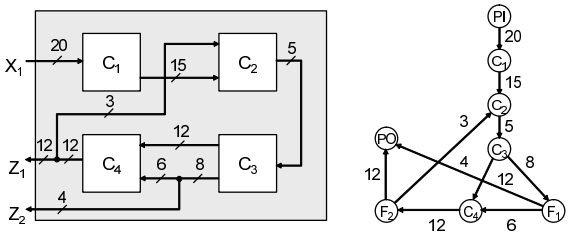

Figure 2. Example SoC S_1 and its system graph

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, [12] 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.

The method first constructs a weighted directed system graph whose vertices are the cores and fanout points of functional interconnects in the SoC and whose edges are functional interconnects between the cores. 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. 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.

Then, it constructs the same weighted directed graph G^* as G except for the weights of edges. The new edge weights can be determined by generating test graph for each core. For core C_j , the test graph $G_j \subseteq G^*$ 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_j) 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. Fig. 4(a) shows the test graph for C_3 in S_1. For each test graph G_j, it imposes

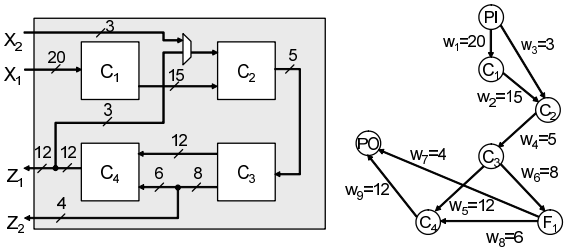

Figure 3. Acyclic SoC and its system graph G

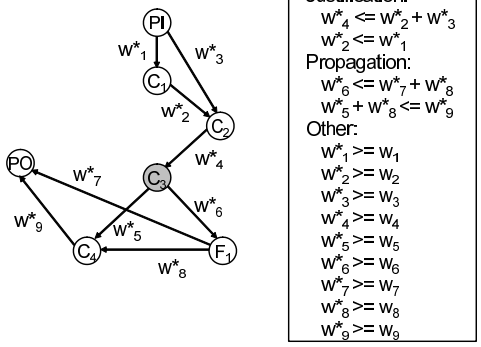

Figure 4. Test graph G_3 and constraints for C_3

a set of constraints on edge weights for the edges in G^* as follows.

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 Optimal Test Bus Design

3.1 Contributions

In this section, we present a graph modeling and ILP modeling method for optimal test bus design. The proposed method is an extension of [12] with respect to the following three points: (1)cost for transparency, (2) test data flow and (3)additional bypass path. Before describing the details

of graph and ILP modeling, we explain effectiveness of the above-mentioned extensions severally.

(**cost for transparency**) The method proposed in [12] considered to minimize only the overhead of additional interconnect area. In other words, the cost of making cores transparent is ignored during the optimization. However, we can get better solution if not only the cost of additional interconnect area but also the cost of making cores transparent are taken into account simultaneously. Fig. 5 is a simple example. We can satisfy the propagation constraints for C_1 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 cost for making C_2 1-bit transparent is 1, then the cost is 6. On the other hand, if we consider the cost of making C_2 transparent as well as the cost of additional interconnect area, then we can get better solution shown in Fig. 5(b) where the cost is 2. Moreover, if we assume that the cost for making C_2 1-bit transparent is 2 and the cost for 1-bit additional interconnect is 1, 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 of making cores transparent as well as the cost of additional interconnect area simultaneously during optimization.

(**test data flow**) In [12], for each edge, the weights of the edge in G must not exceed the weight of the edge in G^*, i.e., $w_i^* \geq w_i$. However, this is not always necessary in order to satisfy the justification and propagation constraints for each core. For example, in Fig. 4, we need $w_5^* + w_8^* \leq w_9^*$ as a propagation constraint for C_3. If we impose the constraint $w_i^* \geq w_i$ upon every edge, then, the cost to satisfy the propagation constraints is 6 ($w_5^* = 12, w_6^* = 8, w_7^* = 4, w_8^* = 6, w_9^* = 18$). Indeed, for each core, the edges directed away from the core vertex and the edges incident on the core vertex must be subject to the constraints $w_i^* \geq w_i$ in order to justify/propagate test patterns/response to/from the cores. On the other hand, however, other edges may have bus width enough for the test data flow (or the justification and propagation) of the core, and accordingly they can be under no such constraints during the test. In this example, we only need 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. ($w_5^* = 12, w_6^* = 8, w_7^* = 4, w_8^* = 4, w_9^* = 16$).

(**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, time for synthesis and layout etc.) of making cores transparent by embedding multiplexers is higher than that of bypass paths added outside the cores by non-embedded multiplexers. Moreover, it may happen that the cost of single-core bypass path (bypass path from a core input to the core output) is higher than that of multi-core bypass path (bypass path from a core input to another core output)) 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.

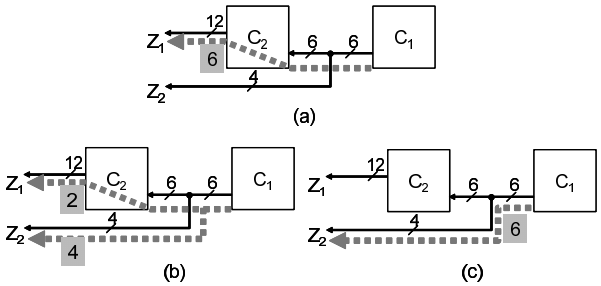

Figure 5. Cost for transparency

3.2 Graph Modeling

In this section, we introduce an extended system graph for optimal test bus 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 [12] shown in Section 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. 6. In Fig. 6, we consider two multi-core bypass paths which correspond to e_{24} and e_{25}, respectively. 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

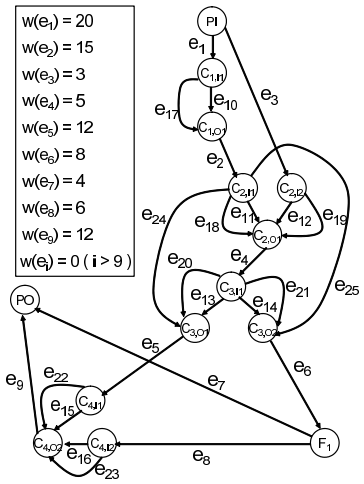

Figure 6. Extended system graph G

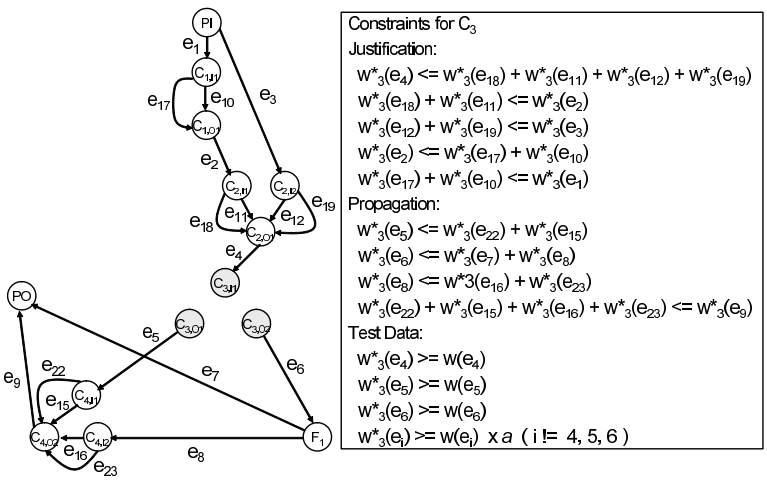

Figure 7. Test graph G_3 and constraints for C_3

ILP problem formulated in the next section.

3.3 ILP Formulation

In this section, we present an ILP model for optimal test bus design. The new edge weights 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. For each edge $e \in E_j$, $w_j^*(e)$ represents a test data flow on e for C_j. For each test graph C_j, we impose a set of constraints on test data flows for the edges in E_j. These constraints are of three types: (1)justification constraints, (2)propagation constraints and (3)test data constraints shown as follows.

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.
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.
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, then $w_j^*(e) \geq w(e)$. Otherwise, $w_j^*(e) \geq w(e) \times a$ where a is a binary constant.

If we set the binary constant a to 1, we can represent the same constraints as [12] 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. 7.

The total cost to satisfy the constraints is given by

$$Cost = \sum_{e \in E} c(e) \cdot (max\,(w^*(e), w(e)) - w(e)) \quad (1)$$

where $c(e)$ is a known constant for each e and denotes the cost for 1-bit increase of e and $w^*(e)$ denotes the maximum test data flow on e (i.e., $w^*(e) = \max_j(w_j^*(e))$). Therefore, the problem to minimize total cost can be expressed an ILP model where the objective is to minimize $Cost$ subject to the constraints on the edge weights.

The advantage of the proposed ILP model is that we can represent various situations including the same problem as [12] by setting $c(e)$. For example, the proposed ILP model can be represent the same problem as [12] 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.

4 Experimental Results

In this section, we present experimental results for SoC S_1 shown in Fig. 2 and SoC S_2 shown in Fig. 8 and Fig. 9.

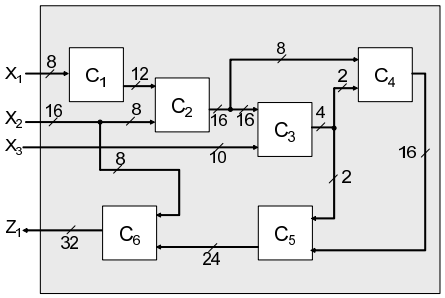

Figure 8. Example SoC S_2

We used the *lp_solve* package from Eindhoven University of Technology [18] for the experiments. We made experiments for the following four cases. All the experimental results can be obtained within 0.1 sec. on a SunBlade 2000 workstation (1.05 GHz with 8GB RAM).

Case1: Same optimization problem as [12] where (1)bypass paths are not allowed and (2)the objective is to minimize only the cost of additional interconnect area.

Case2: Same optimization problem as Case1 except for relaxing the constraints (i.e., $a = 0$ in Case2 while $a = 1$ in Case1).

Case3: Same optimization problem as Case2 except for taking the cost of making cores transparent into consideration to minimize the total cost.

Case4: Same optimization problem as Case3 except for taking the multi-core bypass paths into consideration.

Table 1 shows the results for S_1 where we set the same cost (i.e., $c(e) = 1$) to every edge e except multi-core bypass paths ($e \in E_m$). In Case1 and Case2, the number in parentheses denotes the cost of additional interconnect area and the number outside parentheses denotes the total cost including the cost of making cores transparent, respectively. We can observe the effect of relaxing the constraints by comparing Case2 with Case1. We can reduce the cost of additional interconnect area from 23 to 17. Although the effect of considering the cost for transparency as well as the cost for additional interconnect area simultaneously cannot be observed (compare Cases 2 and 3), taking two multi-core bypass paths into consideration (Case4) achieved a significant reduction in the total cost to 74. We can observe noticeable trends for S_2 in Table 2. In particular, by comparison Cases 2 and 3, we can observe the effect of considering both the core-level cost for transparency and the system-level cost for additional interconnect area simultaneously. In these results, we did not show the case where cores can not 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 $e \in E_b$.

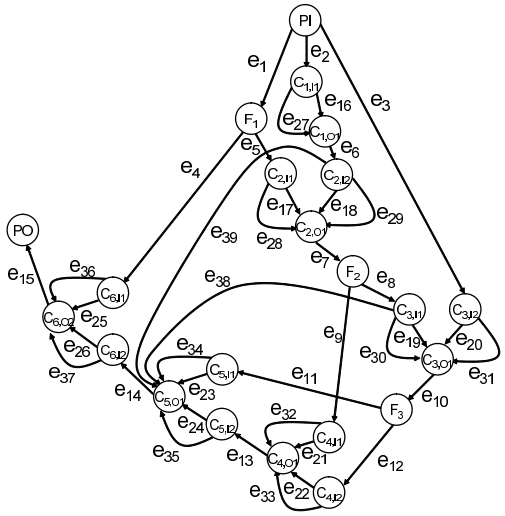

Figure 9. Extended system graph for S_2

5 Conclusion

We proposed a graph model and an ILP model for optimal test bus design for transparency-based SoC testing. The proposed method is an extension of [12] so that not only the cost for system-level interconnet 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 can not 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 [12] by setting constant parameters, and it can produce better results compared to [12] in terms of total cost.

Acknowledgments

This work was supported in part by Japan Society for the Promotion of Science (JSPS) under Grants-in-Aid for Scientific Research B(2)(No. 15300018) and for Young Scientists(B)(No.18700046).

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] V. Iyengar, K. Chakrabarty and E. J. Marinissen, "Test Wrapper and test access mechanism co-optimization for system-on-chip," Journal of Electronic Testing: Theory and Applications, pp. 213–230, Apr. 2002.

[3] W.Zou,S.R.Reddy,I.Pomeranz and Y.Huang, "SOC Test Scheduling Using Simulated Annealing," Proc. 21th VLSI Test Symposium,pp. 325–329,May 2003.

[4] E. J. Marinissen, S. K. Goel, and M. Lousberg, "Wrapper Design for Embedded Core Test," Proc. International Test Conference, pp. 911–920, Oct. 2000.

Table 1. Results for S_1

parameter setting					
		Case1	Case2	Case3	Case4
$c(e_i)$	$e_i \in E_f$ $(1 \le i \le 9)$	1	1	1	1
	$e_i \in E_t$ $(10 \le i \le 16)$	0	0	1	1
	$e_i \in E_s$ $(17 \le i \le 23)$	∞	∞	∞	∞
	$e_{24} \in E_m$	∞	∞	∞	4
	$e_{25} \in E_m$	∞	∞	∞	2
a		1	0	0	0

resulting value				
edge e_i	$w^*(e_i) - w(e_i)$			
e_1	0	0	0	0
e_2	2	0	0	0
e_3	0	0	0	0
e_4	15	13	13	7
e_5	0	0	0	0
e_6	0	0	0	0
e_7	0	4	4	4
e_8	0	0	0	0
e_9	6	0	0	0
e_{10}	17	15	15	15
e_{11}	17	15	15	9
e_{12}	3	3	3	3
e_{13}	12	12	12	12
e_{14}	8	6	6	0
e_{15}	12	12	12	12
e_{16}	6	0	0	0
from e_{17} to e_{23}	0	0	0	0
e_{24}	0	0	0	0
e_{25}	0	0	0	6
Total Cost	98(23)	80(17)	80	74

Table 2. Results for S_2

parameter setting					
		Case1	Case2	Case3	Case4
$c(e_i)$	$e_i \in E_f$ $(1 \le i \le 15)$	1	1	1	1
	e_{16}	0	0	1	1
	e_{17}	0	0	2	2
	e_{18}	0	0	2	2
	e_{19}	0	0	1	1
	e_{20}	0	0	1	1
	e_{21}	0	0	8	8
	e_{22}	0	0	8	8
	e_{23}	0	0	4	4
	e_{24}	0	0	4	4
	e_{25}	0	0	4	4
	e_{26}	0	0	4	4
	$e_i \in E_s$ $(27 \le i \le 37)$	∞	∞	∞	∞
	$e_{38} \in E_m$	∞	∞	∞	4
	$e_{39} \in E_m$	∞	∞	∞	2
a		1	0	0	0

resulting value				
edge e_i	$w^*(e_i) - w(e_i)$			
e_1	4	0	0	0
e_2	4	4	4	4
e_3	0	0	0	0
e_4	0	0	0	0
e_5	4	0	0	0
e_6	0	0	0	0
e_7	8	0	0	0
e_8	0	0	0	0
e_9	0	8	8	4
e_{10}	12	4	4	2
e_{11}	6	6	6	0
e_{12}	6	0	0	2
e_{13}	0	0	0	0
e_{14}	0	0	0	0
e_{15}	0	0	0	0
e_{16}	12	12	12	12
e_{17}	12	8	4	8
e_{18}	12	12	12	8
e_{19}	16	0	0	4
e_{20}	0	10	8	4
e_{21}	8	16	16	12
e_{22}	8	0	0	4
e_{23}	8	8	8	0
e_{24}	16	16	16	16
e_{25}	0	0	0	0
e_{26}	24	24	24	24
from e_{27} to e_{37}	0	0	0	0
e_{38}	0	0	0	0
e_{39}	0	0	0	8
Total Cost	440(44)	404(22)	394	368

[5] E.J. Marinissen, R. Kapur, M. Lousberg, T. McLaurin, M. Ricchetti and Y. Zorian, "On IEEE P1500's Standard for Embedded Core Test," Journal of Electronic Testing: Theory and Applications, pp. 365–383, Aug. 2002.

[6] T. Ono, K. Wakui, H. Hikima, Y. Nakamura and M. Yoshida, "Integrated and automated design-for-testability implementation for cell-based ICs," Proc. 6th Asian Test Symposium, pp. 122–125, Nov. 1997.

[7] 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.

[8] E. Marinissen, R. Arendsen, G. Bos, H. Dingemanse, M. Lousberg and C. Wouters, "A structured and scalable mechanism for test access to embedded reusable cores," Proc. International Test Conference, pp. 284–293, Oct. 1998.

[9] M. Nourani and C. A. Papachristou, "Structural fault testing of embedded cores using pipelining," Journal of Electronic Testing:Theory and Applications 15(1-2), pp. 129–144, Aug. –Oct. 1999.

[10] S. Ravi, G. Lakshminarayana, and N. K. Jha, " Testing of core-based systems-on-a-chip," IEEE Trans. on CAD, Vol. 20, No. 3, pp. 426–439, Mar. 2001.

[11] T. Yoneda and H. Fujiwara, " Design for consecutive testability of system-on-a-chip with built-in self testable cores," Journal of Electronic Testing: Theory and Applications (JETTA) Special Issue on Plug-and-Play Test Automation for System-on-a-Chip, Vol. 18, No. 4/5, pp. 487-501, Aug. 2002.

[12] K. Chakrabarty, " A synthesis-for-transparency approach for hierarchical and system-on-a-chip test," IEEE Trans. on VLSI systems, Vol. 11, No. 2, pp. 167–179, Apr. 2003.

[13] C. A. Papachristou, F. Martin and M. Nourani, " Microprocessor based testing for core-based system on chip," Proc. IEEE Design Automation Conference, pp. 586–591, June 1999.

[14] P. Harrod, " Testing reusable IP - A case study," Proc. 1999 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, 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, 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, 2006.

[18] M.Berkelaar, *lp_solve*, Eindhoven University of Technology, The Netherlands, *ftp://ftp.ics.ele.tue.nl/pub/lp_solve*.