Optimization of Test Wrapper for TSV based 3D SOCs

Surajit Kumar Roy[1], Chandan Giri[2], Sourav Ghosh and Hafizur Rahaman[3]
Dept. of Information Technology
Bengal Engineering & Science University
Shibpur, Howrah - 711103, India
Email:[1]suraroy@gmail.com,[2]chandangiri@gmail.com, [3]rahaman_h@yahoo.co.in

***Abstract*—Embedded core-based three dimensional system-on-chip (3D SOC) is a new design paradigm in modern semiconductor industry. For testing of these 3D SOC efficient testing techniques are required and designing the test wrapper of a core is also an important issue in this respect. In this paper we have addressed a 1500-style wrapper optimization for 3D ICs using Through Silicon Vias (TSVs) as vertical interconnects. It is assumed that the core elements are spanned over several layers of 3D ICs. Here we are trying to design the test wrapper that reduces the testing time of the core. This work is intended to design balanced wrapper chains using available TSVs as there are an upper limit on the total number of TSVs due to small chip area. We propose a polynomial time algorithm of O(N) where N is number of wrapper elements to design the wrapper. Obtained results are presented based on the ITC'02 SOC test benchmarks. The results demonstrate that our algorithm has better performance with respect to both TSV utilization and test time for higher TAM width compared to [10].**

***Keywords*-3D System-on-Chip test; wrapper design; wrapper optimization; test access mechanism**

I. Introduction

Development of advanced semiconductor technologies enable integration of a complete system on a single chip called System-On-Chip (SOC) technology. SOC composed of different types of cores like processors, ROM, RAM, combinational logic, comparators etc. on a single chip. The cores are to be tested for manufacturing defects. As the cores are not directly accessible via I/O pins of the ICs, sophisticated techniques are required to test them and modular testing is an effective approach in this respect. For modular testing approach test access mechanism (TAM) and wrapper were introduced. TAM is used to transport the test data from the test source to the core under test as well as test responses from the core to the test sink. A typical test source provides the test vector to the core. Test sink is an on -chip signature analyzer or an output response comparison circuit. The modular testing is being done by designing the IEEE Std. 1500 [10] wrapper chain. Test wrapper chain is an interface between the core and TAM, that switches a core between functional mode and test mode and vice-versa. Wrapper chain of a core is designed by connecting the core elements like functional IOs, internal scan chains etc., which are called as core wrapper elements or wrapper elements. The testing time of a core is largely depends on wrapper design and TAM optimization, and hence they have great impact on overall SOC testing.

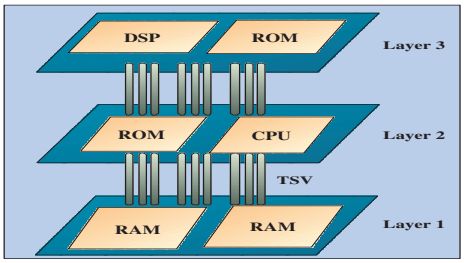

Figure 1. Example 3D ICs

Advancement in VLSI technology and technology scaling have made electronic system's design to shift from 2D design space to 3D design space. In 3D integrated circuit (3D IC) multiple device layers are stacked together and interconnecting them using through silicon vias(TSVs) as shown in Fig. 1. We use TSVs as vertical interconnect because it provides the greatest vertical density among all the vertical interconnect technologies.

3D design has good number of advantages over 2D design like increased performance, greatest on-chip data bandwidth, reduced power consumption and smaller footprint[4]. The main challenge for 3D integration is poor thermal conductivity and heat dissipation which results temperature rise due to the high power density.

The design of a 3D IC can be done at two levels of granularity [10]:

(a) Coarse-granularity partitioning: Here the embedded cores in the SOC is like a 2D design space.
(b) Fine-granularity partitioning: cores are partitioned in multiple layers [5] that shows significant improvement in performance and overall frequencies.

Although manufacturing of 3D ICs are now possible, but the tools and Design-for-testability(DfT) techniques for testing these 3D ICs are not available for commercial use.

In this paper, we are trying to design the wrapper of a core where the core wrapper elements are distributed over several layers of the 3D SOC and the layers are vertically connected through TSVs. The test application time of a core depends on longest wrapper chain. Also in 3D ICs number

978-0-7695-4570-7/11 $26.00 © 2011 IEEE
DOI 10.1109/ISED.2011.26

of available TSVs for test access is limited due to small chip area. Hence our objective is to create balanced wrapper chain to minimize the scan test time by distributing the internal scan chain and input/output of the cores in different layers of the SOC under the constraints of available number of TSVs.

The remainder of the paper are organized in the following manner. Section II discusses the previous works on 3D SOCs testing. Section III describes the problem formulation. Proposed algorithm to solve the wrapper optimization problem is presented in Section IV. An illustrative example of proposed solution is discussed in section V. Section VI presents the experimental results for several cores from ITC'02 benchmark. Finally, Section VII draws the conclusion.

II. Previous Works on Wrapper Design

Number of works in wrapper optimization and test infrastructure design for 2D SOCs [1], [6], [7] have been proposed in literature. These include ILP [1], bin packing [9],[7],Genetic Algorithm[7] and some other methods. These methods did not considered the design problem related to 3D technology.

There exists limited works on 3D SOC testing. TAM design for 3D SOC using integer linear programming(ILP) is proposed in [12] where test time is minimized under TSVs constraint. In [11] authors proposed optimization of scan chain length along with scan power using Genetic Algorithmic approach for 3D SOC. Jiang et al.[13] presented simulated annealing (SA) based algorithms to optimize the pre-bond and post-bond test time of 3D ICs. A heuristic method to reduce weighted test cost with constraints on test pin width in pre-bond and post bond are addressed in [14]. In [15], authors have discussed an optimization method to minimize the test time for a 3DSIC, either for the final stack test or for any number of multiple test insertions during bonding. Noia et al. have presented a 1500-style test wrapper for embedded cores in TSVs based 3D SOCs [10]. In [16], the authors have made an attempt to design 1500 Std. based test wrapper for 3D SOCs using optimum number of TSVs available for testing.

III. Problem Formulation

The purpose of wrapper design algorithm is to create a set of wrapper chains for each core where a wrapper chain includes internal scan chains of the core, input and output cells etc.. The main objective of the wrapper optimization is to organize the wrapper chains in such a way that test time is minimized. The problem can be formulated as follows: *Given a core with different functional parameter such as number of functional inputs, number of functional outputs, set of internal scan chains along with their length, TAM width, determine the distribution of the core elements into the wrapper chains such that the length of longest wrapper chain is minimized under the constraints of maximum number of TSVs available for this core.*

To solve the above problem, we have proposed a heuristic algorithm that tries to distribute the core wrapper elements over several layers of the 3D IC so that the length of the longest wrapper chain is minimized in polynomial time of $O(N)$, where N is the total number of core wrapper elements.

IV. Proposed Methodology

The goal of the proposed heuristic algorithm is to assign the core elements over several layers of IC into a number of wrapper chains using maximum number of available TSVs (TSV_{max}). Scan chains of a core can be distributed across multiple layer of 3D IC and hence the scan-in and scan-out part of wrapper chains also present over different layer. We assume the chip pins are at the lowest layer so, the wrapper chain begins and ends at lowest layer. We assume the lowest layer of the IC is layer number 1, followed by 2, 3 and up to L, the maximum number of layers present in the 3D IC. Now we are given the followings: a set of core wrapper elements $E = \{E_1, E_2, ..., E_{x+y+n}\}$, TAM width ($W_{max}$) and maximum number of available TSVs (TSV_{max}), which are used for routing of the wrapper chains. Here x is the number of functional inputs, y is the number of functional outputs and n is the number of internal scan chains of the core. Here it is considered that during routing of wrapper chains TSVs internal to the scan chains are not counted as in [10].

A. Data Structures

The proposed algorithm 1 have used *Wrapper* and *Element* data structures. The data structure *Element* is presented in Table I contains the information about each core wrapper element E_i of the list E. The information include the type of element (whether I/O wrapper cell or scan chain), layer number in which it is placed and the length of this wrapper element. Data structure *Wrapper*(shown in Table II) maintains three different lists using three variables corresponding to each type of wrapper element E_i. Information about the number of TSVs required at any instant of time (specifically before insertion of an wrapper element in to the wrapper chain) of designing the wrapper is stored in the variable No_of_tsv. The variables $wrapper\_length$ and $No\_of\_element$ are used to hold the length of the wrapper chain and the number of different types of wrapper elements contained by the wrapper chain respectively. The length of wrapper chains before starting of the algorithm are initialized with 0.

B. Heuristic Algorithm

The proposed algorithm tries to create a set of balanced wrapper chain $W = \{W_0, W_1, ..., W_T\}$ where T equal to

Type	Type of the i^{th} element
Layer	Layer number of the i^{th} element
Length	Length of the i^{th} element

Table I
Element DATA STRUCTURE

No_of_tsv	Total number of TSVs required to connect all the elements for a particular wrapper chain
No_of_element	Total number of element for a particular wrapper chain
Wrapper_length	Length of a wrapper chain
***incell**	Point to the list of input wrapper cells
***scanchain**	Point to the list of scan chains
***outcell**	Point to the list of wrapper output cells

Table II
Wrapper DATA STRUCTURE

maximum available TAM width and W_i is the i^{th} wrapper chain. We know that total number of wrapper chains must be equal to the available TAM width. For designing the wrapper chain the algorithm picks up a wrapper element and determines element type. Layer number of that chosen element is considered randomly. But for every wrapper chain we have calculated the required number of wrapper I/O cells where both the wrapper I/O cells are distributed almost equally among the wrapper chains because the assignment of wrapper I/O cells directly affect the number of TSVs required for a chain. So we add a wrapper input or output cell in a wrapper chain if it satisfies the calculated I/O wrapper cell requirement. Otherwise we have to find out another wrapper chain where the wrapper I/O cell requirement is satisfied. When we assign first wrapper element to any one of the wrapper chain, it is assumed that it must be placed at any layer except the lowest. The wrapper elements are assigned to the wrapper chain in clockwise fashion. For example, if four wrapper chains are to be designed then order will be $W_1 \rightarrow W_2 \rightarrow W_3 \rightarrow W_4$. Every time we have added a wrapper element into a wrapper chain, number of TSVs required for that wrapper chain is calculated. In this fashion, we keep on inserting element into the same wrapper chain until no extra TSV is required to connect this new element. If all the available TSVs are utilized, then rest of the elements will be added to the minimum length wrapper chain. This method will be continued until all the elements are assigned. Here we have not counted TSVs that are internal between two scan chains as the assumption made in [10].

C. Upper Bound on the TSVs required for a wrapper chain

A wrapper chain begins with the input cell and then scan chain followed by the output cell. Consider the wrapper chain configuration as shown in Fig. 2. Suppose the input cells are placed in layer 1, 2 and 3. To connect the input cells total number of TSVs required are 2. Then we assume that a

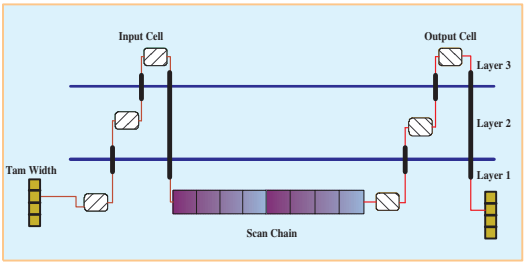

Figure 2. Typical wrapper chain configuration

scan chain located in the lowest layer. After connecting the scan chain with the input cell the total TSVs required are 3.Then we assume that output cells are located at layer 1,2 and 3. After connecting the output cells the total number of TSVs required are 5. Now wrapper chain ends in the lowest layer thus total number of TSVs required are 6. TSVs internal to the scan chains are not considered here. We can add any other elements in the wrapper chain, but no additional TSVs are required. Therefore maximum number of TSVs utilization of any wrapper chain is 6 for three layers core. Similarly if there are four layers core then maximum number of TSVs utilization of any wrapper chain is 8. If all the wrapper elements are in the lowest layer then no TSV is required to design wrapper chain. Thus minimum number of TSVs utilization of any wrapper chain is 0. If all the wrapper chains acquire equal number of TSVs K, therefore we can write $T \times K = TSV_{max}$ and $0 \leq K \leq 2L$ where TSV_{max} is total number of TSVs, T is the TAM width and L is the number of layers in the core.

D. Analysis of the Algorithm

The Createchain() algorithm will attempt to place all N elements in W wrapper chains. According to the algorithm we can add an element to a wrapper chain depending on calculated input cell and output cell requirement. In the best-case every time the I/O cell requirement of the wrapper chain is satisfied and the element is placed in the current chain, so the time complexity is $O(N)$.

If for a I/O cell the I/O requirement of the current wrapper chain is not satisfied, then we have to find out another wrapper chain where the requirement is satisfied. In the worst case scenario we have to search $(W-1)$ wrapper chains to get the desired wrapper chain where the I/O cell is to be placed. There is no requirement check for the scan chain. So we have to check the I/O requirement for $(N-n)$ elements where n is the total number of scan chain. Hence the worst case complexity is $O(N + (W-1) * (N-n))$. As the number of elements in a core is much larger than the width of the corresponding core wrapper that is $W << N$, the algorithm has a the worst case time complexity of $O(N)$.

In the average case half of the cells satisfy the I/O cell requirement of the current wrapper chain. Therefore the

complexity in the average case is $O(N+(N-n)/2*(W-1))$. As said previously $W << N$ thus the average case time complexity is also $O(N)$. Hence the Createchain() runs in polynomial time in N.

V. Illustrative Example

Now we elaborate the algorithm with an example. Here we want to design a wrapper of a core with 10 functional inputs (x), 10 functional outputs(y) and 5 internal scan chains(n) each of length 8, the TAM width is 4 and the number of maximum available TSVs are 14. As the number of wrapper chains is equal to TAM width, $25(x+y+n)$ wrapper elements are to be distributed among 4 wrapper chains. Distributing the wrapper cell almost equally among the wrapper chains we have calculated apriori the required number of I/O wrapper cells for each of the wrapper chains. So after distribution of all the wrapper I/O cells over 4 wrapper chains, each chain contains 3, 3, 2 and 2 number of wrapper input cells and 2, 2, 3 and 3 the number of output wrapper cells respectively. We use these requirements to satisfy TSV constraint because number of wrapper I/O cells directly affects the number of TSVs. During the assignment of I/O wrapper cells into the wrapper chains the requirement is checked, but for scan chains no requirement check is made.

The algorithm randomly chooses different wrapper elements $E_i \epsilon E$ and assign it to $W_j \epsilon W$ where $E = \{E_1, E_2, ..., E_{25}\}$ and wrapper chain $W = \{W_1, W_2, W_3, W_4\}$. Let us consider the first chosen element E_i is of type internal scan chain and it is placed in layer number 2. As it is scan chain so no requirement check is done and the element is assigned to the first wrapper chain W_1 as shown in Fig. 3(a). The wrapper chains begin and end at the lowest layer because all of the chip pins are at the lowest layer. Hence, after placement of the internal scan chain at layer number 2 for W_1, number of TSVs required is 2(one between layer number 1 and layer number 2 and another one between layer number 2 and layer number 1). According to the proposed algorithm the next wrapper element that will be picked from E, must be assigned to the next wrapper chain except W_1 where it will satisfy the condition of apriori calculated required wrapper I/O cells. Suppose the next wrapper element we have picked is a wrapper input cell and it is placed in layer number 2. As the calculated input cell requirement of wrapper chain W_2 is 3 and number of input cells in W_2 is 0, the input cell requirement condition is satisfied for W_2.Thus the currently selected input cell must be placed in wrapper chain W_2. Here also TSVs required for W_2 are 2. In the same way, the wrapper elements are assigned to the other two wrapper chains W_3 and W_4. At this point each wrapper chain contains exactly one wrapper element as shown in Fig. 3(a), and total number of TSVs required up to this stage is 8. This is the completion of first cycle of assigning wrapper elements to every wrapper chain and in a clockwise fashion the next element will be assigned to the wrapper chain W_1.

In the next stage, we picked a wrapper input cell as the next wrapper element and placed it at layer 1 as shown in Fig.3(b). Hence no extra TSV is required for the assigned wrapper input cell because it is in lowest layer. As the number of TSVs before and after the assignment is same, the next wrapper element picked is again added to the same wrapper chain W_1. Similarly, we have assigned one internal scan chain and one wrapper output cell in layer number 3 to W_1 (as in Fig.3(b)). In this fashion we have assigned other wrapper elements into the wrapper chains W_2,W_3,W_4 and after the completion of the second cycle, total number of TSVs required are 12.

In the third cycle we have assigned one output cell in W_1. The next wrapper element may be assigned to W_1 as the number of TSVs before and after the assignment is same. But if the next chosen wrapper element is an output cell, output cell requirement for wrapper chain W_1 will not satisfy and hence it has to be placed in wrapper chain W_2 where the output cell requirement is satisfied. This is shown by arrow headed line in 3(c). We also assign an input and output cell in W_2 and one input cell in W_3. At that point total number of TSVs required are 14 which is equal to the total number of available TSVs. Therefore rest of the wrapper elements are assigned in the lowest layer. Remaining selected wrapper elements must be assigned to the shortest wrapper chain without checking the I/O cell requirement. In Fig.3(c) the individual length of the wrapper chains W_1,W_2,W_3 and W_4 is 19,12,11 and 12 respectively. Wrapper chain having same length giving shortest wrapper length is selected arbitrarily. Here we chosen the wrapper chain W_3. In these way remaining wrapper elements are placed to different wrapper chains as shown in Fig.3(d).

VI. Experimental Results

The proposed heuristic is coded in **C** language and executed on a Intel Core 2 Duo processor having 1GB RAM. The experiments are performed based on the cores from ITC'02 SOC test benchmarks. We have presented the experimental results for the core number 7 of d281 and core number 4 of p93791. We have considered that the number of layers for the SOC chip for each case of the simulation is 3.

A. Results

Table III shows the simulation results for the core 7 of SOC d281 for the range of TAM width between 2 to 6. In Table IV we have presented the simulation results for core 4 of SOC p93791 for the range of TAM width between 2 to 8 . Column 1 of each table lists the TAM width, Column 2 presents the maximum umber of available TSVs (TSV_{max}). Column 3 gives the total number of TSVs utilized for

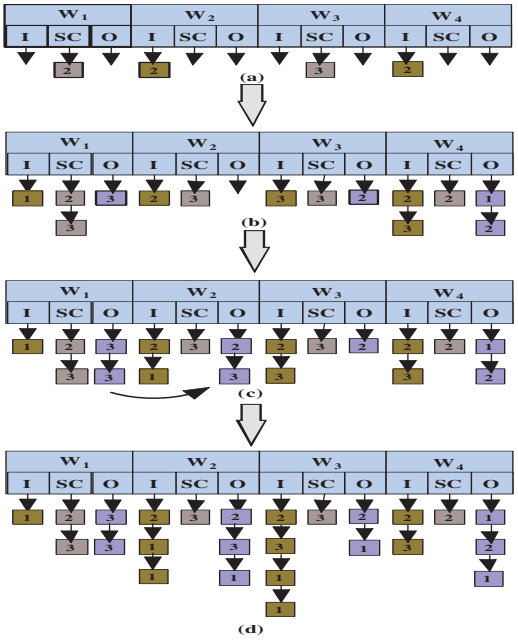

Figure 3. Example of wrapper design

wrapper design using the proposed algorithm. The longest wrapper chain length obtained according to our algorithm is shown in Column number 5. Column number 4 compares with the length of the shortest wrapper length as given in [10]. Column number 6 and 7 compares the CPU running time of the proposed algorithm and the algorithm presented in [10]. The results show that our proposed algorithm utilizes the available number TSVs efficiently and the time required to design the wrapper chain is much less than in [10] for all of the cases. Also for lower number of wrapper chains some TSVs are still unused. Hence, for lower number of wrapper chains where more number of I/O wrapper cells has to be connected we can design that wrapper chain even with less number of TSVs. The results also show that for larger TAM width the wrapper chain is almost balanced. From the Table III and Table IV it is also seen that for some of the cases where the number of available TSVs are less the algorithm in [10] is unable to design the wrapper chain, but our proposed algorithm can design wrapper chain for those cases. Hence, we can conclude that, the proposed algorithm is better than [10] with respect to the following reasons: (a) Complexity is very less, (b) can design the wrapper for even less number of available TSVs and (c) efficient utilization of the TSVs.

VII. Conclusion

In this paper we have presented a polynomial time ($O(N)$) heuristic algorithm to minimize the test time of core-based 3D SOCs. The core wrapper elements are distributed over different wrapper chains and the required number of TSVs are calculated. Our objective was to design the balanced wrapper chains so that the length of the longest wrapper chain is minimum using the available number of TSVs for a particular core. Simulation results are presented on the cores of ITC'02 SOC benchmarks and it shows that the proposed algorithm design the wrapper chain within considerable time which is much less than the algorithm presented in [10] and efficiently utilize the available TSVs.

Algorithm 1: CreateChain

Input : Test set E, Maximum available TSV (TSV_{max}), Number of layers (L), TAM_width (T)
Output: Designed wrapper chain
begin
  Initialize $k = 0$, $TSV\_count = 0$, $tflag = 0$, $pretsv = 0$, $posttsv = 0$, set $V = \phi$, $layer\_number(E_i \in E,$ for all i) $= -1$;
  For each wrapper chain W_k determine wrapper I/O cells requirements R_k, where $k = 1, 2, ..., W$;
  while $E \neq \phi$ **do**
  begin
    if *($pretsv \neq posttsv$)* **then** $k = (k + 1)\%T$;
    if *($tflag == 1$)* **then** find shortest wrapper chain r and set k =r;
    Pick up an element $E_i \in (E - V)$ if available;
    if *($TSV\_count == TSV_{max}$)* **then**
    $tflag = 1$;
    Set $layer\_number(E_i \in (E - V) = 1$;
    if *($length(W_k) == 0$)* **then**
    set $layer\_number(E_i) \in \{2, 3, ..., L\}$;
    else set $layer\_number(E_i) \in \{1, 2, 3, ..., L\}$;
    if *(type_of(E_i)==I/O wrapper cell)* **then**
    begin
      if *($no\_of\_I/O\_wrapper\_cell_k + 1 \leq R_k$)* **then**
      set $r = k$;
      else
      begin
        Find next wrapper chain $W_r(r \neq k)$ that has $No\_of\_I/O\_wrapper\_cell_k + 1 \leq R_k$;
      $No\_of\_I/O\_wrapper\_cell_r+ = 1$;
    if *(type_of(E_i) == scan chain)* **then**
    set $r = k$;
    Add E_i to W_k;
    $No\_of\_elements+ = 1$;
    $pretsv = No\_of\_TSV$;
    $posttsv =$ Required TSV for W_r after insertion of E_i;
    $No\_of\_tsv_r = posttsv$;
    $wrapper\_length(W_r)+ = length(E_i)$;
    $V = V \bigcup E_i$;
    $E = E - E_i$;
    $k = r$;

Acknowledgment

This work is sponsored partly by VLSI Design Project, Govt. of West Bengal and by the UGC, Govt. of India under the grant of F. No. 40-1/2011 (SR), dated 29.07.2011.

SOC Benchmark (d281) Core 7						
TAM Width	Max TSV	Req TSV	Shortest Wrapper Length [10]	Longest Wrapper Length	CPU Time (Min) (Our)	CPU Time (Min) [10]
2	12	10	N/A	1129	0.13	≈ 0.00
	14	11	N/A	1223	0.08	≈ 0.00
	16	10	1623	1223	0.11	0.22
	18	10	1064	1223	0.10	1.42
	22	11	1064	1095	0.10	2.03
3	12	12	N/A	710	0.10	≈0.00
	14	14	N/A	710	0.10	≈0.00
	16	16	1347	710	0.08	0.25
	18	16	752	816	0.08	3.67
	22	15	710	784	0.13	6.73
4	12	12	N/A	532	0.08	≈0.00
	14	14	N/A	532	0.06	0.02
	16	16	1347	532	0.10	0.30
	18	18	611	532	0.10	7.05
	22	22	545	532	0.11	16.17
5	12	12	N/A	426	0.08	0.02
	14	14	N/A	426	0.08	0.05
	16	16	1347	426	0.10	0.47
	18	18	486	426	0.11	11.98
	22	22	427	426	0.11	32.28
6	12	12	N/A	355	0.11	0.02
	14	14	N/A	355	0.10	0.07
	16	16	1347	355	0.10	0.47
	18	18	441	355	0.08	16.75
	22	22	364	355	0.11	53.83

Table III
RESULTS FOR CORE 7 OF D281

SOC Benchmark (p93791) Core 4						
TAM Width	Max TSV	Req TSV	Shortest Wrapper Length [10]	Longest Wrapper Length	CPU Time (Min) (Our)	CPU Time (Min) [10]
2	12	11	N/A	85	0.1	≈0.00
	13	8	132	88	0.11	≈0.00
	14	12	87	91	0.13	≈0.00
	18	11	79	80	0.13	≈0.00
	20	11	77	96	0.11	≈0.00
3	12	12	N/A	53	0.15	≈0.00
	13	13	69	53	0.13	≈ 0.00
	14	14	69	52	0.13	≈ 0.00
	18	15	53	73	0.11	0.03
	20	13	52	64	0.15	0.03
4	12	12	N/A	40	0.1	≈ 0.00
	13	13	59	39	0.1	≈0.00
	14	14	59	40	0.08	0.02
	18	18	43	46	0.1	0.07
	20	20	42	54	0.11	0.08
5	12	12	N/A	31	0.08	≈0.00
	13	13	59	33	0.08	≈0.00
	14	14	59	32	0.1	0.02
	18	18	37	36	0.11	0.12
	20	20	33	31	0.08	0.12
6	12	12	N/A	27	0.08	≈0.00
	13	13	59	27	0.13	≈ 0.00
	14	14	47	28	0.15	0.03
	18	18	31	27	0.11	0.18
	20	20	31	31	0.11	0.32
7	12	12	N/A	24	0.08	≈0.00
	13	13	59	24	0.1	0.02
	14	14	47	24	0.08	0.07
	18	18	31	23	0.1	0.43
	20	20	29	24	0.13	0.87
8	12	12	N/A	21	0.06	≈0.00
	13	13	59	21	0.11	0.08
	14	14	47	21	0.1	0.18
	18	18	27	23	0.08	1.37
	20	20	27	21	0.06	3.32

Table IV
RESULTS FOR CORE 4 OF P93791

REFERENCES

[1] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test Wrapper and Test Access Mechanism Co-Optimization for System-on-Chip", In Journal of Electronic Testing: Theory and Applications(JETTA), vol. 18, pp. 213230, 2002.

[2] Y. Xie, G. H. Loh, and K. Bernstein, "Design space exploration for 3Darchitectures", J. Emerg. Technol. Comput. Syst., vol. 2, no. 2,2006.

[3] "IEEE Std. 1500: IEEE Standard Testability method for Embedded Core-based Integrated Circuits", IEEE Press, 2005.

[4] Y. Xie, G. H. Loh, B. Black, and K. Bernstein, "Design Space Exploration for 3D Architectures", In ACM Journal of Emerging Technology and Computer Systems, vol. 2, no. 2, pp. 65103, 2006.

[5] K. Banerjee, S. J. Souri, P. Kapur, and K. C. Saraswat, "3-D ICS: A Novel Chip Design for Improving deep sub-micrometer Interconnect Performance and System-on-Chip Integration", Proceedings of the IEEE, vol. 89, no. 5, pp. 602 633, May 2001.

[6] S. K. Goel and E. J. Marinissen, "SOC Test Architecture Design for Efficient Utilization of Test Bandwidth", In ACM Trans. Design Automation of Electronic Systems, vol. 8, no. 4, pp. 399429, 2003.

[7] C. Giri, S. Sarkar, and S. Chattopadhyaya, "A Genetic Algorithm Based Heuristic Technique for Power Constrained Test Scheduling in Core-based SOCs", in Proc. IEEE Intl. Conf. on IFIP VLSI-SOC, 2007, pp. 320323.

[8] K. Puttaswamy and G. H. Loh, "Thermal herding: Microarchitecture techniques for controlling hotspots in high performance 3D integrated processors", in IEEE High Performance Computer Architecture, 2007, pp. 193204.

[9] Y. Huang, S. M. Reddy, W.-T. Cheng, P. Reuter, N. Mukherjee,C.-C. Tsai, O. Samman, and Y. Zaidan, "Optimal Core Wrapper width selection and SOC Test Scheduling based on 3-D Bin Packing Algorithm", in Proc. International Test Conference, 2002, pp. 7482.

[10] B. Noia, K. Chakrabarty, and Y. Xie, "Test-Wrapper Optimization for Embedded Cores in TSV-Based Three-Dimensional SOCs", in IEEE International Conference on Computer Design, 2009, pp. 7077.

[11] C. Giri, S. K. Roy, B. Banerjee, and H. Rahaman, "Scan Chain Design Targeting Dual Power and Delay Optimization for 3D Integrated Circuits", in Proc. IEEE Intl. Conf. on Advances in Computing, Control, and Telecommunication Technologies, India,2009, pp. 845849.

[12] X. Wu, Y. Chen, K. Chakrabarty, and Y. Xie, "Test Access Mechanism Optimization for Core-based Three-dimensional SOCs", in IEEE International Conference on Computer Design, 2008, pp. 212218.

[13] L. Jiang, L. Huang, and Q. Xu, "Test Architecture Design and Optimization for Three-dimensional SOCs", in Proc. Design,Automation and Test in Europe, 2009, pp. 220225.

[14] L. Jiang, Q. Xu, K. Chakrabarty, and T. Mak, "Layout driven Test-architecture Design and Optimization for 3D SoCs Under Prebond Test-pin-count Constraint", in IEEE International Conference on Computer Design, 2009, pp. 191196.

[15] B. Noia, K. Chakrabarty, , and E. J. Marinissen, "Optimization Methods for Post-Bond Die-Internal/External Testing in 3D Stacked ICs", in Proc. International Test Conference, 2010.

[16] S. K. Roy, S. Ghosh, H. Rahaman, and C. Giri, "Test Wrapper Design for 3D System-on-Chip Using Optimized Number of TSVs", in IEEE Intl. Symposium on Electronic System Design, Orissa, Bhubaneswar, India, December 2010.