Layout-Driven Test-Architecture Design and Optimization for 3D SoCs under Pre-Bond Test-Pin-Count Constraint

Li Jiang[†*], Qiang Xu[†*], Krishnendu Chakrabarty[‡], and T. M. Mak[§]

[†]Deptartment of CS&E, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
[*]CAS-CUHK Shenzhen Institute of Advanced Integration Technology
[‡]Deptartment of ECE, Duke University, Durham, NC
[§]Intel Corporation, Santa Clara, CA

ABSTRACT

We propose a layout-driven test-architecture design and optimization technique for core-based system-on-chips (SoCs) that are fabricated using three-dimensional (3D) integration. In contrast to prior work, we consider the pre-bond test-pin-count constraint during optimization since these pins occupy large silicon area that cannot be used in functional mode. In addition, the proposed test-architecture design takes the SoC layout into consideration and facilitates the sharing of test wires between pre-bond tests and post-bond test, which significantly reduces the routing cost for a test-access mechanism in 3D technology. Experimental results for the ITC'02 SoC benchmarks circuits demonstrate the effectiveness of the proposed solution.

1. INTRODUCTION

As system-on-a-chip (SoC) designs become increasingly complex, interconnects have emerged as the performance and power limiter for giga-scale integrated circuits (ICs). Three-dimensional (3D) technology is able to provide abundant interconnect resources with improved performance and less communication energy by integrating multiple silicon dies with short and dense through-silicon vias (TSVs). As a result, it has become a promising solution to address the interconnect problem [3]. 3D technology also facilitates the integration of disparate technologies such as microelectromechanical systems (MEMS) and various kinds of sensors as they can be fabricated on different silicon layers separately before integration, thereby offering a genuine single-chip system solution. Because of the above benefits, industry experts predict that 3D ICs will occupy a big market share for future semiconductor products [21], despite the still unresolved testing and thermal-management challenges.

In 3D ICs, silicon dies at different layers can be built in three ways: wafer-to-wafer (W2W) bonding [19], die-to-die (D2D) bonding [13], or die-to-wafer (D2W) bonding (for 3D ICs built on two semiconductor wafers only) [4]. Known good dies (KGDs) can be attached through pre-bond wafer-level testing to achieve higher manufacturing yield in D2D bonding or D2W bonding [3, 15]. Therefore, when the die size is large and/or the defect density is high, they are preferred over W2W bonding. In terms of bonding direction, there can be face-to-face bonding or face-to-back bonding. The former allows more interconnects between active devices on different layers but it limits the number of stacked dies to be two; the latter is a scalable solution that supports more stacking layers.

In order to enable pre-bond tests and improve manufacturing yield for 3D ICs, we need to fabricate a number of test pads on the silicon die so that the automatic test equipment (ATE) can probe it during testing. The test pads, however, occupy much larger area compared to the TSVs. Typically, one single test pad can consume area equivalent to hundreds of front-side vias. Therefore, if a large number of test pads are fabricated, the benefits of exploiting TSVs for interconnecting active devices between layers are significantly reduced. As a result, it is essential to take the pre-bond test-pin-count constraint into consideration during test planning. Related prior work in test architecture design and optimization for 3D SoCs [8], however, tries to integrate pre-bond tests and post-bond test together and may lead to high test pad requirement for certain dies.

To tackle the above problem, in this work, we design test architectures for pre-bond tests and post-bond test separately so that the test-pin-count constraint can be satisfied in pre-bond tests. By doing so, however, the routing cost for test access mechanisms (TAMs) may be dramatically increased as pre-bond tests and post-bond test have different TAMs. To address this issue, we propose optimization methods that allow us to share routing resources between pre-bond tests and post-bond test as much as possible. Also, we show how to optimize test architectures to further reduce TAM routing cost with little impact on testing time. Experimental results on several 3D adaptations of the ITC'02 benchmark circuits show that, the proposed layout-driven test architecture design and optimization techniques for 3D SoCs can reduce TAM wire length by up to 50% with only a small increase in testing time.

The remainder of this paper is organized as follows. Section 2 reviews related work and motivates this paper. The problem investigated in this paper is then formulated in Section 3. Next, our proposed layout-driven test architecture design and optimization techniques are detailed in Section 4. Section 5 presents experimental results to show the advantages of the proposed method. Finally, Section 6 concludes this paper.

2. PRELIMINARIES AND MOTIVATION

2.1 Prior Work in Testing 3D ICs

Test techniques and design-for-testability (DfT) solutions for 3D ICs are critical issues for the success of 3D technology, as pointed out in [14, 16]. However, only limited work has been done in this emerging area. Lewis and Lee [12] proposed a scan-island-based design to enable pre-bond tests for incomplete circuits at the architecture level. Wu *et al.* [18] studied several scan chain design approaches for 3D ICs and compared their routing costs. The above works mainly target 3D ICs that put functional blocks in different silicon layers.

For 3D SoCs with entire embedded cores on different layers, modular testing is an attractive solution as it facilitates the reuse of test patterns. While test architecture design and optimization for two-dimensional SoCs have been subject to extensive research [20], these solutions are not readily applicable for testing 3D SoCs. Recently, a test-access mechanism (TAM) optimization technique was proposed
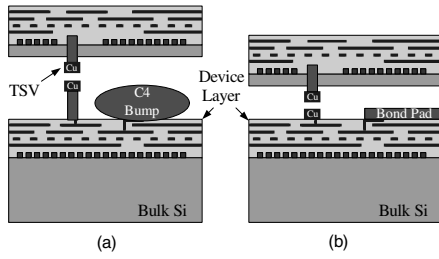
Figure 1: Pre-bond Test Pad: (a) C4 Bump as Test Pad, (b) Wire-Bond as Test Pad.
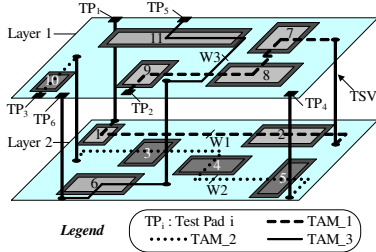

Figure 2: Test architecture for an example 3D SoC.

in [17] to minimize the testing time of 3D SOCs, under limits on the number of TSVs utilized by TAMs. However, pre-bond tests were not considered in this work and hence it can only provide cost-effective solutions for 3D SoCs manufactured with W2W bonding technology. Jiang *et al.* [8] proposed simulated annealing (SA) based algorithms to optimize modular SoC test architecture considering both pre-bond tests and post-bond test. In this work, the same TAMs that traverse multiple layers in post-bond testing are fully reused for pre-bond tests. Consequently, TAMs can be divided into multiple parts and distributed among the different silicon layers. As all the TAM segments in a particular silicon layer need to be probed during pre-bond testing, a large amount of test pads may be required for those silicon dies that contain many TAM segments. This can be a serious issue in pre-bond testing, as shown in the following section.

2.2 Test-Pin-Count Constraint

When conducting pre-bond tests for silicon dies at wafer-level, one of the biggest challenges is how to probe the silicon die effectively. As shown in [1], since fine-grained touchdown probe needles are not available in the next decade, producing dense probe arrays to connect to the ATE is not a viable solution, at least for the near future. Consequently, we have to fabricate test pads (C4 bump or wire bond, see Fig. 1) on silicon dies and rely on conventional probing techniques to connect them to the ATE during pre-bond testing [14].

At the same time, however, it is not possible to fabricate a large number of test pads for pre-bond testing in 3D ICs. This is because of the following reasons. According to [1], the pitch for C4 bumps is around $120\mu m$, which is much larger than that of TSV ($1.7\mu m$ as shown in [22] and this figure keeps shrinking with technology improvements). In other words, one single test pad can consume area equivalent to hundreds of TSVs (see Fig. 1). As these test pads have to be put at the "keep-out area" for TSVs (i.e., TSVs need to keep some distance from any other component), the benefits of exploiting dense TSVs for interconnecting active devices between layers are significantly diminished with the increase of test pads [11]. In 3D technology, except for the bottom layer, the silicon bulks in other layers are thinned for the ease of TSV fabrication. If we conduct pre-bond tests before thinning, we may not be able to detect the failures introduced during the chemical mechanical polishing (CMP) process. If, however, we probe the thinned wafer instead, the probe force (typically $3 - 10g$ per probe and $60 - 120kg$ per wafer) during testing becomes a serious concern as these thinned wafers are not mechanically strong enough. Again, it is desired to have less test pads (probes/touchdowns) for silicon dies in pre-bond testing.
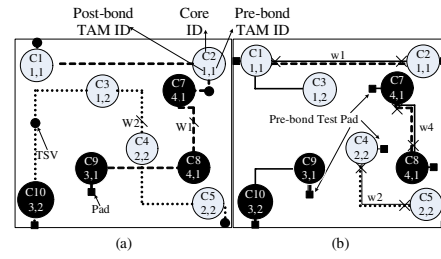
Figure 3: Routing resource sharing example: (a) Test Architecture During Post-bond Test, (b) Reuse TAM During Pre-bond Test.

2.3 Motivation

The most straightforward solution to take pre-bond test-pin-count constraint into consideration during the 3D SoC test architecture design and optimization process is to design *separate* test architectures for pre-bond tests and post-bond test. By doing so, however, the total TAM routing cost for 3D SoCs can be quite high as we have dedicated TAMs for pre-bond tests, resulting in degradation of the chip's routability. As we need to link cores using both pre-bond TAMs and post-bond TAMs and they are used at different times, a natural question is whether we can share some of the routing resources between the two types of TAMs.

We use the following example to demonstrate the possibility of sharing routing resources and its potential benefits. Consider a two-layer 3D SoC containing 11 cores, in which six of them (C1 to C6) are on the bottom layer while the other five cores (C7 to C11) are on the top layer. Similar to [8], for the sake of TSV count consideration, we assume a post-bond TAM involved in several layers will route through all cores tested with this TAM on one layer before it goes through TSVs to connect cores in other layers. In this example 3D SoC, three TAMs are used for post-bond testing and they are shown in Fig. 2. As an example, TAM_1 connects C1, C2, C7, C8, and C9 with TAM width $W1$, starting from test pad TP_1 and ending at test pad TP_2.

For the ease of discussion, we map a few cores in the 3D SoC onto one layer as shown in Fig. 3(a). In this figure, each vertex represents a core, in which the upper label is the core ID, while the lower one denotes the pre-bond TAM ID and post-bond TAM ID that this core belongs to. In Fig. 3(b), we show how pre-bond TAMs can reuse the existing test wires for post-bond testing, wherein the solid lines are pre-bond TAMs. It can be easily observed that those wires having both solid and dashed/doted lines can be shared between pre-bond test and post-bond test, which can significantly reduce the total routing cost for TAMs in 3D SoCs. Note that, during pre-bond test, the end points of each TAM are directly routed to deliver test data on its own silicon layer. Here, we assume that these test pad is near the end point, so that we can ignore the distance between end points and test pads.

Obviously, some design-for-testability (DfT) circuitries need to be introduced to enable the routing resource sharing between pre-bond test and post-bond. To be specific, we need: (i) certain multiplexers to select the different test data source for pre-bond test and post-bond test (see the "×" point shown in Fig. 3(b)); (ii) reconfigurable test wrappers for cores that have different TAM width between pre-bond test and post-bond test (e.g., [9, 10]); (iii) the necessary control mechanisms (typically by introducing extra instructions in test wrapper and JTAG controller).

3. PROBLEM FORMULATION

The layout-driven 3D SoC test architecture design and optimization problem investigated in this paper can be formulated as follows:

Problem: Given

- the set of cores C on the 3D SoC, and the test parameters for each core $c \in C$;

- the layout of the 3D SoC, i.e., the physical position of every core c, including which layer it sits on and its X-Y coordinate on that layer;
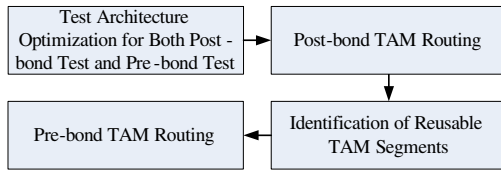
Figure 4: Design Flow For Scheme 1.

- the maximum available TAM width for post-bond test W_{post};

- the pre-bond test-pin-count constraint W_{pre};

Our objective is to determine the number of pre-bond TAMs N_{pre}, the number of post-bond TAM N_{post}, the core assignment associated with each TAM t_i, and the width of each TAMs W_{t_i}, so that the total testing cost, $C_{total} = C_{time} \times \alpha + C_{route} \times (1 - \alpha)$, is minimized ($\alpha$ is a weighting factor designated by users). Note that C_{time} and C_{route} represent the total testing time of the 3D SoC and the total TAM wire length for the 3D SoC, respectively.

For the case that the pre-bond TAMs and post-bond TAMs are not shared, the routing cost C_{route} is simply the total wire length of the two kind of TAMs, that is,

$$C_{route} = \sum_{i=0}^{i<N_{pre}} W_{t_i} \times L_{t_i} + \sum_{i=0}^{i<N_{post}} W_{t_i} \times L_{t_i} \qquad (1)$$

Here, L_{t_i} denotes the wire length for TAM t_i, and we calculate it using the sum of Manhattan distance between adjacent cores in this TAM.

When considering the sharing of pre-bond TAMs and post-bond TAMs, suppose the total length for the shared wires is C_{reused}. The routing cost C_{route} becomes

$$C_{route} = \sum_{i=0}^{i<N_{pre}} W_{t_i} \times L_{t_i} + \sum_{i=0}^{i<N_{post}} (W_{t_i} \times L_{t_i}) - C_{reused} \qquad (2)$$

It is worth noting that test wrapper design and optimization is a subproblem of the above problem, and we use the algorithms in [7, 9] to optimize the IEEE Std. 1500-compliant test wrapper and the reconfigurable test wrapper, respectively.

4. LAYOUT-DRIVEN TEST ARCHITECTURE DESIGN AND OPTIMIZATION

We address the above problem progressively in this section, denoted as Scheme 1 and Scheme 2, respectively. In Scheme 1, we consider the case that test architectures for both pre-bond tests and post-bond test are fixed, and we propose a greedy heuristic to share test wires between pre-bond TAM and post-bond TAM as much as possible. In Scheme 2, to further reduce routing cost, we consider flexible pre-bond test architecture while keeping the post-bond test architecture and its TAM routing unchanged, and then we optimize the pre-bond test architecture so that the total cost C_{total} is minimized. The reason behind the above strategy is: (i) making both pre-bond test architecture and post-bond test architecture flexible would lead to an extremely large solution space, and hence it is rather difficult to find a good solution within limited computational time; (ii) making pre-bond test architecture (instead of the post-bond test architecture) flexible has the benefit that it only affects the routing in one layer.

4.1 Scheme 1: TAM Wire Reuse with Fixed Test Architectures

The design flow for this scheme is shown in Fig. 4. Firstly, we optimize the test architecture for both pre-bond tests and post-bond test, which gives us the number of TAMs, the width for each TAM, and the cores tested on each TAM for each kind of test. Then, we take the 3D SoC layout into consideration and conduct post-bond TAM routing. Next, we identify reusable TAM segments out of the post-bond TAM and conduct pre-bond TAM routing to share them as much as possible. We elaborate the above procedures in detail in the following.

4.1.1 Post-bond TAM Routing

Given a post-bond TAM with several cores tested on it, we first map these cores belonging to different layers onto one virtual layer (as shown in Section 2.3). For the connections that link two cores on different layers, we can ignore the routing cost for the TSVs due to its short length. Given the test architecture, we are to route all the cores belonging to the same TAM sequentially, and hence the routing problem is equivalent to the NP-Hard Traveling Salesman (TSP) problem [5].

To tackle this problem, we first construct a complete graph for all the cores (vertices) belonging to the TAM and the weight of each edge represent its routing cost (i.e., the distance between the two cores multiply the TAM width) between the linked two cores (denoted as $\mathcal{SG}$, see Fig. 5(a)). We have another acyclic graph used to store the final routing result (initialized with no edges), denoted as $\mathcal{EG}$. We consider all the edges in $\mathcal{SG}$ as candidate TAM segments and gradually build $\mathcal{EG}$, using the algorithm shown in Fig. 6.

The input to the post-bond TAM routing algorithm is a set of cores which belong to a TAM, and the output is a core sequence indicating the routing order for the cores on this TAM and its routing cost. In lines 1-4, we construct the completed graph $\mathcal{SG}$ and assign the weight $(L_{ti} \times W_{ti})$ on them. In line 5, we sort all the edges according to their weights. Then, in every iteration (lines 6-10), we move edges from $\mathcal{SG}$ to $\mathcal{EG}$ in a greedy manner (i.e., we move the edge with the smallest weight), e.g., the solid line A-B with length 1 in Fig. 5(b) and the solid line B-C with length 3 in Fig. 5(c)). As the procedure going, more edges (i.e., TAM segments) are moved into $\mathcal{EG}$ and they are gradually linked together as a path (e.g., the linked path A-B-C in Fig. 5(c)). It is important to note that, there are two kinds of redundant edges that should be deleted in every iteration after a new edge is moved into $\mathcal{EG}$ (line 10): (i) Any edge should be deleted if either of its vertex is an *internal vertex* (all vertices except two end points of a path) in current $\mathcal{EG}$; (ii) Any edge that would generate a cycle in $\mathcal{EG}$ should be deleted since $\mathcal{EG}$ should be acyclic all the time. Taking Fig. 5(c) as an example, edge B-E and B-D are the first kind of redundant edges, since vertex B is a internal vertex. Edge A-C is the second kind of redundant edge, since A-B-C-A will become a cyclic graph if we move A-C into $\mathcal{EG}$. Finally, all the paths are linked together to form one single path (e.g., solid lines A-B-C-D-E in Fig. 5(d)), which gives us the final TAM routing order and its cost, as returned from lines 11-12.

4.1.2 Reuse Strategy for TAM Routing Resources

Given the TAM routing for post-bond test, our problem now is to route the pre-bond TAM in such a manner that we can reuse the post-bond TAM test wires as much as possible. To reduce the problem complexity, we first divide every TAM into a set of TAM segments, each linking two adjacent cores on the same silicon layer belonging to the TAM. After routing all the post-bond TAMs, every post-bond TAM segment is considered to be reusable by any pre-bond TAM segment that is on the same silicon layer. For the sake of simplicity, we consider each TAM segment of pre-bond test can reuse test wires from only one TAM segment of post-bond test, and each TAM segment of post-bond test can be reused by only one TAM segment of pre-bond test. It should be noted that, we have excluded those TAM segments that link two cores on different layers.

We examine several scenarios to illustrate how we can reuse post-bond TAM routing resources for pre-bond tests (see Fig. 7). With
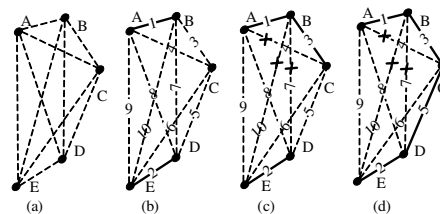
Figure 5: An Example of Post-Bond TAM Routing.

	$C = \{core_1, ..., core_n\} \rightarrow C' = \{core'_1, ..., core'_n\}$
1	Construct a complete graph $\mathcal{SG} = (V, E)$ from all cores in C;
2	**for all** edges $e_{ij} \in E$ {
3	$W(e_{ij}) := w(r) \cdot d(m_i, m_j)$; }
4	$Sum = 0$;
5	**SORT** all the edges $e_{ij} \in E \rightarrow E'$;
6	**while** $E' \neq Empty$ {
7	Pop first edge e_{kl} from E';
8	Add e_{kl} into result graph $\mathcal{EG}$;
9	Sum += $W(e_{kl})$;
10	delete redundant edges in $\mathcal{SG}$;
11	Obtain C' from $\mathcal{EG}$;
12	Obtain Sum;

Figure 6: Post-Bond TAM Routing Algorithm

	$\{C_1, ..., C_n\} \in Layer_i \rightarrow \{C'_1, ..., C'_n\} \in Layer_i$
1	Get the set of reusable post-bond TAM segment F;
2	**for all** TAM_i in this layer {
3	Construct a complete graph $G_i = (V_i, E_i)$ from all cores in C_i belong to TAM_i;
4	put all G_i together into $\mathcal{SG}$; }
5	**for all** edges $e_{ij} \in \mathcal{SG}$ {
6	$W(e_{ij}) := w(r) \cdot d(m_i, m_j)$;
7	add $W(e_{ij}, \varnothing)$ into list $WList(e_{ij})$;
8	**for all** edges $f_{kl} \in F$ {
9	calculate the routing cost after reusing $W(e_{ij}, f_{kl}) = minWidth(e_{ij}, f_{kl}) \times L_{reuse}(e_{ij}, f_{kl})$;
10	record the routing cost $W(e_{ij}, f_{kl})$ and corresponding post-bond TAM segment f_{kl} into list $WList(e_{ij})$; }
11	SORT all the results in $WList(e_{ij})$ in ascending order; }
12	$Sum = 0$;
13	**while** $E \neq Empty$ {
14	find edge $e_{ij} \in E$ with the minimum routing cost $W(e_{ij}, f_x) \in WList(e_{ij})$;
15	delete e_{ij} from E and add it into edge list: TAM_i;
16	Sum += $W(e_{ij})$;
17	**for all** $e_{kl} \in E$ {
18	if exist f_x, remove $W(e_{kl}, f_x)$ from $WList(e_{kl})$; }
19	delete redundant edges from $\mathcal{SG}$;
20	Obtain $\{C'_1, ..., C'_n\}$ from $\{TAM_1, ..., TAM_n\}$;
21	Obtain Sum;

Figure 8: Greedy Heuristic for Pre-Bond TAM Routing

given core layout position (modeled as the center point of the cores), we can draw a bounding rectangle for each TAM segment as shown in Fig. 7(a). To connect these two cores, we can have any routes within this bounding rectangle as long as there is no detour (e.g., route A, B or C), and the Manhattan distance of these routes (i.e., the half perimeter of the bounding rectangle) are all the same.

Let us consider the 3D SoC example demonstrated in Fig. 2. TAM segments C1-C2 and C3-C4 are from post-bond TAM 1 and TAM 3 (see Fig. 7(b)-(d)). Fig. 7(b) depicts the case that a pre-bond TAM segment C1-C3 needs to be routed; while Fig. 7(c) presents another case that we need to route another pre-bond TAM segment C3-C4.

Considering the bounding rectangles for TAM segments discussed above, it is clear that the *coincided rectangle* is where we can route the pre-bond TAM that reuses post-bond TAM routing resources (i.e., the grey parts in Fig. 7(b)-(d)). It is important to note that, however, the reusable wire length is not always the half perimeter of the coincided rectangle (see Fig. 7(d)), and it is calculated as follows.
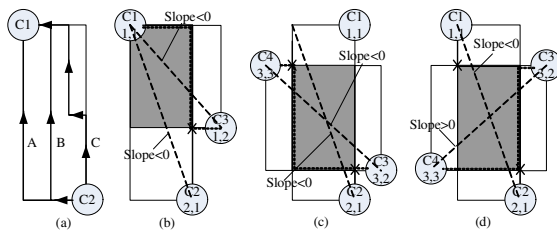
Let us denote the slope of diagonal line with its two end points placed from up-left to bottom-right as negative (e.g., C1-C2 of Fig. 7(b)-(d), C1-C3 of Fig. 7(b)-(c) and C4-C3 of Fig. 7(c)). On the contrary, the slope of diagonal line with its two end points placed from up-right to bottom-left is positive (C3-C4 of Fig. 7(d)). Considering two TAM segments that share test wires, if the slopes of their diagonal lines are the same (i.e., all negative or all positive), as shown in Fig. 7(b)-(c), the reusable wire length is half perimeter of coincided rectangle. If the slopes are different (i.e., one is negative while the other is positive), however, the reusable wire length is the longer edge of the coincided rectangle, as shown in Fig. 7(d).

In view of the above discussion, the problem of reusing post-bond TAM routing resources in a pre-bond TAM cane be stated in terms of how to combine the one-to-one pairs of TAM segments (one from pre-bond test and the other from post-bond test) so that the total routing cost is minimized. We propose a greedy heuristic to solve this problem, as described next.

4.1.3 Greedy Heuristic for Pre-Bond TAM Routing

Fig. 8 presents our proposed greedy heuristic for pre-bond TAM routing, which tries to reuse the routing resources of post-bond TAMs as much as possible.

In line 1, we acquire the possible reusable TAM segments for post-bond test. Similar to the post-bond TAM routing algorithm shown in Fig. 6, we move the edge with the lowest routing cost from $\mathcal{SG}$ to $\mathcal{EG}$ iteratively in a greedy manner. In lines 2-4, we construct a completed graph for every TAM for pre-bond test in the layer, and put

Figure 7: Reusable Routing Resources Represented by Bounding Rectangle.

all these complete graphs together into $\mathcal{SG}$. The reason behind this is that a reusable post-bond TAM segment can be a reusable candidate for TAM segments from more than one pre-bond TAMs. Since each TAM segment of pre-bond test has more than one reusable candidates and each reusable candidates can only be reused at most once, we build a list for each TAM segment of pre-bond test, and store all possible reusable candidates into the list (lines 8-10). To be specific, if one edge cannot reuse any of the TAM segments of post-bond test in that layer, we simply use the original routing cost, calculated by the wire length multiplied by its TAM width, and add it into the list (lines 6-7). If it has a reusable candidate, its routing cost is updated accordingly. In addition, the TAM widths can be different between pre-bond TAM and post-bond TAM. We choose the smaller one to calculate the routing cost of reused wires by multiplying it with the reused wire length. And then we use the original routing cost minus the routing cost of reused wire as the new routing cost of this edge (line 9).

Here, we maintain the list for each edge to keep all the possible reusable TAM segments, and their corresponding updated routing costs (line 10). After sorting the list according to their routing cost (line 11), the head item of each list is the edge with the least routing cost (either with or without reuse strategy). In every iteration, we choose the edge with least routing cost (i.e. value of head item in the list linking to this edge) and move it into $\mathcal{EG}$ (lines 12-15). Since every reusable candidate can only be reused for at most once, we delete this reused segment from all other edges in $\mathcal{SG}$ (lines 17-18). Finally, we obtain the routing result and its cost (lines 19-21).

We take an example extended from Fig. 3 to elaborate the above process. After constructing the complete graph from all cores in pre-bond test, we obtain the list of TAM segments shown in Fig. 9, which keeps all possible reusable post-bond TAM segment in the list of every pre-bond TAM segment. For example, for segment [TAM1(C2,C3)], there are two reusable candidates in its list, that is, post-bond TAM segment (C1,C2) and (C3,C4), The corresponding routing costs are 3 and 10, respectively. The original routing cost is 18. In this example, we pick up the pre-bond TAM segment [TAM1(C1,C2)] first since it has the minimum routing cost 0 by reusing the post-bond TAM segment [0(C1,C2)]. We then move [TAM1(C1,C2)] from $\mathcal{SG}$ to the $\mathcal{EG}$. Afterwards, the reusable post-bond TAM segments [(C1,C2)] are deleted from all the lists. Then, in the second iteration, [TAM2(C4,C5)] is chosen, and the reusable post-bond TAM segment [(C4,C5)] is deleted. Next, we know that [TAM1(C1,C3)] has the minimum routing cost of 8 without any reuse. Again, it is moved to $\mathcal{EG}$. At last, we delete the redundant segment [TAM1(C2,C3)].
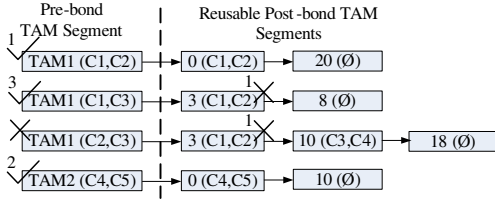
Figure 9: Example for the Proposed Algorithm to Reuse TAM Routing Resources.

4.2 Scheme 2: TAM Wire Reuse with Flexible Pre-bond Test Architecture

By changing the test architecture for pre-bond tests, we can further reduce their routing cost as it is possible to share more routing resources from post-bond TAM. However, it may lead to the increase of testing time, since we change the test architecture which can previously lead to a near optimal solution in terms of testing time cost. As a result, we would like to sacrifice only limited testing time to obtain much better routing cost. In order to achieve the above objective, we extend the simulated annealing-based 3D SoC test architecture optimization procedure presented in [8]. Our design flow for Scheme 2 is shown in Fig. 10.

Similar to [8], the optimization procedure is comprised of two parts: the *outer SA-based core assignment* and the *inner heuristic-based TAM width allocation*. In the SA-based core assignment procedure, cores are randomly moved between multiple TAMs according to certain rules to give a core assignment solution. Note that, the SA procedure guarantees that we are able to reach all possible core assignment solutions with limited number of movements. Then, for each solution, the heuristic-based TAM width allocation procedure is called (see Fig.11), which starts from the initial solution (line 1) and iteratively tries to find the TAM that leads to the lowest total test cost after assigning one-bit wire (line 5-12). In line 7, we use proposed greedy reusable heuristic (as shown in Fig. 8) to calculate its routing cost, and obtain the total test cost including both routing cost and testing time cost (line 8). If this one-bit TAM wire can result in test cost reduction, we will allocate it in this iteration (line 13-15). If not, we increase the width of the to-be-assigned TAM wire by one-bit and try next iteration without any allocation (line 16-17), until a lower cost is found. Eventually, we obtain the pre-bond TAM design with the least test cost (see Fig. 10). It should be noted that the optimization for post-bond test architecture only needs to be done once in the whole procedure in Fig. 10. The same as reusable TAM segment identification is. There is a global variable namely *temperature* related to the acceptance probabilities of each solution. It is set to be a large value when the simulated annealing procedure starts, and gradually decreases. Finally, the simulated annealing procedure terminates when the temperature is lower than a threshold.

5. EXPERIMENTAL RESULTS

5.1 Experimental Setup

To demonstrate the effectiveness of the proposed layout-driven 3D SoC test architecture design and optimization technique, we present experimental results for four revised ITC'02 benchmark SoCs (p22810, p34392, p93791, and t512505). We map these SoCs onto three silicon layers randomly and try to balance the total area of each layer, in
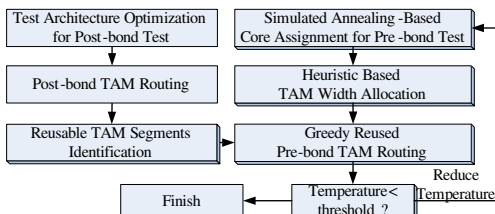

Figure 10: Design Flow for Scheme 2.

1	Allocate one bit width to every TAM
2	Set $b = 1$
3	**While** no more unassigned TAM width
4	$Cost_{min} \leftarrow \infty$
5	**For** each TAM
6	Allocate b bit width to this TAM;
7	Compute routing cost by greedy reusable heuristic;
8	Compute the cost of entire TAM architecture
9	**If** $Cost < Cost_{min}$
10	$Cost_{min} = Cost$
11	Keep this TAM as the only candidate
12	Restore this b bit width
13	**If** $Cost_{min}$ reduces
14	Allocate b bit to the recorded TAM
15	Set $b = 1$
16	**Else**
17	Increase b by one

Figure 11: Heuristic-based TAM Width Allocation.

which the area for a core is estimated based on the number of internal inputs/outputs and scan cells (if any). An academic floorplanner is then utilized to get the coordinates for each core, which are used for wire length calculation.

Test Bus architecture is assumed in our experiments. The pre-bond TAM width is fixed to be 16 by taking the test-pin-count constraint into consideration. We compare three kinds of test architecture design and optimization solutions. The first one (denoted as *No Reuse*), implemented the algorithm in [6] to optimize testing time and it uses the TAM routing algorithm shown in Fig. 6 to route both post-bond TAMs and pre-bond TAMs without sharing routing resources between the two kinds of TAMs. The second one (denoted as *Reuse*), resorts to the same heuristic to optimize testing time as No Reuse, but uses the greedy heuristic algorithm shown in Fig. 8 to route pre-bond TAMs. The last scheme (denoted as *SA*), has the same procedures as *Reuse* to optimize post-bond testing time and post-bond TAM routing, and it uses the SA-based optimization procedure shown in Section 4.2 to adjust the pre-bond test architectures for further test cost reduction. We do not compare against [8], because it does not take test-pin-count constraint into consideration and it may result in pre-bond test architecture with more than 16-bit TAMs, preventing fair comparison.

5.2 Results and Discussion

As can be observed from Tables 1, the testing time of *No Reuse* scheme and *Reuse* scheme (i.e., Scheme 1) are the same as they employ the same test architecture (with different routing strategies only). In most cases, *SA* scheme (i.e., Scheme 2) slightly increase the pre-bond testing time since it sacrifices some testing time to achieve reduced routing cost, but no more than 1% or 2% except for p34392 with large post-bond TAM width. In very few cases, both the testing time and the routing cost for the *SA* scheme are the smallest among the three schemes (e.g., p34392 with TAM width 40).

The routing cost reduction brought by the greedy TAM reusing algorithm used in *Reuse* scheme is considerable when comparing with *No Reuse* scheme. The reduction ratio can be as high as -21.23% for p34392 with TAM width 40. With flexible pre-bond test architecture used in the *SA* scheme, the savings in routing cost is even larger, in the range between -24.87% and -49.39%. The average routing reduction is around 33%, 38%, 46%, and 28% for the four benchmark SoCs. p93791 produces better results because there is no stand-out large core in this SoC, which can serve as a bottleneck during the optimization process. By contrast, t512505 has a large core that alone requires a large post-bond TAM width on its own, which essentially reduces the reusable TAM segments.

Generally speaking, with the growth of post-bond TAM width from 16 to 64, the routing cost reduction ratio increases in the beginning and drops at the end. The main reason is that, when post-bond TAM width grows, the width of reusable TAM segments also goes up; while the TAM width keeps to be 16 for pre-bond tests, the demanded reusable TAM width does not increase, leaving more reusable TAM width idle.

Finally, we present the layout of one layer in 3D SoC p93791 to demonstrate the effectiveness of our TAM reuse strategy (see Fig. 12).

Width	Total Testing Time			Ratio	Routing Cost			Ratio		Total Testing Time			Ratio	Routing Cost			Ratio	
(bit)	No Reuse	Reuse	SA	Δ^T(%)	No Reuse	Reuse	SA	Δ_1^W(%)	Δ_2^W(%)	NoReuse	Reuse	SA	Δ^T(%)	No Reuse	Reuse	SA	Δ_1^W(%)	Δ_2^W(%)
	p22810									p34392								
16	948714	948714	959753	1.16	18677	15922	12648	-14.7	-32.3	2100312	2100312	2088962	-0.54	15154	13595	7760	-10.29	-48.79
24	730866	730866	738800	1.09	19549	17744	14567	-9.23	-25.5	1802341	1802341	1791078	-0.62	27624	22766	17422	-17.59	-36.93
32	647043	647043	653060	0.93	17289	15445	10898	-10.6	-36.9	1592668	1592668	1595286	0.16	25220	20946	13949	-16.95	-44.69
40	608139	608139	611087	0.48	14547	11826	8240	-18.7	-43.4	1579078	1579078	1538795	-2.55	43384	34172	24439	-21.23	-43.67
48	584380	584380	587328	0.50	25532	23402	19181	-8.34	-24.9	1572840	1572840	1567728	-0.33	33544	28460	23705	-15.16	-29.33
56	562561	562561	563882	0.23	17138	15739	10754	-8.16	-37.3	1571728	1571728	1819571	15.77	34068	29669	23423	-12.91	-31.25
64	550549	550549	553497	0.54	17642	16548	11320	-6.20	-35.8	1571728	1571728	1758060	11.86	34068	29669	23145	-12.91	-32.06
	p93791									t512505								
16	3631177	3631177	3726714	2.63	35709	32674	18803	-8.5	-47.34	23494872	23494872	23494872	0	2693	2409	1974	-10.55	-26.70
24	2782827	2782827	2791223	0.30	42034	36818	23727	-12.41	-43.55	23417347	23417347	23494872	0.33	2687	2403	1968	-10.57	-26.76
32	2347166	2347166	2390750	1.86	41403	38383	20953	-7.29	-49.39	18232745	18232745	18310270	0.43	1723	1441	1004	-16.37	-41.73
40	2099123	2099123	2153380	2.58	44550	40509	24709	-9.07	-44.54	18192297	18192297	18438359	1.35	2721	2405	2002	-11.61	-26.42
48	1932476	1932476	1946263	0.71	43546	37730	22749	-13.36	-47.76	18192297	18192297	18352952	0.88	2721	2405	2002	-11.61	-26.42
56	1814195	1814195	1842030	1.53	61754	54135	33205	-12.34	-46.23	18192297	18192297	18269822	0.43	2721	2405	2002	-11.61	-26.42
64	1708376	1708376	1737586	1.71	52686	46992	26974	-10.81	-48.80	18192297	18192297	18482093	1.59	2721	2405	2002	-11.61	-26.42

Δ_1^T: Difference ratio on total testing time between SA and reuse (testing of reuse and N-reuse is the same);

Δ_1^W / Δ_2^W: Difference ratio on wire length between Reuse and No Reuse / SA and No Reuse.

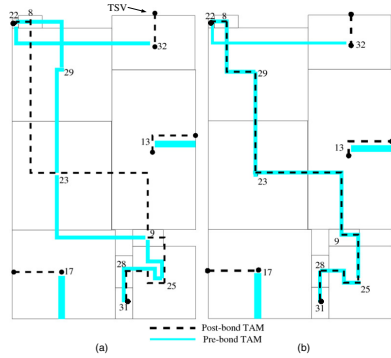**Table 1: Experimental Results for 3D SoC p22810 and p34392.**

Figure 12: Pre-Bond TAM Routing in p93791. (a) without Reusing Post-Bond TAMs; (b) Reusing Post-Bond TAMs.

The TAM segments of post-bond TAM on this layer is shown as dashed lines; while the solid lines are the pre-bond TAMs in this layer. Note that, if a TAM only goes through one single core in this layer, it cannot be reused for pre-bond TAM (e.g., the one for cores 13). Fig. 12(a) depicts the test wires without reusing any post-bond TAM segments. With our reuse methodology as shown in Fig. 12(b), we can see that the the routing overhead for TAMs can be significantly reduced.

6. CONCLUSION

In 3D technology, test pads occupy a much larger area when compared to TSVs and hence we can only fabricate a limited number of test pads for pre-bond testing. In contrast to prior work that does not take such pre-bond test-pin-count constraint into consideration during the 3D SoC test-architecture design and optimization process, we design dedicated pre-bond and post-bond test architectures to satisfy the given test pad constraint. Then, we present novel layout-driven optimization techniques to share the TAM routing resources between pre-bond tests and post-bond test, which can significantly reduce TAM routing cost with little impact on testing time, as shown in our experimental results for ITC'02 SoC benchmark circuits.

7. ACKNOWLEDGEMENTS

8. REFERENCES

[1] International Technology Roadmap for Semiconductors (ITRS). 2009. *Available: www.itrs.net*, 2009.

[2] K. Banerjee, et al. 3-D ICs: A Novel Chip Design for Improving Deep-Submicrometer Interconnect Performance and Systems-on-Chip Integration. In *Proc. of the IEEE*, 89(5):602–633, 2001.

[3] W. R. Davis, et al. Demystifying 3D ICs: The Pros and Cons of Going Vertical. *IEEE Design& Test*, pp. 498–510, 2005.

[4] T. Fukushima, et al. New Three-Dimensional Integration Technology Using Chip-to-Wafer Bonding to Achieve Ultimate Super-Chip Integration. *Japanese Journal of Applied Physics*, 45(4B):3030, 2006.

[5] S. Goel and E. Marinissen. Layout-driven SOC Test Architecture Design for Test Time and Wire Length Minimization. In *Proc. DATE*, pp. 738–743, 2003.

[6] S. K. Goel and E. J. Marinissen. SOC Test Architecture Design for Efficient Utilization of Test Bandwidth. *ACM Transactions on Design Automation of Electronic Systems*, 8(4):399–429, Oct. 2003.

[7] V. Iyengar, K. Chakrabarty, and E. J. Marinissen. Co-Optimization of Test Wrapper and Test Access Architecture for Embedded Cores. *Journal of Electronic Testing: Theory and Applications*, 18(2):213–230, Apr. 2002.

[8] L. Jiang, L. Huang, and Q. Xu. Test Architecture Design and Optimization for Three-Dimensional SoCs. In *Proc. DATE*, pp. 220–225, 2009.

[9] S. Koranne. Design of Reconfigurable Access Wrappers for Embedded Core Based SoC Test. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*,11(5):955–960, 2003.

[10] E. Larsson and Z. Peng. A Reconfigurable Power-Conscious Core Wrapper and its Application to SOC Test Scheduling. In *Proc. ITC*, pp. 1135–1144, 2003.

[11] H.-H. S. Lee and K. Chakrabarty. Test Challenges for 3D Integrated Circuits. In *Special Issue on 3D IC Design and Test. DATE*, 2009.

[12] D. L. Lewis and H.-H. S. Lee. A Scan-Island Based Design Enabling Pre-Bond Testability in Die-Stacked Microprocessors. In *Proc. ITC*, pp. 1–8, 2007.

[13] G. H. Loh, Y. Xie, and B. Black. Processor Design in 3D Die-Stacking Technologies. *IEEE Micro*, 27(3):31–48, 2007.

[14] T. M. Mak., et al. Testing of 3D Circuites. In *Handbook of 3D Integration:Technology and Applications using 3D Integrated Circuits.*, Wiley-CVH, 2008.

[15] G. Smith, et al. Yield Considerations in the Choice of 3D Technology. In *Proc.ISSM*, pp. 1–3, 2007.

[16] T. Vucurevich. The Long Road to 3-D Integration: Are We There Yet? *Keynote speech at the 3D Architecture Conference*, 2007.

[17] X. Wu, et al. Test-Access Mechanism Optimization for Core-Based Three-Dimensional SOCs. In *Proc. ICCD*, 2008.

[18] X. Wu, P. Falkenstern, and Y. Xie. Scan Chain Design for Three-Dimensional Integrated Circuits (3D ICs). In *Proc. ICCD*, pp. 208–214, 2007.

[19] Y. Xie, et al. Design Space Exploration for 3D Architectures. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 2(2):65–103, 2006.

[20] Q. Xu and N. Nicolici. Resource-Constrained System-on-a-Chip Test: A Survey. *IEE Proc., Computers and Digital Techniques*, 152(1):67–81, January 2005.

[21] Yole Development. Market Trends for 3D Stacking. www.yole.fr.

[22] L. Zhou, C. Wakayama, and C. Shi. CASCADE: A Standard Supercell Design Methodology With Congestion-Driven Placement for Three-Dimensional Interconnect-Heavy Very Large-Scale Integrated Circuits. *IEEE Transaction on Computer Aided Design of Integrated Circuits*, 26(7):1270–1282, 2007.