Integrated Test-Architecture Optimization and Thermal-Aware Test Scheduling for 3-D SoCs Under Pre-Bond Test-Pin-Count Constraint

Li Jiang, *Student Member, IEEE*, Qiang Xu, *Member, IEEE*, Krishnendu Chakrabarty, *Fellow, IEEE*, and T. M. Mak, *Senior Member, IEEE*

***Abstract*—We propose a layout-driven test-architecture design and optimization technique for core-based system-on-chips (SoCs) that are fabricated with three-dimensional (3-D) integration technology. 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 test-access mechanisms. In addition, a thermal-aware test scheduling algorithm is proposed to eliminate hot spots during manufacturing test. Experimental results for the ITC'02 SoC benchmarks circuits demonstrate the effectiveness of the proposed solution.**

***Index Terms*— Pre-bond test, test architecture design and optimization, thermal-aware test scheduling, three-dimensional (3-D) integrated circuits (ICs), through-silicon via (TSV).**

I. 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 (3-D) 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 [1]. 3-D 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 3-D ICs will occupy a big market share for future semiconductor products [2], despite the still unresolved testing and thermal-management challenges.

In 3-D ICs, silicon dies at different layers can be built in three ways: wafer-to-wafer (W2W) bonding [3], die-to-die (D2D) bonding [4], or die-to-wafer (D2W) bonding (for 3-D ICs built on two semiconductor wafers only) [5]. Known good dies (KGDs) can be attached through pre-bond wafer-level testing to achieve higher manufacturing yield in D2D bonding or D2W bonding [1], [6]. 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 3-D 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 [8]. The test pads, however, occupy much larger area than TSVs. Typically, one single test pad can consume area equivalent to hundreds of front-side vias. 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 3-D SoCs [9], however, tries to integrate pre-bond tests and post-bond test together and may lead to high test pad requirement for certain dies. In addition, stacking silicon dies in the vertical direction makes heat dissipation a serious concern for 3-D ICs [10]. Overheating during manufacturing test may lead to permanent chip damage, thereby decreasing 3-D IC yield. Thus, it is essential to take thermal issues into consideration during 3-D SoC test architecture design and optimization (particularly during post-bond test), in order to avoid the yield loss problem.

Manuscript received August 04, 2010; revised February 21, 2011 and April 28, 2011; accepted May 12, 2011. Date of publication July 29, 2011; date of current version July 05, 2012. This work was supported in part by the General Research Fund under Grant CUHK417807 and Grant CUHK418708 from Hong Kong SAR Research Grants Council (RGC), the National Science Foundation of China (NSFC) under Grant 60876029, and the NSFC/RGC Joint Research Scheme under Grant N_CUHK417/08. The work of K. Chakrabarty was supported in part by the National Science Foundation under Grant 1017391 and the Semiconductor Research Corporation under Contract 2118. A preliminary version of this paper appeared in the Proc. IEEE/ACM International Conference on Computer-Aided Design, 2009.

L. Jiang is with the CUhk REliable Computing Laboratory (CURE), Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong (e-mail: ljiang@cse.cuhk.edu.hk).

Q. Xu is with the CUhk REliable Computing Laboratory (CURE), Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong, and also with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences (e-mail: qxu@cse.cuhk.edu.hk).

K. Chakrabarty is with Department of Electrical and Computer Engineering, Duke University, Durham, NC, USA (e-mail: krish@ee.duke.edu).

T. M. Mak is with Intel Corporation, Santa Clara, CA 95051 USA (e-mail: t.m.mak@intel.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2011.2160410

To tackle the pre-bond test-pin-count constraint problems, 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. To tackle the heat dissipation problem, a novel thermal-aware test scheduling algorithm is proposed to reduce the hot-spot temperature of the SoC during post-bond test. Experimental results on several 3-D adaptations of the ITC'02 benchmark circuits demonstrate the effectiveness of the proposed solution.

The remainder of this paper is organized as follows. Section II reviews related work and motivates this paper. The problem investigated in this paper is then formulated in Section III. Next, our proposed layout-driven test architecture design and optimization techniques and the thermal-aware test scheduling algorithms are detailed in Section IV and Section V, respectively. Section VI presents our experimental results for benchmark circuits. Finally, Section VII concludes this paper.

II. Preliminaries and Motivation

A. Prior Work in SoC Testing

When testing SoC devices, embedded cores are typically tested as separate, standalone units, wherein test wrappers are constructed to isolate them from the environment during test while dedicated test access mechanisms (TAMs) facilitate the transportation of test stimuli/responses between the core-under-test (CUT) and test source/sink (e.g., tester).

When the system architecture and the floorplan of the SoC are determined, i.e., when we are given known core tests (from core provider or developed in-house) and the position of each core, the system integrator needs to design a test architecture and a test schedule to minimize the test cost, which must account for the test application time and design-for-testability (DfT) overhead. Then, after the test architecture is developed and inserted into the circuit, the system integrator continues with physical design and the remaining design tasks in the IC design flow. The optimization of the above modular test architectures and test scheduling has been subject to extensive research [11].

Various constraints need to be considered during the test scheduling process, which determines the start and end test times for all of the cores. As overheating can damage the CUT, it is important to take thermal issues into consideration during test scheduling. Power-constrained test scheduling techniques (e.g., [12]–[15]) are beneficial, but, due to the nonuniform spatial power distribution across the chip, limiting the maximum chip-level power consumption only does not avoid local hot spots [16].

To address the above problem, Rosinger *et al.* [17], [18] proposed using a simple thermal model to guide test scheduling. This thermal model captures the heat transfer paths originating at the cores under test in a given test session and considers the lateral flow of heat to its neighboring cores. In [16], Liu *et al.* considered "hot spots" (any cores with excessive temperature) during test and proposed two algorithms that attempt to spread heat more evenly over the chip. In [19], He *et al.* proposed to conduct test partitioning and interleaving to allow hot cores to cool off while the freed test resources are used to test other cores.

B. Prior Work in Testing 3-D ICs

Test techniques and DfT solutions for 3-D ICs are critical issues for the success of 3-D technology, as pointed out in [8] and [20]. However, only limited work has been done in this emerging area.

Lewis and Lee [21] proposed a scan-island-based design to enable pre-bond tests for incomplete circuits at the architecture level. Wu *et al.* [22] studied several scan chain design approaches for 3-D ICs and compared their routing costs. The above works mainly target 3-D ICs that put functional blocks in different silicon layers.

For 3-D 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 2-D SoCs have been subject to extensive research [11], these solutions are not readily applicable for testing 3-D SoCs due to the newly introduced constraints and optimization objective. Marinissen *et al.* [23] proposed a novel test wrapper design for cores in 3-D SoCs, but it did not address the test architecture optimization problem. Recently, a test architecture optimization technique was proposed in [24] to minimize the testing time of 3-D 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 3-D SoCs manufactured with W2W bonding technology. Noia *et al.* [25] considered the case with given fixed or yet-to-be-designed test architecture on each die, which is able to optimize for post-bond test only. Jiang *et al.* [9] also proposed simulated annealing (SA)-based algorithms to optimize modular SoC test architecture considering both pre-bond and post-bond tests. 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 of 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 [26], as shown in the following section.

C. Test-Pin-Count Constraint in 3-D IC Pre-Bond Testing

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 [27], since fine-grained touchdown probe needles are not available in the next decade, producing

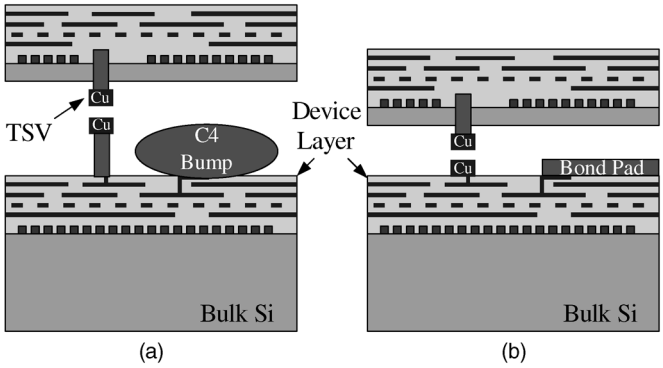


Fig. 1. Pre-bond test pad. (a) C4 bump as test pad. (b) Wire-bond as test pad.

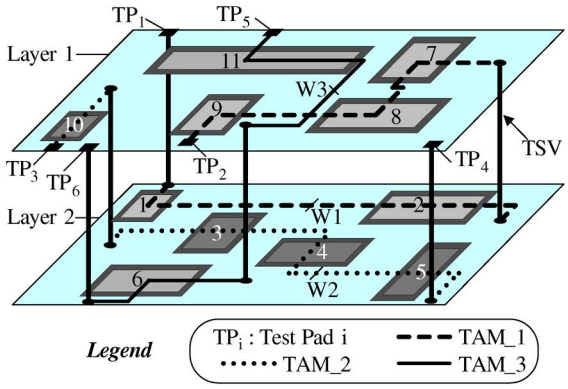


Fig. 2. Test architecture for an example 3-D SoC.

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 [20]. At the same time, however, it is not possible to fabricate a large number of test pads for pre-bond testing in 3-D ICs. This is due to the following reasons. According to [27], the pitch for C4 bumps is around 120 μm, which is much larger than that of TSV (1.7 μm, as shown in [28], and this figure keeps shrinking with technology improvements). In other words, one single test pad can consume an 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 [29]. In 3-D 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 3–10 g per probe and 60–120 kg 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.

D. Motivation

The most straightforward solution to take pre-bond test-pin-count constraint into consideration during the 3-D 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 3-D 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

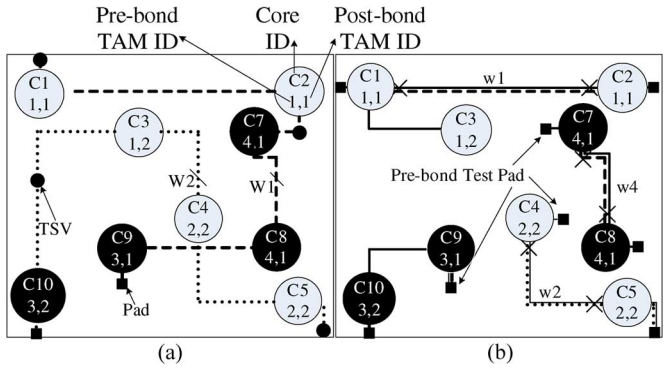


Fig. 3. Routing resource sharing example. (a) Test architecture during post-bond test. (b) Reuse TAM during pre-bond test.

a two-layer 3-D 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 [9], 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 3-D 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 3-D 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 to which this core belongs.

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 3-D 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.

In addition, as overheating is a serious concern for 3-D ICs, it is essential to take thermal issues into consideration during test planning, particularly for post-bond testing. Otherwise, it is likely that certain adjacent hot cores are tested simultaneously, and thus cause permanent damage to the circuit. In this work, after the optimized test architecture is obtained, we use a thermal-aware test scheduling algorithm for post-bond testing to determine the start time and end time of each core test, in order to minimize the temperature of hot spots during testing. Note that we do not apply the proposed thermal-aware scheduling for pre-bond tests because the thermal issue is not so critical in wafer-level pre-bond test.

Clearly, some DfT circuitries need to be introduced to enable the routing resource sharing between pre-bond test and post-bond. To be specific, we need: 1) 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)]; 2) reconfigurable test wrappers for cores that have different TAM width between pre-bond test and post-bond test (e.g., [30] and [31]); and 3) the necessary control mechanisms (typically by introducing extra instructions in test wrapper and JTAG controller).

III. Problem Formulation

The problem investigated in this paper includes two parts: 1) the test architecture design that determines the width of each TAM partition and the core assignment to each TAM during testing and 2) test scheduling that determines the start time and end time for each core under test. While the test architecture design and the test scheduling problems are interrelated, integrating the two problems together constitutes a much larger solution space and may prevent us from finding a good solution in a computationally efficient manner. Since the final testing time and TAM routing cost are mainly determined by the test architecture design while test scheduling itself is sufficiently flexible to avoid testing hot cores concurrently whenever possible, we divide our 3-D SoC test architecture design and optimization problem naturally into two subproblems, as formulated below.

A. Test Architecture Design Under Pre-Bond Test-Pin-Count Constraint

Problem: Given:

- set of cores C on the 3-D SoC, and the test parameters for each core $c \in C$;
- layout of the 3-D SoC, i.e., the physical position of every core c, including which layer it sits on and its X-Y coordinate on that layer;
- maximum available TAM width for post-bond test W_{post};[1]
- 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, $C_{\text{total}} = C_{\text{time}} \times \alpha + C_{\text{route}} \times (1-\alpha)$, is minimized ($\alpha$ is a weighting factor designated by users).

[1]Generally speaking, W_{post} is determined by the system integrator, considering the available test pins in the package and test cost in terms of testing time and DfT overhead.

Note that C_{time} and C_{route} represent the total testing time of the 3-D SoC and the total TAM wire length for the 3-D 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,

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

Here, L_{t_i} denotes the wire length for TAM t_i. Assuming TSVs can be placed within bounding rectangle of two cores, 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

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

Note that the proposed architecture involves some extra DfT cost, e.g., multiplexers used to choose between pre-bond testing and post-bond testing mode and the corresponding test wrapper control instructions. However, such cost is usually not a serious concern when compared with testing time and TAM routing cost and, hence, ignored in our cost mode. We have also ignored the routing cost from cores to test pads in this work because the distances between test pads and embedded cores are quite small when compared to TAMs that route through multiple layers.

It is also worth noting that test wrapper design and optimization is a subproblem of the above problem, whose objective is to balance wrapper scan chain lengths according to the TAM width allocated to the core under test [32]. In this work, we use the algorithms in [30] and [33] to optimize the IEEE Std. 1500-compliant test wrapper and the reconfigurable test wrapper, respectively.

B. Thermal-Aware Test Scheduling for Post-Bond Test

It is well known that heat transfer can be described as currents passing through thermal resistors, leading to a temperature difference analogous to voltage drop [34]. Using this model, we define our thermal-aware test scheduling problem as follows.

Problem: Given:

- test architecture in the 3-D SoC, that is, 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};
- layout of the 3-D SoC, i.e., the physical position of every core c, including which layer it sits on and its X-Y coordinate on that layer;
- average power consumption for each core under test;
- thermal resistance between neighboring cores.

Our objective is to determine the start time and end time of each core during post-bond testing, such that the temperature of the hot spot core of the SoC under test is minimized.

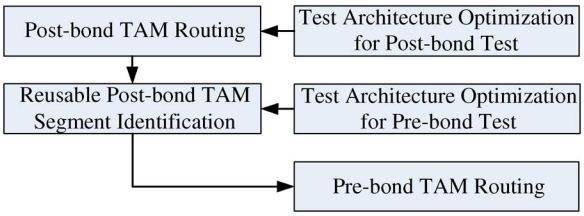


Fig. 4. Design flow for scheme 1.

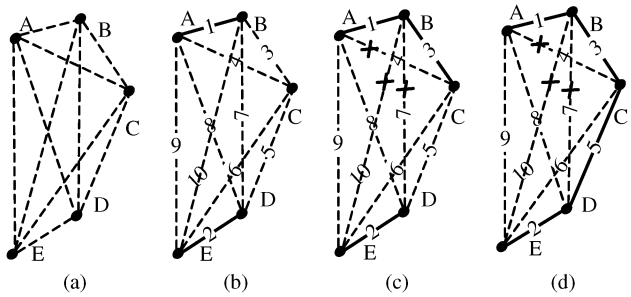


Fig. 5. Example of post-bond TAM routing.

$C = \{core_1, ..., core_n\} \rightarrow C' = \{core'_1, ..., core'_n\}$

```
1   Construct a complete graph SG = (V,E) from all cores in C;
2   for all edges e_ij ∈ E{
3       W(e_ij) := w(r) · d(m_i,m_j);}
4   Sum = 0;
5   SORT all the edges e_ij ∈ E → E';
6   while E' ≠ Empty{
7       Pop first edge e_kl from E';
8       Add e_kl into result graph EG;
9       Sum += W(e_kl);
10      delete redundant edges in SG;
11  Obtain C' from EG;
12  Obtain Sum;
```

Fig. 6. Post-bond TAM routing algorithm.

IV. 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: 1) 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 and 2) making pre-bond test architecture (instead of the post-bond test architecture) flexible has the benefit that it only affects the routing in individual layers.

A. Scheme 1: TAM Wire Reuse With Fixed Test Architectures

The design flow for this scheme is shown in Fig. 4. First, we optimize the test architecture for both pre-bond tests and post-bond test, using the heuristic proposed in [35], 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 3-D SoC layout into consideration and conduct post-bond TAM routing. Next, we identify reusable TAM segments out of the post-bond TAM with the core assignment information from pre-bond TAM design. At last, we conduct pre-bond TAM routing to share them as much as possible.

Before we elaborate the proposed TAM routing technique in detail, for the sake of self-containment, we briefly introduce [35] here. To minimize testing time, this algorithm first creates an initial test architecture by assigning value 1 to each core's TAM width. Since the overall testing time of the system equals the bottleneck TAM with the longest test application time, in the second and third steps, the algorithm iteratively optimizes through merging TAMs and distributing freed TAM resources. Either two nonbottleneck TAMs are merged with less TAM width to release freed TAM resources to the bottleneck TAM, or the bottleneck TAMs is merged with another TAM to decrease testing time. In the last step, the algorithm tries to further minimize testing time by placing one of the cores assigned to the bottleneck TAM to another TAM.

1) Post-Band 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 II-C). 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 of the cores belonging to the same TAM sequentially, and hence the routing problem is equivalent to the NP-Hard Traveling Salesman (TSP) problem [35].

To tackle this problem, we first construct a complete graph for all the cores (vertices) belonging to the TAM and the weight (W) of each edge represent its routing cost (i.e., the distance between two cores m_i and m_j $(d(m_i, m_j))$ multiply the TAM width $(w(r))$) 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 of 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 $(w_{ti} \times t_{ti})$ on them. In line 5, we sort all of 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 progresses, 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): 1) 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 $\mathcal{EG}$ and 2) any edge

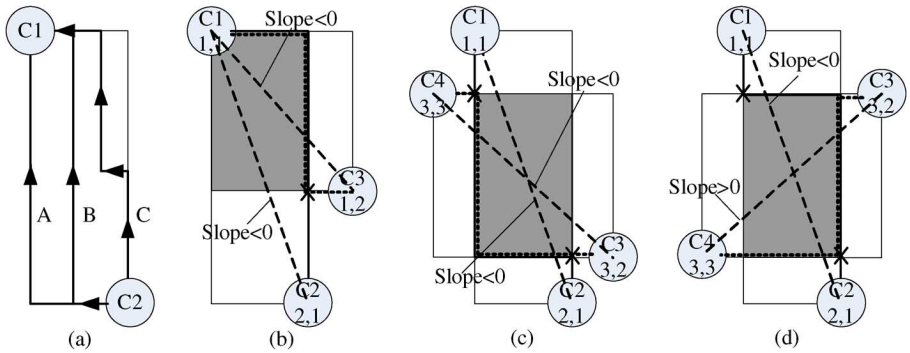


Fig. 7. Reusable routing resources represented by a bounding rectangle.

that would generate a cycle in $\mathcal{EG}$ should be deleted since $\mathcal{EG}$ should be acyclic all of the time. Taking Fig. 5(c) as an example, edges 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.

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 of 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 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 3-D 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 gray parts in Fig. 7(b)–(d)]. It is important to note however, that 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 the 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) and (c), the reusable wire length is half the perimeter of the 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 can 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 will be described next.

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 all of 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

```
{C1,...,Cn} ∈ Layer_i → {C'1,...,C'n} ∈ 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 SG;}
5   for all edges e_ij ∈ SG{
6       W(e_ij) := w(r) · d(m_i, m_j);
7       add W(e_ij, ∅) into list WList(e_ij);
8       for all edges f_kl ∈ F{
9           calculate the routing cost after reusing
            W(e_ij, f_kl) = minWidth(e_ij, f_kl) × 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 ≠ Empty{
14      find edge e_ij ∈ E with the minimum
            routing cost W(e_ij, f_x) ∈ 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 ∈ E{
18          if exist f_x, remove W(e_kl, f_x) from WList(e_kl);}
19      delete redundant edges from SG;
20  Obtain {C'1,...,C'n} from {TAM1,...,TAMn};
21  Obtain Sum;
```

Fig. 8. Greedy heuristic for pre-bond TAM routing.

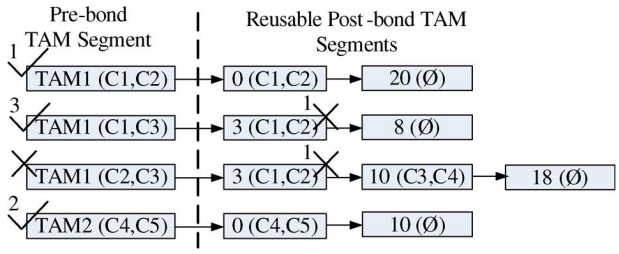


Fig. 9. Example for the proposed algorithm to reuse TAM routing resources.

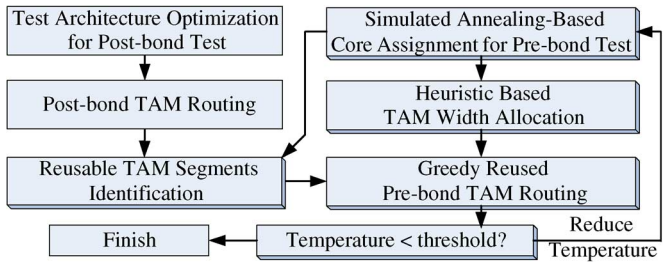


Fig. 10. Design flow for Scheme 2.

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 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)].

B. 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 3-D SoC test architecture optimization procedure presented in [9]. Our design flow for Scheme 2 is shown in Fig. 10.

Similar to [9], 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 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.

```
1   Allocate one bit width to every TAM
2   Set b = 1
3   While no more unassigned TAM width
4   Cost_min ← ∞
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
```

Fig. 11. Heuristic-based TAM width allocation.

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.

V. Thermal-Aware Test Scheduling for Post-Bond Test

After post-bond test architecture optimization, the TAM width allocation and core assignments on each TAM are determined. In this section, we apply thermal-aware test scheduling to reduce the hot-spot temperature. First, as temperature simulation is extremely time-consuming, instead of directly using temperature values, we present an efficient and effective 3-D thermal cost model to guide our test scheduling process. We then use a greedy heuristic to iteratively reduce the thermal cost of our test schedule. Finally, when a small amount of test time can be sacrificed, we carefully insert idle time into our test schedule to cool hot-spot cores. It should be noted that, as long as the temperature is not too high to result in damage or reliability loss of the circuit, it is acceptable to have hot-spots in the circuit from the testing perspective. Therefore, we do not try to even the temperature of various cores during test. Instead, our objective is to reduce the temperature of the hot-spot rather than to eliminate hot-spots in this work.

A. Thermal Cost Function

We integrate an existing 2-D lateral thermal resistive model presented in [36] with 3-D compact resistive network model as in [10] (see Fig. 12). To model thermal dissipation, the dies are split into coarse-grained tiles, wherein thermal resistances are inserted in between neighboring tiles. That is, if two cores are on the same layer, there exists a horizontal thermal resistance only if they have adjacent border (in Fig. 12(a), $\mathrm{Core}_A - \mathrm{Core}_B$, $\mathrm{Core}_A - \mathrm{Core}_C$ and $\mathrm{Core}_C - \mathrm{Core}_B$). If two cores are on neighboring layers, there exists a vertical thermal resistance between them if their footprints overlap with each other. A hot core will dissipate heat through both its lateral neighbors and its vertical neighbors. The temperature could be calculated using

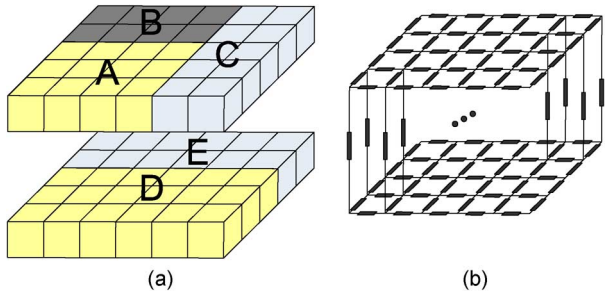


Fig. 12. 3-D lateral thermal-resistive model. (a) Tiles stack array in 3-D SoC. (b) Compact resistive network thermal model.

an Elmore-delay-like closed-form formula. For a tile i, the temperature increase due to power dissipation of tile j could be calculated as

$$T = \sum_{i=1}^{n} \left(R_i \sum_{j=i}^{n} P_j \right). \tag{3}$$

Let $R_{i,j}$ be the thermal resistance from core c_j to c_i $((R_{i,i} = 0))$, let $R_{\mathrm{TOT,j}}$ be the total thermal resistance from core c_j, Pavg_j be the average test power consumption of c_j, and $\mathrm{Trel}_{i,j}$ be the relevant time of the two scheduled cores that are tested concurrently. We use the definition of thermal cost as the thermal contribution from a core to another core [36]. We build the thermal cost induced from testing core c_j to core c_i as follows:

$$\mathrm{Tcst}_j(c_i) = \frac{R_{i,j}}{R_{\mathrm{TOT,j}}} \times \mathrm{Pavg}_j \times \mathrm{Trel}_{i,j}. \tag{4}$$

Using (4), the total thermal contribution of other cores to core c_i for a certain test schedule can be computed as

$$\mathrm{Tcst}_{\mathrm{TOT}}(c_i) = \sum_{j=1}^{N} Tcst_j(c_i). \tag{5}$$

The thermal cost of c_i itself is given in

$$STcst(c_i) = Pavg_i \times TAT_i \tag{6}$$

wherein TAT_i is the test time of c_i.

Combining with (5) and (6), we can get the total thermal cost of a core with a given test schedule as

$$\mathrm{Tcst}(c_i) = S\mathrm{Tcst}(c_i) + \mathrm{Tcst}_{\mathrm{TOT}}(c_i). \tag{7}$$

B. Test Scheduling Algorithm

The basic idea of our thermal-aware test scheduling heuristic is to iteratively change our test schedule with reduced maximum thermal cost as constraint. To achieve this, inspired by [37], we schedule those hot cores as early and as quickly as possible in each TAM to obtain an initial maximum thermal cost during the *initialization* procedure. That is, we first calculate the self thermal cost of each core $(S\mathrm{Tcst}(\mathrm{core}))$ and sort these cores on each TAM in descending order to construct the sorted list. We treat it as the initial test schedule and calculate the thermal cost

```
Input:Sorted Lists:S={TAM_1,...,TAM_n};
      Maximum Thermal Cost, Max(Tcst);
Output:Schedule, S'={TAM'_1,...,TAM'_n};
1   S' = ∅
2   for each TAM TAM'_i, SST(TAM'_i) = 0;
3   while S ≠ ∅
4       choose TAM'_i with Min(SST(TAM'_i)) and TAM_i ≠ ∅
5       for each core c_k ∈ TAM_i
6           move c_k from TAM_i to TAM'_i;
7           ST(c_k) = SST(TAM'_i);
8           if ∃c_j ∈ S' that Tcst(c_j) ≥ Max(Tcst)
9               move c_k back to TAM_i;
10          else update SST(TAM'_i)+=TestTime(c_k), and break;
11      if no core in TAM_i is able to be scheduled
12          update the SST(TAM'_i) with the minimum
            start schedule time of all other TAMs except TAM'_i;
13          choose the first core in TAM_i and goto 5;
14  if the total inserted idle time violate the time budget, goto 16
15  for current S', find the new Max(Tcst)
    backup current schedule S'_pre = S' and goto 1;
16  restore the previous schedule S' = S'_pre and end;
```

Fig. 13. Proposed test scheduling heuristic.

of each core $\mathrm{Tcst(core)}$ and obtain the maximum one, denoted as $\mathrm{Max(Tcst)}$.

Our objective during the test scheduling process is then to minimize the maximum thermal cost whenever possible. As shown in Fig. 13, this procedure has two inputs: the initial test schedule (S) with the sorted core list on each TAM according to their self thermal cost obtained from the first step, and the initial maximum thermal cost $(\mathrm{Max(Tcst)})$. The output of this procedure is a new test schedule S'. During the scheduling process, we gradually move cores from S to S' and try to reduce the maximal thermal cost of the new test schedule. For the ease of explanations, we use the following terms. S is composed of $\mathrm{TAM}_1, \ldots, \mathrm{TAM}_n$ while S' is composed of $\mathrm{TAM}'_1, \ldots, \mathrm{TAM}'_n$. TAM'_i is the corresponding TAM of TAM_i in the output test schedule. We use $\mathrm{SST(TAM}_i)$ to denote the start time of the next core to be scheduled in TAM i. In addition, we use $\mathrm{TestTime}(c_k)$ and $ST(c_k)$ to represent the test time of core c_k and the its start time in the test schedule.

In the beginning of the procedure, we initialize the output schedule S' and the start schedule time of each TAM (lines 1–2). We choose an unfinished TAM TAM'_i (i.e., a TAM with unscheduled cores) with the earliest start schedule time (line 4). Then we try to schedule a core c_k into TAM'_i using the current start schedule time as its start time (lines 6–7). Once this new core is scheduled, we check whether any core in the current schedule S' has a thermal cost that is greater than or equal to the current maximum thermal cost, indicating that the new scheduled core c_k causes a new hot spot (line 8). If such a core exists, we cancel the schedule of c_k and move the scheduled core back to TAM_i (line 9). We next try to schedule the next core with smaller thermal cost in this TAM (line 6). After scheduling a core in TAM_i without violating the maximum thermal cost, we confirm this schedule by updating the start schedule time of TAM'_i (line 10). If, however, none of the cores in TAM_i can be scheduled without generating new hot spot, we insert idle time into TAM'_i by updating the $\mathrm{SST(TAM}'_i)$ to be the earliest start schedule time of all other TAMs in S' except the current TAM (line 11–13). In other words, the next time when we schedule a core onto TAM'_i, there must be at least one less core under concurrent test, thereby reducing thermal cost. The above procedure continues until all the cores are scheduled. Afterwards, we update the maximum thermal cost and use it as the new constraint to guide our scheduling process to generate another test schedule (line 15).

Clearly, inserting idle time into the test schedule can reduce thermal cost, but it also has an adverse impact on testing time. To address this problem, we consider to generate our new test schedule under a testing time extension budge T_{budge}, defined as a percentage of the original SoC testing time designated by the user. Whenever a new test schedule is generated with reduced thermal cost, we calculate the new testing time and our procedure ends when it exceeds our testing time budget, in which case we restore the previous test schedule and use it our final solution (line 16).

VI. Experimental Results

A. Experimental Setup

To demonstrate the effectiveness of the proposed layout-driven 3-D 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 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. We assume that the test power consumption of a core is proportional to the total number of flip-flops. For thermal simulation, we use an academic tool *Hotspot* [34] in grid mode, which can obtain the temperature of hot spot cores in 3-D SoCs.

Test Bus architecture is assumed in our experiments. The pre-bond TAM width is fixed to be 16 bit to take test-pin-count constraint into consideration. We compare three kinds of test architecture design and optimization solutions. The first one (denoted as *No Reuse*) implements the algorithm in [38] 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 IV-B to adjust the pre-bond test architectures for further test cost reduction.

B. Results and Discussion

As can be observed from Table I, the testing times of the *No Reuse* scheme and the *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.,

TABLE I
EXPERIMENTAL RESULTS FOR BENCHMARK 3-D SOCS

Width	Total Testing Time				Ratio	Routing Cost				Ratio		#Test-Pin	
(bit)	[9]	No Reuse	Reuse	SA	$\Delta^T(\%)$	[9]	No Reuse	Reuse	SA	$\Delta_1^W(\%)$	$\Delta_2^W(\%)$	[9]	SA
	p22810												
16	1062281	948714	948714	959753	1.16	12095	18677	15922	12648	-14.7	-32.3	16	16
24	780763	730866	730866	738800	1.09	14614	19549	17744	14567	-9.23	-25.5	24	16
32	627148	647043	647043	653060	0.93	18062	17289	15445	10898	-10.6	-36.9	32	16
40	534329	608139	608139	611087	0.48	24926	14547	11826	8240	-18.7	-43.4	38	16
48	500868	584380	584380	587328	0.50	32084	25532	23402	19181	-8.34	-24.9	48	16
56	435664	562561	562561	563882	0.23	37302	17138	15739	10754	-8.16	-37.3	56	16
64	421677	550549	550549	553497	0.54	35286	17642	16548	11320	-6.20	-35.8	64	16
	p34392												
16	2611379	2100312	2100312	2088962	-0.54	14296	15154	13595	7760	-10.29	-48.79	16	16
24	1868264	1802341	1802341	1791078	-0.62	9236	27624	22766	17422	-17.59	-36.93	24	16
32	1498194	1592668	1592668	1595286	0.16	10373	25220	20946	13949	-16.95	-44.69	32	16
40	1441485	1579078	1579078	1538795	-2.55	16661	43384	34172	24439	-21.23	-43.67	40	16
48	1422943	1572840	1572840	1567728	-0.33	14591	33544	28460	23705	-15.16	-29.33	37	16
56	1420363	1571728	1571728	1819571	15.77	12754	34068	29669	23423	-12.91	-31.25	53	16
64	1413013	1571728	1571728	1758060	11.86	19529	34068	29669	23145	-12.91	-32.06	53	16
	p93791												
16	3891988	3631177	3631177	3726714	2.63	21261	35709	32674	18803	-8.5	-47.34	16	16
24	2492638	2782827	2782827	2791223	0.30	40449	42034	36818	23727	-12.41	-43.55	24	16
32	1857887	2347166	2347166	2390750	1.86	58976	41403	38383	20953	-7.29	-49.39	32	16
40	1553100	2099123	2099123	2153380	2.58	54216	44550	40509	24709	-9.07	-44.54	40	16
48	1281244	1932476	1932476	1946263	0.71	82712	43546	37730	22749	-13.36	-47.76	48	16
56	1128667	1814195	1814195	1842030	1.53	73107	61754	54135	33205	-12.34	-46.23	56	16
64	988558	1708376	1708376	1737586	1.71	88378	52686	46992	26974	-10.81	-48.80	64	16
	t512505												
16	27190110	23494872	23494872	23494872	0	8522	2693	2409	1974	-10.55	-26.70	16	16
24	26778656	23417347	23417347	23494872	0.33	4449	2687	2403	1968	-10.57	-26.76	24	16
32	17510811	18232745	18232745	18310270	0.43	8621	1723	1441	1004	-16.37	-41.73	32	16
40	13028909	18192297	18192297	18438359	1.35	4718	2721	2405	2002	-11.61	-26.42	29	16
48	12967247	18192297	18192297	18352952	0.88	4143	2721	2405	2002	-11.61	-26.42	29	16
56	12967247	18192297	18192297	18269822	0.43	3084	2721	2405	2002	-11.61	-26.42	29	16
64	12967247	18192297	18192297	18482093	1.59	2940	2721	2405	2002	-11.61	-26.42	29	16

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.
#Test-Pin denotes the pre-bond test pin count.

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.

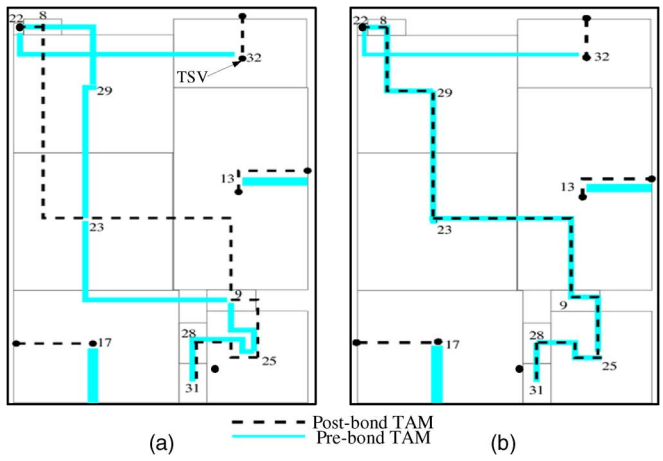


Fig. 14. Pre-bond TAM routing in p93791. (a) Without reusing post-bond TAMs. (b) Reusing post-bond TAMs.

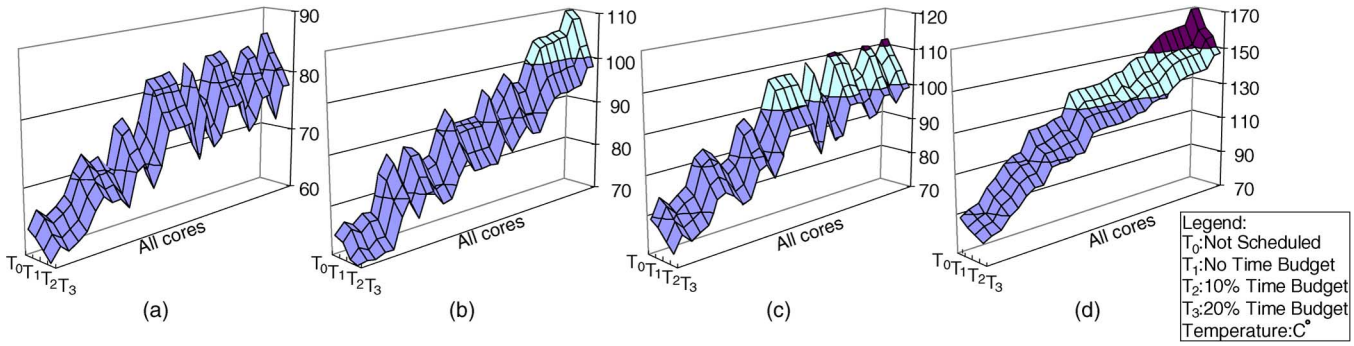


Fig. 15. Case study: steady temperature of all cores in SoC p93791. (a) $\mathrm{Width} = 16$. (b) $\mathrm{Width} = 32$. (c) $\mathrm{Width} = 48$. (d) $\mathrm{Width} = 64$.

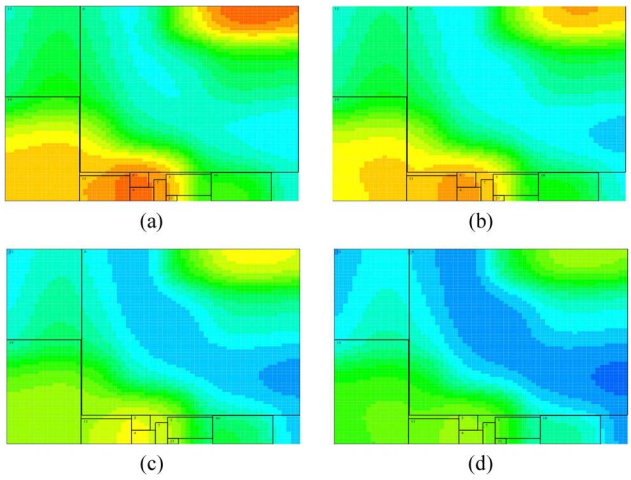


Fig. 16. Hot-spot simulated temperature of 48-bit TAM width using top layers floorplaning as background. (a) Before scheduling. (b) No idle time. (c) Idle time with 10% budget. (d) Idle time with 20% budget.

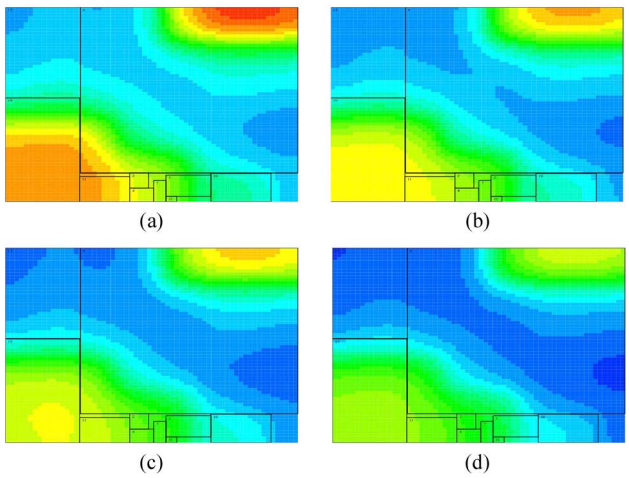


Fig. 17. Hot-spot simulated temperature of 64-bit TAM width using top layers floorplaning as background. (a) Before scheduling. (b) No idle time. (c) Idle time with 10% budget. (d) Idle time with 20% budget.

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.

We have also compared our experimental results with [9] in Table I. As can be seen from the table, while the proposed solution always has lower routing cost, [9] results in smaller testing time when the given TAM width is large. This is mainly because there is no pre-bond test-pin-count constraint in [9], and hence it can make use of more TAM width during pre-bond testing with reduced testing time when the total TAM width is large.

In Fig. 14, we present the layout of one layer in 3-D SoC p93791 to demonstrate the effectiveness of our TAM reuse strategy (see). 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. 14(a) depicts the test wires without reusing any post-bond TAM segments. With our reuse methodology as shown in Fig. 14(b), we can see that the routing overhead for TAMs can be significantly reduced.

Fig. 15 demonstrates the effectiveness of the proposed thermal-aware scheduling technique for SoC p93791. In most cases, as the total TAM width increases, the testing time is reduced, but the hot-spot temperature is increased. This is because there are usually more TAMs in a test architecture with a larger TAM width, and hence more cores are under test simultaneously, generating more heat. With some relaxation on testing time, the hot spot temperature continues to reduce significantly (from T_1 to T_3).

Figs. 16 and 17 demonstrate the temperature distribution in 3-D SoC p93791 using the floorplanning of the top layer as the background, when the total TAM width is set to be 48 and 64 in respect. As can be seen from these figures, there are two hot spots in the original test schedule. Using the proposed thermal-aware test scheduling algorithm, the temperature of most cores are reduced and there are less hot spots in the final test schedule.

VII. Conclusion

In 3-D technology, test pads occupy a much larger area when compared with 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 3-D 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. In addition, a thermal-aware test scheduling algorithm is utilized to reduce the hot spot temperature in post-bond testing. Experimental results for benchmark circuits demonstrate the effectiveness of the proposed solution.

REFERENCES

[1] W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer, and P. D. Franzon, "Demystifying 3-D ICs: The pros and cons of going vertical," *IEEE Design Test Comput.*, vol. 22, no. 6, pp. 498–510, 2005.

[2] "Yole development: Market trends for 3-D stacking." [Online]. Available: http://www.yole.fr

[3] Y. Xie, G. H. Loh, B. Black, and K. Bernstein, "Design space exploration for 3-D architectures," *J. Emerg. Technol. Comput. Syst.*, vol. 2, no. 2, pp. 65–103, 2006.

[4] G. H. Loh, Y. Xie, and B. Black, "Processor design in 3-D die-stacking technologies," *IEEE Micro*, vol. 27, no. 3, pp. 31–48, 2007.

[5] T. Fukushima, Y. Yamada, H. Kikuchi, and M. Koyanagi, "New three-dimensional integration technology ssing chip-to-wafer bonding to achieve ultimate super-chip integration," *Jpn. J. Appl.*, vol. 45, no. 4B, p. 3030, 2006.

[6] G. Smith, L. Smith, S. Hosali, and S. Arkalgud, "Yield considerations in the choice of 3-D technology," in *Proc. Int. Symp. Semicond. Manuf.*, 2007, pp. 1–3.

[7] H. H. S. Lee and K. Chakrabarty, "Test challenges for 3-D integrated circuits," *IEEE Design Test Comput.*, vol. 26, no. 5, pp. 26–35, 2009.

[8] E. J. Marinissen and Y. Zorian, "Testing 3-D chips containing through-silicon vias," in *Proc. IEEE Int. Test Conf.*, 2009, pp. 1–11.

[9] L. Jiang, L. Huang, and Q. Xu, "Test architecture design and optimization for three-dimensional SoCs," in *Proc. IEEE/ACM Design, Autom., Test Europe*, 2009, pp. 220–225.

[10] J. Cong, J. Wei, and Y. Zhang, "A thermal-driven floorplanning algorithm for 3-D ICs," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, 2004, pp. 306–313.

[11] Q. Xu and N. Nicolici, "Resource-constrained system-on-a-chip test: A survey," in *Proc. Inst. Electr. Eng. Comput. Digit. Tech.*, 2005, vol. 152, no. 1, pp. 67–81.

[12] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test access mechanism optimization, test scheduling, and tester data volume reduction for system-on-chip," *IEEE Trans. Comput.*, vol. 52, no. 12, pp. 1619–1632, Dec. 2003.

[13] Y. Xia, M. Chrzanowska-Jeske, B. Wang, and M. Jeske, "Using a distributed rectangle bin-packing approach for core-based SoC test scheduling with power constraints," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2003, pp. 100–105.

[14] Q. Xu and N. Nicolici, "Multi-frequency test access mechanism design for modular SoC testing," in *Proc. IEEE Asian Test Symp.*, 2004, pp. 2–7.

[15] 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. IEEE Int. Test Conf.*, 2002, pp. 74–82.

[16] C. Liu, K. Veeraraghavan, and V. Iyengar, "Thermal-aware test scheduling and hot spot temperature minimization for core-based systems," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst.*, 2005, pp. 552–560.

[17] P. Rosinger, B. Al-Hashimi, and K. Chakrabarty, "Rapid generation of thermal-safe test schedules," in *Proc. Conf. Design, Autom. Test in Europe*, 2005, vol. 2, pp. 840–845.

[18] P. Rosinger, B. M. Al-Hashimi, and K. Chakrabarty, "Thermal-safe test scheduling for core-based system-on-chip integrated circuits," *IEEE Trans. Comput.-Aided Des. (CAD) Integr. Circuits Syst.*, vol. 25, no. 11, pp. 2502–2512, Nov. 2006.

[19] Z. He, Z. Peng, P. Eles, P. Rosinger, and B. M. Al-Hashimi, "Thermal-aware SoC test scheduling with test set partitioning and interleaving," *J. Electron. Testing*, vol. 24, no. 1, pp. 247–257, 2008.

[20] T. M. Mak, P. Garrou, C. Bower, and P. Ramm, "Handbook of 3-D Integration: Technology and Applications using 3-D Integrated Circuits," in *Testing of 3-D Circuites*. New York: Wiley-CVH, 2008.

[21] D. L. Lewis and H.-H. S. Lee, "A scan-island based design enabling pre-bond testability in die-stacked microprocessors," in *Proc. IEEE Int. Test Conf.*, 2007, pp. 1–8.

[22] X. Wu, P. Falkenstern, and Y. Xie, "Scan chain design for three-dimensional integrated circuits (3D ICs)," in *Proc. Int. Conf. Comput. Des.*, 2007, pp. 208–214.

[23] E. J. Marinissen, J. Verbree, and M. Konijnenburg, "A structured and scalable test access architecture for TSV-based 3-D stacked ICs," in *Proc. VLSI Test Symp.*, 2010, pp. 269–274.

[24] X. Wu, Y. Chen, K. Chakrabarty, and Y. Xie, "Test-access mechanism optimization for core-based three-dimensional SOCs," in *Proc. Int. Conf. Comput. Des.*, 2008, pp. 212–218.

[25] B. Noia, S. K. Goel, K. Chakrabarty, E. J. Marinissen, and J. Verbree, "Test-architecture optimization for TSV-based 3-D stacked ICs," in *Proc. IEEE Eur. Test Symp.*, 2010, pp. 24–29.

[26] L. Jiang, Q. Xu, K. Chakrabarty, and T. M. Mak, "Layout-driven test-architecture design and optimization for 3-D SoCs under pre-bond test-pin-count constraint," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2009, pp. 191–196.

[27] International Technology Roadmap for Semiconductors. [Online]. Available: http://www.itrs.net 2009

[28] L. Zhou, C. Wakayama, and C. Shi, "CASCADE: A standard super-cell design methodology with congestion-driven placement for three-dimensional interconnect-heavy very large-scale integrated circuits," *IEEE Trans. Comput.-Aided Des. (CAD) Integr. Circuits Syst.*, vol. 26, no. 7, pp. 1270–1282, Jul. 2007.

[29] H. H. Lee and K. Chakrabarty, "Test challenges for 3-D integrated circuits," *IEEE Design Test Comput.*, vol. 26, no. 5, pp. 26–35, 2009.

[30] S. Koranne, "Design of reconfigurable access wrappers for embedded core based SoC test," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 5, pp. 955–960, May 2003.

[31] E. Larsson and Z. Peng, "A reconfigurable power-conscious core wrapper and its application to SoC test scheduling," in *Proc. IEEE Int. Test Conf.*, 2003, pp. 1135–1144.

[32] Q. Xu and N. Nicolici, "Wrapper design for testing IP cores with multiple clock domains," in *Proc. IEEE/ACM Design, Autom., Test Europe*, 2004, pp. 416–421.

[33] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Co-optimization of test wrapper and test access architecture for embedded cores," *J. Electron. Testing: Theory Applic.*, vol. 18, no. 2, pp. 213–230, 2002.

[34] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-aware microarchitecture: Modeling and implementation," *ACM Trans. Architecture Code Optim.*, vol. 1, no. 1, pp. 94–125, 2004.

[35] S. K. Goel and E. J. Marinissen, "Layout-driven SoC test architecture design for test time and wire length minimization," in *Proc. Des., Autom., Test Europe Conf.*, 2003, pp. 738–743.

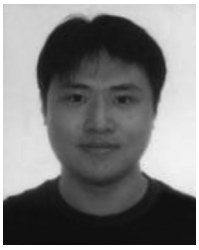
[36] T. E. Yu, T. Yoneda, K. Chakrabarty, and H. Fujiwara, "Test infrastructure design for core-based system-on-chip under cycle-accurate thermal constraints," in *Proc. Asia South Pacific Design Autom. Conf.*, 2009, pp. 793–798.

[37] T. E. Yu, T. Yoneda, K. Chakrabarty, and H. Fujiwara, "Thermal-safe test access mechanism and wrapper co-optimization for system-on-chip," in *Proc. Asian Test Symp.*, 2007, pp. 187–192.

[38] S. K. Goel and E. J. Marinissen, "SoC test architecture design for efficient utilization of test bandwidth," *ACM Trans. Design Autom. Electron. Syst.*, vol. 8, no. 4, pp. 399–429, 2003.

Li Jiang (S'08) received the B.S. degree in computer science and technology from Shanghai Jiaotong University, Shanghai, China, in 2007. He is currently working toward the Ph.D. degree at the CUHK Reliable Computer Laboratory (CURE Laboratory), Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong.

His research interests include VLSI testing and fault-tolerant computing.



Qiang Xu (M'06) received the B.E. and M.E. degrees in telecommunication engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 1997 and 2000, respectively, and the Ph.D. degree in electrical and computer engineering from McMaster University, Hamilton, ON, Canada.

Since 2005, he has been an Assistant Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, where he leads the CUHK Reliable Computing Laboratory (CURE Lab.). His research interests range from test and debug of system-on-a-chip integrated circuits to fault tolerance and reliable computing. He has authored or coauthored more than 70 technical papers in these areas.

Dr. Xu received the Best Paper Award at the 2004 IEEE/ACM Design, Automation and Test in Europe Conference (DATE) and has several other papers nominated for best paper award at prestigious conferences like ICCAD and DATE. He is currently serving as an Associate Editor for *IEEE Design and Test of Computers*.

Krishnendu Chakrabarty (F'08) received the B.Tech. degree from the Indian Institute of Technology, Kharagpur, India, in 1990, and the M.S.E. and Ph.D. degrees from the University of Michigan, Ann Arbor, in 1992 and 1995, respectively.

He is now Professor of Electrical and Computer Engineering with Duke University, Chapel Hill, NC. He is also a Chair Professor in Software Theory with Tsinghua University, Beijing, China. His current research projects include: testing and design-for-testability of integrated circuits; digital microfluidics, biochips, and cyberphysical systems; optimization of digital print and production system infrastructure. He has authored 10 books on these topics (with two more books in press), published over 360 papers in journals and refereed conference proceedings, and given over 140 invited, keynote, and plenary talks. He is the Editor-in-Chief of the *ACM Journal on Emerging Technologies in Computing Systems* and serves as an editor of the *Journal of Electronic Testing: Theory and Applications*.

Prof. Chakrabarty is a Golden Core Member of the IEEE Computer Society and a Distinguished Engineer of the Association for Computing Machinery. He was a 2009 Invitational Fellow of the Japan Society for the Promotion of Science (JSPS). He was a recipient of the 2008 Duke University Graduate School Dean's Award for excellence in mentoring and the 2010 Capers and Marion McDonald Award for Excellence in Mentoring and Advising, Pratt School of Engineering, Duke University. He was also a recipient of the National Science Foundation Early Faculty (CAREER) Award, the Office of Naval Research Young Investigator Award, the Humboldt Research Fellowship from the Alexander von Humboldt Foundation, Germany, and several best papers awards at IEEE conferences. He served as a Distinguished Visitor of the IEEE Computer Society during 2005–2007 and as a Distinguished Lecturer of the IEEE Circuits and Systems Society during 2006–2007. Currently, he serves as an ACM Distinguished Speaker as well as a Distinguished Visitor of the IEEE Computer Society for 2010–2012. He is the Editor-in-Chief of *IEEE Design & Test of Computers*. He is also an associate editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II, and the IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS.

T. M. Mak (SM'06) received the bachelor's degree from Hong Kong Polytechnic University.

He is a Technologist with Intel Corporation's Sort/Test Technology Development Group, working on test methodology development. He has been with Intel Corporation, Santa Clara, CA, for over 27 years and has worked on a variety of areas including test development, product engineering, design automation, and design for test. He mentored MARCO/FCRP (Focus Center Research Program) research for five years. He twice (1997 and 2004) received the SRC Outstanding Industrial Mentor Award. His current research interest ranges from defect-based testing, fault effects as a result of nanometer technology, circuit-level and physical design test issues, IO interface, memory and analog testing, and fault-tolerant and online testing. He received the Best Paper Award at the 2004 International Test Conference and a Best Panel Award from 2004 VTS. He currently holds 14 patents with one pending.