國立交通大學

電信工程研究所

碩 士 論 文

應用於三維積體電路在矽穿孔的限制下
的掃描鏈重排序設計方法

Through-Silicon-Via(TSV)-constrained Scan Chain Reordering
for Three-dimensional(3D) Circuits

研 究 生：陳韋廷

指導教授：溫宏斌　教授

中 華 民 國 九 十 九 年 七 月

應用於三維積體電路在矽穿孔的限制下的掃描鏈重排序設計方法

Through-Silicon-Via(TSV)-constrained Scan Chain Reordering for Three-dimensional(3D) Circuits

研 究 生：陳韋廷　　Student：Wei-Ting Chen

指導教授：溫宏斌　　Advisor：Hung-Pin Wen

國 立 交 通 大 學

電信工程研究所

碩 士 論 文

A Thesis

Submitted to Institute of Communication Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Communication Engineering

July 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年七月

應用於三維積體電路在矽穿孔的限制下的掃描鏈重排序設計方法

學生：陳韋廷　　指導教授：溫宏斌

國立交通大學電信工程研究所碩士班

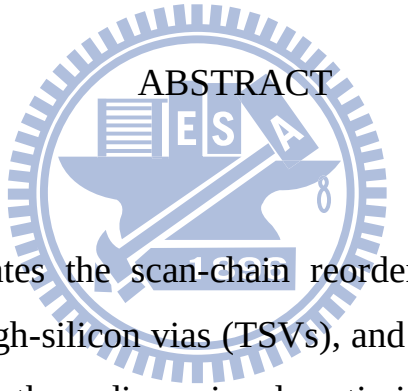
摘　　要

本論文定義出在利用一定數量的矽穿孔的掃描鏈重排序問題，而且提出了一個有效率的兩階段演算法去解決該問題。在三維積體電路最佳化中，我們在第一階段使用 *Multiple Fragment Heuristic* 的貪婪演算法並且使用處理最靠近點對的資料結構 *FastPair* 去得到一個好的初始解，包括掃描鏈線長和測試時的功率消耗。而在演算法的第二階段，提出了三維平坦化(*3D Planarization*)去減少所使用的線長和功率消耗，也提出了三維釋放(*3D Relaxation*)去減少矽穿孔的使用量以符合數量限制。最後，實驗結果顯現出該演算法比以基因演算法為基礎的先前技術，在可以比較的效能下，提出的演算法比先前技術快上百倍以上。由此證明，該演算法可以實際應用在三維積體電路以矽穿孔數量為限制的掃描鏈重排序。

Through-Silicon-Via(TSV)-constrained Scan Chain Reordering for Three-dimensional(3D) Circuits

student：Wei-Ting Chen　　Advisors：Dr. Hung-Pin Wen

Institute of Communication Engineering
National Chiao Tung University



ABSTRACT

This thesis formulates the scan-chain reordering problem considering a limited number of through-silicon vias (TSVs), and further develops an efficient 2-stage algorithm. For three-dimensional optimization, a greedy algorithm named *Multiple Fragment Heuristic* combined with a dynamic closest-pair data structure *FastPair* is proposed to derive a good initial solution at stage 1. Later, stage 2 proceeds two local refinements *3D Planarization* and *3D Relaxation* to reduce the wire/power cost and the number of TSVs in use, respectively. Experiments show that the proposed algorithm can result in a comparable performance to a genetic-algorithm-based method but can run at least 2-order faster, which evidently makes it more practical for TSV-constrained scan-chain reordering for 3D ICs.

誌　　謝

本論文可以順利完成，要感謝指導教授溫宏斌老師細心且不離不棄的指導。在這兩年的研究過程，曾有一度因遭受失敗而迷失了自己，用自我封閉來沉澱心情，然而，等我再次擁抱研究時，老師毫不猶豫接納我，並指導我研究的方向，更重要的是，老師的教學態度使我有能力且不懼怕的去處理任何事，讓我建立起自信，也對研究產生熱情和積極的態度。對未來，這段碩士生活給予了極大的幫助。

再來，要感謝的是 CIA 實驗室的所有成員。心思細膩的振源、好運動且勇於嘗試的佳伶、愛說話的開心果千慧、又懶惰又有效率的雨欣、好學的運動健將彥后，都是陪伴我兩年的好夥伴。怡璋、南旭，還有優秀的家慶、欣恬，期待你們可以在以後的日子裡發光發熱。

接著要感謝我的父母和弟弟，在求學的過程，給予經濟和精神上的支持。

最後，要感謝我的摯愛雨姍，一路上有你，苦一點也願意。自從大學時和你交往後，就有了努力的動機，畢業之後，也順利考上了理想的研究所，因為有你，我的生命慢慢增加了價值。而在這兩年，也因為有你的支持與諒解，所有事情都變得簡單了。

我要感謝身邊的所有人，因為即使只有小小的互助，但也成就了大大的美好。給予大家一句賈伯斯的名言，「Stay Hungry，Stay Foolish。」，共勉之。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Interconnect along with technology scaling plays an important role in deciding circuit performance. Structural three-dimensional (3D) integration is emerging as a promising solution to reduce the length of long interconnects across the circuit [12]. Moreover, 3D integration provides many other advantages over the traditional 2D implementation, such as better packaging efficiency and higher transistor density. These advantages, collectively, not only provide significant performance improvement but also alleviate the problems caused by lengthy interconnects [1, 5, 13, 23]. Therefore, different vertical-integration techniques have been proposed for manufacture, including wire bounds, solder balls, contactless, and through silicon vias (TSVs) [17]. Among all, TSVs provide the best performance of interconnect on timing and power.

Electrical characteristics of TSVs from different processes could result in different performance for 3D IC designs. Several types of TSVs with different aspect ratios are studied in [11, 14, 19, 20]. In general, the resistance of single TSV is small and the corresponding capacitance is proportional to the height of TSV. The impact of self-inductance and mutual inductance of TSVs has not been well-studied. But a long path through a large number of TSVs is known to bring potential and unpredictable timing failure. According to [4], RTI demonstrated that the current yield of TSVs is about 99.98% from manufacturing 65536 TSVs. Therefore, the yield for a design with 100 TSVs would drop to 98.01% due to manufacturing defect in TSVs. If the design contains 1000 TSVs, the yield would further drop to 81.88Considering the yield loss, the usage of TSV is typically limited during the designs of 3D ICs.

Scan chain design is the most prevailing DfT (Design-for-Testability) technique which aims to reduce the difficulty of testing on the circuit-under-test (CUT). In order to guarantee high fault coverage on complex designs, the CUT is modified during the synthesis stage to enhance its controllability and observability. All flip-flops (FFs) are replaced by multiplexed-input scan FFs with multiple operation modes. A conceptual scan-based design is illustrated as Figure 1. During the test mode, i.e. signal *test* is activated, the values of one test pattern are shifted to FFs of the scan chain in sequel. Later, the pattern is applied to the combinational logic through the primary inputs under the normal mode. The response values are finally captured at the primary outputs and shifted out through the scan chain under test mode again. Scan test reduces the sequential problem into the combinational

problem and thus can achieve high coverage efficiently.

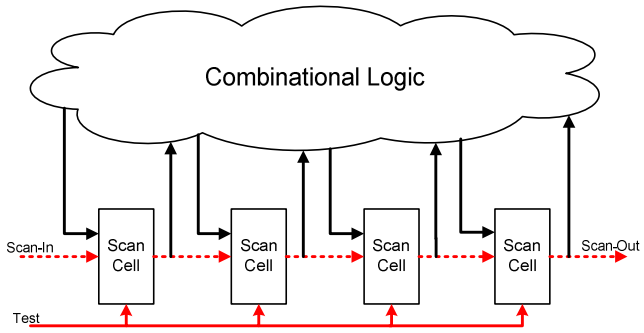


Figure 1.1: A general model for scan chain

Although scan cells enhance the testability on the CUT, the stitching wire of the scan chain can be lengthy and may deteriorate the signal integrity or even violate the timing constraint. Therefore, scan chain reordering referring to the order decision for scan cells based on the physical information, is widely studied and layout-based techniques [8, 9, 15] have been shown to reduce the scan stitching wire effectively.

Test power has always been the concern of scan test due to the trait of test pattern and the shift. The higher logic switching activities in the combinational logic resulting from patterns generated by ATPG and LFSR generated without considering the functionality of the circuit. The scan chain shift operation also causes the high toggle rate during testing. Generally, different methods have been reported to solve the power-related problem in the CUT, such as power-aware test pattern generation, test-pattern-filling technique, primary input controlling, and scan chain reordering. Scan chain reordering technique offers several advantages over other technique, including no negative effects in the test application time and fault coverage, easily combined to the design flow and other power reduction techniques. Several reordering techniques are proposed to reduce the power consumption during testing and also consider the scan stitching wire issue [3, 10, 16, 22].

To further study the interconnects in 3D ICs, Yuan et al [21] shows that the scan stitching wire in a multi-layer circuit is shorten comparing to wire length in the planar circuit. Several scan reordering approaches for 3D ICs are accordingly proposed: VIA3D, MAP3D and OPT3D. VIA3D uses the least number of TSVs to alleviate TSV impact on the scan stitching wire. MAP3D first maps all scan cells onto one single layer and then use the 2D

scan chain reordering. OPT3D considers the TSV impact in the computation of cost for the scan stitching wire. The OPT3D approach outperforms the other two in terms of total wire cost. The experimental results in [21] also suggest that the more TSVs in use in the scan chain, the smaller scan stitching wire cost. Such observation combing with the TSV-induced yield loss indicates an important tradeoff between the scan stitching wire and the number of TSVs in use. Therefore, the limitation of TSV in use must be considered due to the prevention of yield loss.

In this paper, TSV-based scan-chain reordering is first analyzed and formulated into a Traveling Salesman Problem (TSP). Later, a fast algorithm is developed to minimize the scan stitching wire or scan-induced power dissipation, and meet the limitation of TSVs in use simultaneously for 3D ICs. Our algorithm consists of two phases: first, we construct an initial simple path through all scan cells using a greedy algorithm named *Multiple Fragment Heuristic* via a dynamic closest pair data structure *FastPair*. Second, we propose new two local refinement techniques *3D Planarization* and *3D Relaxation* to reduce the wire cost or the power dissipation and to meet the TSV constraint, respectively. Experiments show the practicality of our algorithm by producing the comparable length of scan stitching wire to that from the genetic algorithm (GA) but runs at least three-order faster with the TSV constraint.

The rest of this paper is organized as follows. In Chapter 2, we present the problem formulation for TSV-constrained scan chain reordering for 3D ICs. In Chapter 3, a *multiple fragment heuristic* with the support of *FastPair* is first developed to obtain an initial good solution. Then *3D Planarization* and *3D Relaxation* for minimizing scan stitching wire cost or scan-induced power dissipation, and meeting the limitation of TSVs in use are detailed, respectively. Chapter 4 shows the experimental results including the comparison between our algorithm and the GA algorithm with and without the limitation of TSVs in use in terms of scan stitching wire and runtime over a variety of benchmark circuits. Finally, we draw our conclusion and outline future works in Chapter 5.

Chapter 2

Problem Formulation for TSV-based 3D Scan Chain Reordering

In this section, we formulate two scan chain reordering problems for 3D ICs. One target is to optimize the scan stitching wire for preventing the routing congestion and the timing violation; the other is to minimize the scan-induced power dissipation during testing in order to avoid the damage and the reliability degradation for CUT. We will first introduce the traditional scan reordering problem for wire reduction and define a new model for TSV-based 3D ICs. Then we will review the background knowledge on power-optimized scan reordering, and we will define the 3D power-optimized scan reordering problem at last.

2.1 Minimizing Scan-stitching Wire

The problem of planar scan-chain reordering to minimize the scan stitching wire cost can be formulated into:

Input: a CUT C with n scan cells $\{c_0, c_1, \ldots, c_{n-1}\}$ and their locations $\{(x_0, y_0), (x_1, y_1), \ldots, (x_{n-1}, y_{n-1})\}$

Output: a scan-cell ordering is formed as follow $\langle c_{\pi(0)}, c_{\pi(1)}, \ldots, c_{\pi(n-1)} \rangle$ such that the total cost of scan stitching wire

$$\sum_{i=1}^{n-1} \left| x_{\pi(i)} - x_{\pi(i-1)} \right| + \left| y_{\pi(i)} - y_{\pi(i-1)} \right| \tag{2.1}$$

is minimized.

In Equation (2.1), $x_{\pi(i)}$ and $y_{\pi(i)}$ denote the x and y coordinates of the i^{th} scan cell in the formed scan-cell ordering, respectively. All *FF*s are on the same plane and the stitching wire cost is defined as the Manhattan distance between c_i and c_{i+1} in this formulation. However, for 3D ICs, *FF*s can be located across different layers and thus the TSV cost for connecting two cross-layer *FF*s is not considered. In order to consider the TSV cost in TSV-based 3D scan-chain reordering, the layer information needs to be included:

$$\{(x_0, y_0, L_0), (x_1, y_1, L_1), \ldots, (x_{n-1}, y_{n-1}, L_{n-1})\}$$

And the total cost of scan stitching wire is modified as follow

$$\sum_{i=1}^{n-1} \left|x_{\pi(i)} - x_{\pi(i-1)}\right| + \left|y_{\pi(i)} - y_{\pi(i-1)}\right| \\ +c_{TSV} \times \left|L_{\pi(i)} - L_{\pi(i-1)}\right| \tag{2.2}$$

In Equation (2.2), c_{TSV} denotes the equivalent stitching wire cost for one TSV connecting two consecutive layers. Without losing the generality, c_{TSV} is defined as the height H_{TSV} for one single-layer TSV in this paper. Hence, the total TSV cost in the stitching wire is denoted as TC_{total} and modeled into:

$$TC_{total} = \sum_{i=1}^{n-1} \left|L_{\pi(i)} - L_{\pi(i-1)}\right| \tag{2.3}$$

According to such modified formulation for TSV-based scan-chain reordering problem, two approaches are proposed in [21]: one is developed on the basis of genetic algorithm (GA), and the other is based on integer linear programming (ILP). Although the GA-based approach may possibly find the near-optimal solution, the quality of one found solution cannot be guaranteed under the limited time. Moreover, the ILP-based approach that shall find the optimal solution may not be able produce a feasible solution within the limited time. Experimental results in [21] shows the ILP-based approach can estimate the lower-bound values for the total scan stitching wire cost quickly, but cannot produce a feasible ordering of scan cells in time.

To be more practical, a new fast algorithm needs to be developed and overcomes the runtime issue. Therefore, we propose an efficient 2-stage algorithm. In stage 1, we convert the 3D scan-chain reordering problem into a TSP problem. Then, a tour-construction heuristic [2] with the support of a particular closest-pair data structure named *FastPair* [6] to stitch a simple path as one initial solution. During stage 2, the local refinement by *3D Planarization* and constraint solving by *3D Relaxation* start to minimize the total wire cost and to reduce the number of TSVs in use, respectively. Figure 2.1 shows the overall flow, and more details are given in the following section. Note that although the proposed algorithm is currently applied for one scan chain, it is easy to scale to support multiple scan chains when scan-cell partitioning is available.

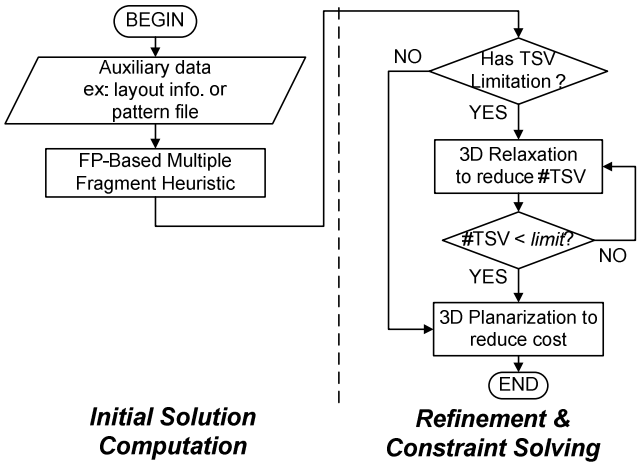


Figure 2.1: Flow of proposed scan reordering algorithm

2.2 Reducing Scan-induced Power

In this problem, the goal of scan chain reordering is to find a scan chain ordering while minimizing the power dissipation during the scan shifting operations. Integrate scan chain reordering techniques into the current design flow would be easy while maintaining the original fault coverage and test application time. The only limitation for scan chain ordering techniques is that it must target a fixed set of test patterns generated by ATPG. In the following subsections, we will briefly introduce the background on power-aware scan chain reordering and then formulate this problem for TSV-based 3D ICs.

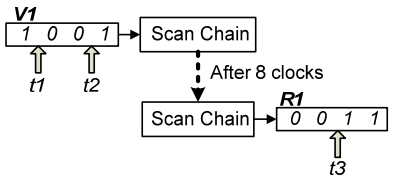
Estimation for Power Dissipation

Previous power-optimized reordering techniques concern the total power and the peak power consumption. The total power consumption is the sum of power consumed during testing, and the peak power consumption is the highest power consumption at any given test pattern. Therefore, the dynamic power consumption is modeled into

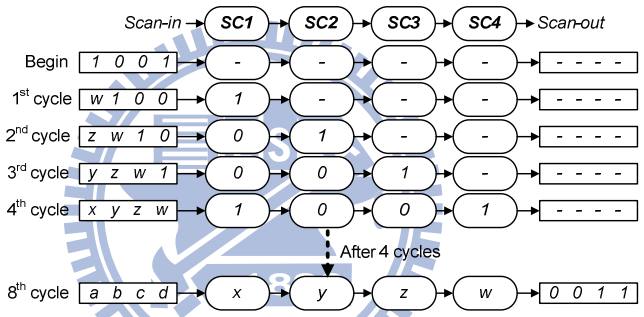
$$P = 0.5 \cdot C_{ld} \cdot V_{dd}^2 \cdot F \cdot S \tag{2.4}$$

where P is the dynamic power consumption, C_{ld} is the load capacitor, V_{dd} is the supply voltage, S is the switching activity, and F is the clock frequency.

According to Equation (2.4), the power consumption during scan shifting operations completely depends on the switching activities in CUT. In practice, it is time-consuming to count the exact number of all switching activities in CUT. It has been proved in [18] that the number of scan chain transitions and the triggered logic elements transitions in CUT are highly correlated. In the other word, the number of transitions in the scan chain is a good estimation for total switching activities in CUT.



(a) Simple view



(b) Detailed view

Figure 2.2: Transitions of one test pattern for 4-cell scan chain

Figure 2.2(a) shows a 4-cell scan chain by applying an input vector and capturing the corresponding output response. There are two transition $t1$, $t2$ in the input vector *V1* and one transition $t3$ in output response *R1*. And we use 8 clocks to finish feeding the input vector and capturing the output response.

Figure 2.2(b) describes the details of operations in this case. We assume that the unknown states are stored in the 4 scan cells. At the first clock, the first value is scanned in the first scan cell *SC1*. After scanning in the second value 0, the state of the scan cell *SC1* will change from 1 to 0. This transition $t1$ will cause the switching activities of related logic elements in CUT. At the third clock, the transition $t1$ makes the value toggle of the scan

cell *SC2*, which switches the related logic elements in the CUT; the same condition occurs at the fourth clock for the scan cell *SC3*. On the other hand, the transition $t2$ only toggle the state of the scan cell *SC1* at the fourth clock during scan-in operation. Therefore, the first transition created by the test pattern causes the largest number of switching activities in the CUT.

The second vector (w, x, y, z) is applied after using 4 clock cycles to apply the first test vector *V1*. Meanwhile, the first output response $(0, 0, 1, 1)$ is captured at the same time. Hence, the transition $t3$ also causes the related switching activities in the CUT and the total number of scan-out operations depends on the position of $t3$ in the scan chain. In contrast of scan-in operations, the last-out transition in the output response causes the largest number of switching activities in the CUT among the other transitions. Consequently, the total switching activities in CUT during shifting operations depend on the transitions in the scan chain and the corresponding positions. Thus, [18] defined the number of weighted transitions as follow,

$$Weighted\_Transitions = \sum (Size\_of\_Scan\_Chain - Position\_of\_Transition)$$

where $Weighted\_Transitions$ represents the real switching activities in the CUT, $Size\_of\_Scan\_Chain$ is the total number of scan cells, and $Position\_of\_Transition$ is indexed from the different beginnings between input vector and output response. Hence, every transition in the input vector or output response has its own weight that reflects the real conditions. We will explain the weighted transitions by defining some notations used in the rest of paper.

$\{c_0, c_1, \ldots, c_{n-1}\}$: The n scan cells in the CUT C.

$O = \langle c_{\pi(0)}, c_{\pi(1)}, \ldots, c_{\pi(n-1)} \rangle$: The scan chain ordering with n scan cells.

$V = \{v_0, v_1, \ldots, v_{n-1}\}$: A n-bit input pattern, where v_i is scanned in the scan cell c_i during scan testing. Therefore, $\langle v_{\pi(0)}, v_{\pi(1)}, \ldots, v_{\pi(n-1)} \rangle$ represents an input pattern for a given scan chain ordering.

$R = \{r_0, r_1, \ldots, r_{n-1}\}$: A n-bit output response, where r_i is scanned in the scan cell c_i during scan testing. Therefore, $\langle r_{\pi(0)}, r_{\pi(1)}, \ldots, r_{\pi(n-1)} \rangle$ represents an output response for a given scan chain ordering.

The weighted transitions of an input vector V and an output response R are defined as Equation (2.5) and Equation (2.6), respectively.

$$VWT(V) = \sum_{i=1}^{n-1} i \cdot (v_{\pi(i)} \oplus v_{\pi(i-1)}) \tag{2.5}$$

$$RWT(R) = \sum_{i=1}^{n-1} (n-i) \cdot (r_{\pi(i)} \oplus r_{\pi(i-1)}) \tag{2.6}$$

where $VWT(V)$ and $RWT(R)$ are denoted as the weighted transitions for the input vector V and the output response R; the $\oplus$ operator checks the difference between two adjacent bits. The i and $(n-i)$ represent different weighting rules for scan-in and scan-out operations respectively. Generally, above equations can be easily extended into the following equations for m test patterns.

$$VWT(\{V^1, V^2, \ldots, V^m\}) = \sum_{i=1}^{n-1} \sum_{j=1}^{m} i \cdot (v^j_{\pi(i)} \oplus v^j_{\pi(i-1)}) \tag{2.7}$$

$$RWT(\{R^1, R^2, \ldots, R^m\}) = \sum_{i=1}^{n-1} \sum_{j=1}^{m} (n-i) \cdot (r^j_{\pi(i)} \oplus r^j_{\pi(i-1)}) \tag{2.8}$$

The V^j and R^j are the j^{th} input vector and the j^{th} output response in the set of m test pattern respectively, and the $v^j_{\pi(i)}(r^j_{\pi(i)})$ is the bit being scanned in the i^{th} scan cell of the chain ordering and located at the j^{th} input vector(output response).

In addition to scan-in and scan-out transitions, the peak transitions are taken account into the total weighted transitions. A peak transition is occurred when there is a difference between the last-out bit of the j^{th} output response and the first-in bit of the $(j+1)^{th}$ input vector. Since a peak transition causes all scan cells to toggle, the weight of peak transitions is the size of the scan chain. The weighted peak transition is denoted by PWT and defined as follow,

$$PWT = \sum_{j=1}^{m-1} n \cdot (r^j_{\pi(0)} \oplus v^{j+1}_{\pi(n-1)}) \tag{2.9}$$

Consequently, we denote the total weighted transition TWT which equals to the sum of VWT, RWT, and PWT.

Figure 2.3 shows two examples of calculating the total weighted transitions. The CUT with 4 scan cells has the scan chain ordering, and three test patterns are applied during scan testing. Hence, the total transitions, the transitions for input vectors, the transitions for output response, the peak transitions and the corresponding weights in different positions are shown in the Figure 2.3. In Figure 2.3(a), the scan chain has the initial ordering $(1, 2, 3, 4)$. Thus, $VWT(\{V^1, V^2, V^3\}) = 1 \cdot 1 + 1 \cdot 2 + 3 \cdot 3 = 12$, $RWT(\{R^1, R^2, R^3\}) = 2 \cdot 3 + 1 \cdot 2 + 1 \cdot 1 = 9$, and $PWT = 2 \cdot 4 = 8$. The total weighted transitions TWT is 12 + 9 + 8 = 29. However, Figure 2.3(b) represents the power-optimized ordering $(2, 3, 4, 1)$ by scanning in the same test patterns. Thus, $VWT(\{V^1, V^2, V^3\}) = 1{\cdot}1{+}3{\cdot}2{+}1{\cdot}3 = 11$, $RWT(\{R^1, R^2, R^3\}) = 1{\cdot}3{+}1{\cdot}2{+}2{\cdot}1 = 7$, and $PWT = 0 \cdot 4 = 0$. The total weighted transitions TWT is 11 + 7 + 0 = 18. Therefore, the total power reduction rate is almost 38% and the number of peak transitions is reduced from 2 to 0.

Formulation for TSV-based 3D ICs

The problem of scan-chain reordering to minimize the scan-shift power dissipation can be formulated into:

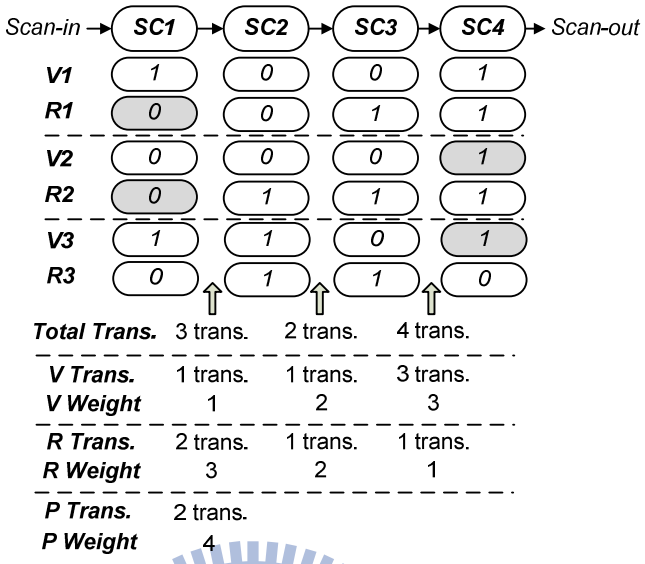
Input: a CUT C with n scan cells $\{c_0, c_1, \ldots, c_{n-1}\}$, their layer information $\{(L_0), (L_1), \ldots, (L_{n-1})\}$, and a fixed set of m test patterns $\{V^1, R^1, V^2, R^2, \ldots, V^m, R^m\}$

Output: a scan-cell ordering is formed as follow $\langle c_{\pi(0)}, c_{\pi(1)}, \ldots, c_{\pi(n-1)} \rangle$ such that the total weighted transitions $TWT(\{V^1, R^1, V^2, R^2, \ldots, V^m, R^m\})$ is minimized with(without) the TSV constraint

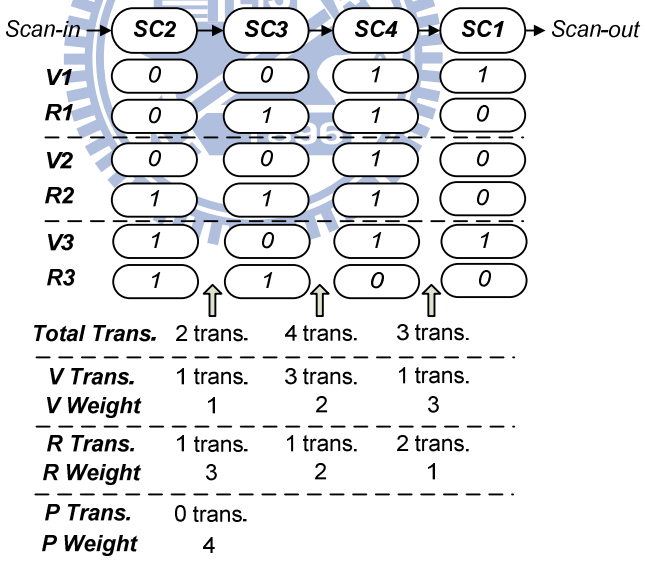
Comparing with the scan wire reduction problem, we only concern the layer information of the scan cells since the problem is not related to their locations and the objective function. Therefore, we only need to calculate the total TSV cost by using the Equation 2.3 under the limitation of TSV.

According to such formulation for power-minimized problem for TSV-based 3D ICs, [7] also uses the GA approach to solve this problem, which is time-consuming and unstable for the quality of solutions. We also use the same flow of the proposed algorithm, as

illustrated in Figure 2.1, to solve this power-optimized problem. Some processes need to be modified due to different objective functions. In the beginning, we establish a look-up table storing the pair-wise cost, because the high complexity of calculations for pair-wise cost function for the power-optimized problem. There are several modifications while performing proposed algorithm since the objective weighted transitions depends on the transition positions in the scan chain. However, we also overcome the runtime issue in this problem and more details are addressed in the next section.



(a) Initial ordering



(b) Power-optimized ordering

Figure 2.3: Calculations for weighted transitions

Chapter 3

Proposed Scan Chain Reordering Algorithm

In this section, we introduce this algorithm by considering scan wire and scan-induced power optimized problems in 3.1 and 3.2, respectively.

3.1 Minimizing Scan-stitching Wire

According to Figure 2.1, in stage one, *Multiple Fragment Heuristic* is applied because it is one of the state-of-the-art tour-construction heuristics known for TSP problems [2]. Moreover, a dynamic closest-pair data structure *FastPair* [6] can further be used to facilitate the considerable computation of pair-wise cost in the tour-construction heuristic. After stage 1, an initial solution is obtained and sent to stage 2 which performs *3D Planarization* to lower the total wire cost and *3D Relaxation* to reduce the total number of TSVs in use.

Initial Solution Computation

First, a good initial ordering of scan cells needs to be constructed in stage 1 and can be solved by one TSP heuristic called *Multiple Fragment Heuristic*. This heuristic finds the shortest edge between the endpoints of two different paths each time until all points are connected. Each point has its own path connecting to it itself in the beginning. The procedure of *Multiple Fragment Heuristic* is shown as follow,

$MULTIPLE\_FRAGMENT\_HEURISTIC(C)$
1 **for** each cell $c_i \in C$
2 **do** $endpoint(c_i) \leftarrow c_i$
3 **while** $|C| > 2$
4 **do** $(c_i, c_j) \leftarrow CLOSEST\_PAIR(C)$
5 $c_x \leftarrow endpoint(c_i)$
6 $c_y \leftarrow endpoint(c_j)$
7 $endpoint(c_x) \leftarrow c_y$
8 $endpoint(c_y) \leftarrow c_x$
9 **if** $c_x \neq c_i$
10 **then** delete c_i from C

11 **if** $c_y \neq c_j$

12 **then** delete c_j from C

The **for** loop of line 1-2 sets the *endpoint* of each cell to itself. The *endpoint* of each cell will be updated and then check if the illegal conditions occur. The **while** loop of lines 3-12 grows the traversal of scan cells and conforms to the property of the simple path until the number of point set C is less than two. Figure 3.1(a) is an example with no edge in the original graph. Then the first shortest edge between node *2* and node *3* denoted by *(2,3)* (i.e. the minimum cost) will be connected as the dotted line. Figure 3.1(b) shows the decision of the shortest edges among the remaining five points. The gray nodes are deleted in the early stages since they cannot form any new edges in a simple path. The solid line represents the connected edge while the dotted line connects the next candidate node. In this example, edge *(3,4)* is added among five points *1, 3, 4, 6, 7*. This path will continue growing until only two nodes *1, 7* in the CUT. Finally, connecting all points by *n-1* edges forms a simple path as an initial ordering of scan cells in Figure 3.1(c).

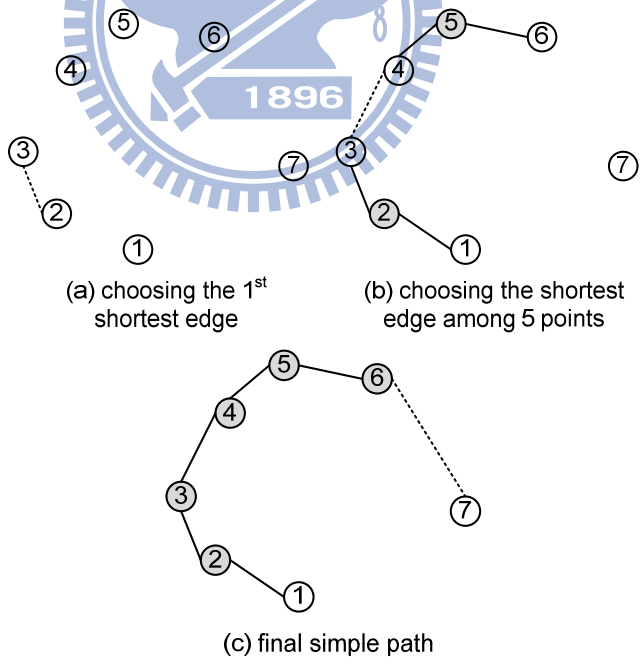


Figure 3.1: Applying *Multiple Fragment Heuristic* to 7 points

FastPair is first proposed for handling dynamic closest-pair problems with pair-wise cost functions [6] and similar to the neighbor heuristic that each point stores its nearest neighbor, creates initial neighbor values differently, and insertion never changes values of old neighbors. Before introducing the *FastPair* data structure, we must understand the neighbor heuristic. This heuristic maintains the closest point with each point p of point set S and the corresponding distance

$$d\,(p) = min_{q \in S-\{p\}} D(p, q) \tag{3.1}$$

which $D(p, q)$ is a user-defined function. Hence, this function computes the distance between scan cells. That is,

$$D_1(p, q) = |x_p - x_q| + |y_p - y_q| + c_{TSV} \times |L_p - L_q| \tag{3.2}$$

Therefore, these data are maintained for all operations, including insertion, deletion, and query. A query is scanning all the distances and selecting the smallest one. However, this heuristic is the simple maintenance for storing the closest point via graph structure and illustrated in Figure 3.2. Figure 3.2(a) shows the initialization of the neighbor heuristic. The closest nodes of nodes *1, 6, 7* are *2, 5, 6*, respectively; the closest nodes *2, 4* are *3, 5* and vice versa. After deleting the node *5*, two nodes *4, 6* need to update their closest nodes and then connect to nodes *3, 4*, respectively. Such result is illustrated as Figure 3.2(b).

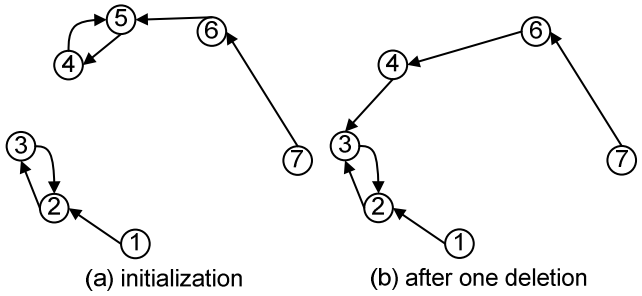


Figure 3.2: Illustration of the neighbor heuristic

FastPair can be explained in terms of the neighbor heuristic. In this method, the data structure is initialized with a single directed path, instead of computing all possible distances. Such structure promises only one update after deleting one node, which differs

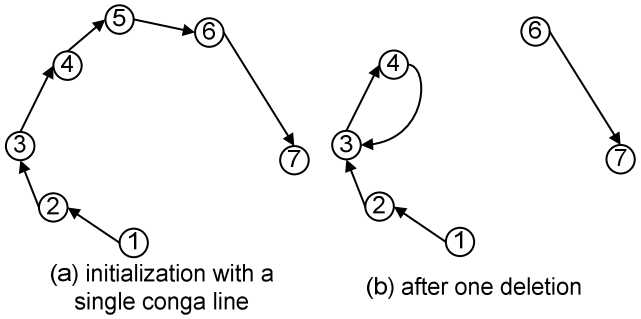


Figure 3.3: Illustration of the *FastPair* method

from the neighbor heuristic. Figure 3.3 shows a simple example with graph view. In the Figure 3.3(a), every node has its own closest node by only checking the node without any degree. Therefore, this method only updates a closest node of node *4* by deleting one node in the Figure 3.3(b).

Although no improving on the neighbor heuristic in the complexity bounds, *FastPair* still outperforms experimentally. In [6], runtimes to delete an object, and to query the closest pair among several different closest pair data structures are thoroughly compared, and *FastPair* stands out to be the best one for most applications. In our algorithm, *Multiple Fragment Heuristic* can be implemented in the greedy fashion with the support of *FastPair* and runs in the time complexity of $O(n^2)$.

Local Refinement & Constraint Solving

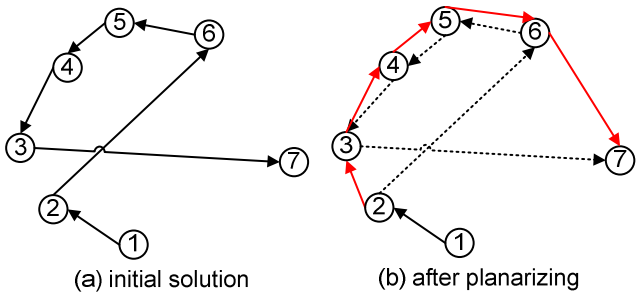
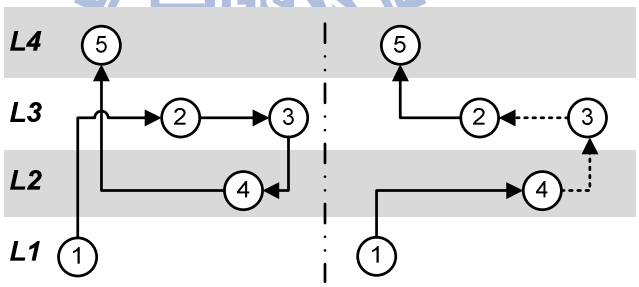


Figure 3.4: A 6-point example for *Planarization*

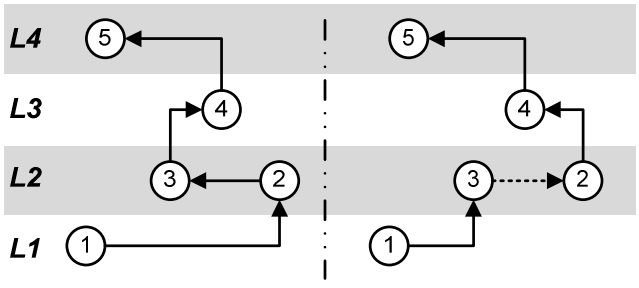
After having the initial solution, the second stage of our algorithm further applies two strategies to optimize the total wire cost and/or relax the TSVs in use. Figure 3.4(a) shows

an initial path with the un-optimized scan stitching wire cost. In the optimization study, (2D) Planarization is one of the most common techniques to reduce the total traversal cost for the TSP problem. The key idea behind is to transform a graph into a planar graph which does not contain any cross edge on the plane. A modified tour with cross-edge removal results in a shorter cost than that from the initial tour. Figure 3.4(b) shows such an example. Cross edges, *(2, 6)* and *(3, 7)* are replaced by edge *(2, 3)* and *(6, 7)*.

From a different point of view, such operation behaves like the reverse of a fragment, i.e. the sequence of node traversal. Reversing the fragment from $2 \rightarrow 6 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 7$ into $2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7$ can effectively reduce the total stitching wire cost. To generalize this idea, we reverse any possible fragment with edge length starting from of *1* to *(n-1)* and test if such reversion can reduce the total wire cost. Such *local refinement technique* is called *3D Planarization* and runs in time complexity of $O(k_1 n^2)$ where k_1 denotes the constant number of iterations. After constructing the initial solution, a small $k_1 << n$ usually suffice to do a good optimization in our experiment.



(a) Reducing the wire cost and #tsv



(b) Reducing the wire cost but no #tsv saving

Figure 3.5: An example for *3D Planarization*

The key reason by using above refinement technique is to avoid checking the cross

edges in the 3D space. Hence, Figure 3.5 shows two examples of using *3D Planarization* to reduce the total wire cost. In Figure 3.5, $L1, L2, L3, L4$ represents the $1^{st}, 2^{nd}, 3^{rd}, 4^{th}$ layer, respectively and the connection of two scan cells among two neighboring layers requires one TSV. Figure 3.5(a) shows a refinement example for the wire cost reduction and in-use TSV relaxation. The left part of Figure 3.5(a) represents an initial scan-chain ordering: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$. After reversing the fragment $2 \rightarrow 3 \rightarrow 4$ to $4 \rightarrow 3 \rightarrow 2$ shown as the right part of Figure 3.5(a), the total wire cost is reduced: the reversed fragment has the same cost, but the transition of $1 \rightarrow 2$ to $1 \rightarrow 4$ reduces the wire cost successfully as well as one TSV. The transition of $4 \rightarrow 5$ to $2 \rightarrow 5$ also reduces the extra wire cost and one more TSV. Similarly, Figure 3.5(b) shows another example for refinement but with no TSV relaxation. After reversing the fragment $2 \rightarrow 3$ to $3 \rightarrow 2$, no TSV can be saved but the wire cost can be reduced.

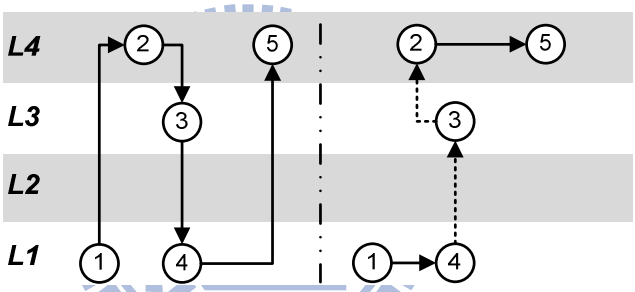


Figure 3.6: An example for *3D Relaxation*

A similar *constraint solving technique* named *3D Relaxation* to reduce the number of TSVs in use is also proposed and aims to fulfill the TSV constraint imposed for alleviating the yield loss. *3D Relaxation* reverses all fragments of *1* to *(n-1)* edges again to find the best reduction of TSVs in use until the target number is achieved. Later, *3D Planarization* is also performed to further reduce the total wire cost but use no extra TSV. Figure 3.6 shows an example for illustrating the *3D Relaxation* Process. The left part of Figure 3.6 represents an initial scan-chain ordering: $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$. After reversing the fragment $2 \rightarrow 3 \rightarrow 4$ to $4 \rightarrow 3 \rightarrow 2$, the total TSV cost can be effectively reduced and shown as the right part of Figure 3.6. The reversed fragment results in the same cost, but the replacement of connecting $1 \rightarrow 2$ to $1 \rightarrow 4$, $4 \rightarrow 5$ to $2 \rightarrow 5$, saves the total of six TSVs.

The time complexity for the constraint solving technique is also $O(k_2n^2)$ and k_2 depends on the relaxed TSV number, i.e. the difference between the initial and the limitation TSV number. The total complexity in the 2^{nd} phase is $T(n) = O(k_1n^2) + O(k_2n^2) = O(n^2)$. To sum up, the above 2-phase scan-chain reordering algorithm is efficient and can run much faster than previous techniques [21].

3.2 Reducing Scan-induced Power

We use the same flow of proposed algorithm to solve the power-optimized problem, and address the differences with the wire-minimized problem in this part. According to section 2.2, we take the pattern information as input, and the objective is to minimize the total weighted transitions. Again, the computation for the total weighted transitions needs to know the whole scan cell ordering due to the position-related property. However, we only get the scan-induced transitions between scan cells in the beginning.

In the initial solution computation, we change the user-defined function $D(p, q)$ in Equation (3.1) to count the scan-induced transitions between scan cells by considering the m test patterns. That is,

$$D_2(p,q) = \sum_{j=1}^{m} \left[\left(v_p^j \oplus v_q^j \right) + \left(r_p^j \oplus r_q^j \right) \right] \tag{3.3}$$

whose notations is defined in the section 2.2. Since every computation in the Equation(3.3) costs $O(m)$ in m test patterns, a look-up table storing the pair-wise transitions is established to avoid the huge number of calculations during performing proposed algorithm. After constructing the scan cells ordering, the sum of the total transitions between cells is minimized under the property of *Multiple Fragments Heuristic*. Therefore, we improve the total weighted transitions by rotating it n times and choose the best solution. Figure 2.3 shows a simple example. The total scan-induced transitions between cells are counted in the first row whose header is *Total Trans.*. Although the sum of the total transitions are the same in this two scan cell orderings, the ordering illustrated as Figure 2.3(b) has the best solution in the total weighted transition by rotating the initial ordering $SC1 \rightarrow SC2 \rightarrow SC3 \rightarrow SC4$ 3 times.

In the local refinement, a better solution is confirmed by checking the best total weighted transitions among the results by rotating n times. Although the construction of the look-up table spends more time than the cost computation in the scan-stitching wire minimized problem, the total time complexity is $O(n^2)$ and faster than the state-of-the-art technique proposed in [7].

Chapter 4

Experiment Results

A reference flow for 3D scan designs in [21] requires an academic placement-and-routing tool *MITPR3D* which is not available publicly. Therefore, we modify the flow and utilize commercial software for 3D scan designs and Figure 4.1 illustrates our flow for 3D scan designs. In the left part of Figure 4.1, we first partition a original design to N layers which minimizes the number of TSVs in use and balances the area between different layers. After obtaining N-layer designs, Design Compiler does logic synthesis for each layer designs. Then, all FFs are placed in N-layer designs with scan cells and placed by First Encounter. Finally, all planar placements are combined into one single 3D placement and output the locations of all scan cells from DEF (Design Exchange Format) files. Therefore, we get all layout information, which can perform the proposed algorithm for minimizing the scan-stitching wire cost.

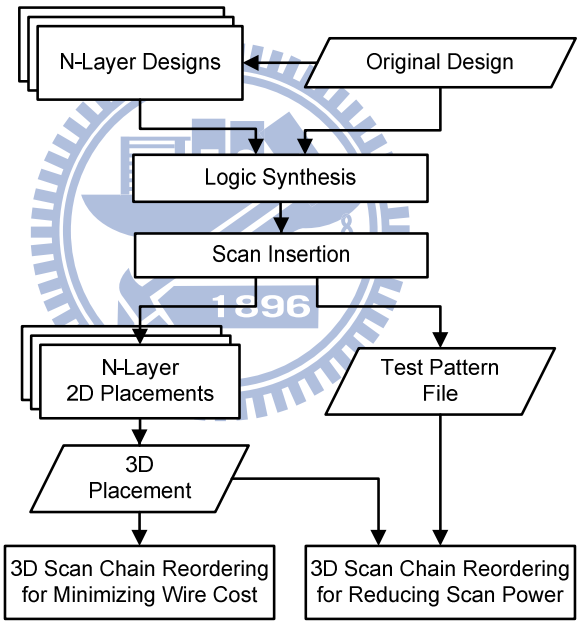


Figure 4.1: Proposed 3D scan design flow

In the right part of Figure 4.1, we also do logic synthesis and scan insertion for the original design and then get the test pattern information via the STIL (Standard Test Interface Language) file by using the commercial tool TetraMax. Two input data including the layout information and the test pattern are achieved to reduce the scan-induced power by performing the proposed algorithm.

For fair comparison, we use the same settings as the previous GA-based approach uses in [21]. The population size is set to be 2000; the same operators are used and include reproduction, crossover, mutation; the GA stops until no more than 0.0001% improvement on the fitness score (a.k.a. the total scan-stitching wire cost or the total weighted transitions) can be obtained for last 1000 generations.

Both the proposed 3D scan-chain reordering algorithm and previous GA-based approach are exercised on a Linux machine with a Pentium Core Duo (2.4GHz) processor and 4GB memory. TSMC .18μm library is used and the height of a TSV is set as 10μm while the partitions for 3D ICs range from 2 to 5. ISCAS89 benchmark circuits are used to conduct experiments.

In this section, all experiments are divided into two parts. The first part is to minimize the scan-stitching wire cost. In the second part, the results considering the reduction of scan-induced power are proposed.

4.1 Minimizing Scan-stitching Wire

Table 4.1 shows the performance of our algorithm without the limitation of TSVs for all circuits. The first column shows the name of ISCAS89 circuits and the total number of scan cells in the parenthesis; the second column shows the number of layer of 3D circuits; the third and fourth columns show the total wire cost in μm of the initial ordering and the new ordering, respectively; the fifth column shows the reduction rate of total wire cost from the initial ordering to the new one; the sixth and seventh columns show the TSV cost in the initial ordering and the reordering after performing the local refinement technique *3D Planarization*, respectively;

As we can see, fewer TSVs in the new ordering are used than that from the initial one since the initial solution serves as the initial point for the TSV optimization. Then, the technique *3D Planarization* shows a good reduction rate of 4.72% averagely over the initial solution on all circuits.

Table 4.2 shows the performance of our algorithm with the TSV constraint. The third and fourth columns show the total wire cost in μm after performing the constraint solver

Table 4.1: Performance for wire minimzation without the TSV constraint

circuit(#FF)	#layer	stage 1 (μm)	stage 2 (μm)	red. (%)	initial #TSV	final #TSV
s1423(74)	3	1338	1324	**1.05**	21	21
	4	1430	1386	**3.08**	27	25
	5	1208	1190	**1.49**	21	21
s5378(179)	3	3432	3246	**5.42**	44	44
	4	3222	3190	**0.99**	39	43
	5	3128	2990	**4.41**	28	28
s9234(211)	3	4786	4582	**4.26**	71	61
	4	4886	4598	**5.89**	81	73
	5	4108	3944	**3.99**	50	48
s15850(597)	3	12472	11714	**6.08**	181	161
	4	11984	11496	**4.07**	239	231
	5	11006	10360	**5.87**	153	155
s13207(669)	3	12736	11972	**6.00**	185	173
	4	11608	10928	**5.88**	139	123
	5	11566	11200	**3.16**	247	237
s38584(1452)	3	31570	29344	**7.05**	475	457
	4	30640	28780	**6.07**	575	551
	5	30676	28842	**5.98**	596	568
s38417(1636)	3	32314	30564	**5.42**	445	425
	4	31820	30076	**5.48**	592	572
	5	30446	28798	**5.41**	536	506
s35932(1728)	3	32380	30698	**5.19**	628	608
	4	31218	29514	**5.46**	505	487
	5	30792	29146	**5.35**	552	532
			average	**4.72**		

Table 4.2: Performance for wire minimization with the TSV constraint

circuit (#FF)	#layer	stage 2a (μm)	stage 2b (μm)	red. (%)	initial #TSV	target #TSV / final #TSV
s1423(74)	3	1360	1336	**1.76**	21	20/19
	4	1564	1466	**6.27**	27	20/19
	5	1302	1222	**6.14**	21	20/17
s5378(179)	3	5280	3484	**34.02**	44	20/20
	4	3788	3256	**14.04**	39	20/19
	5	3252	3054	**6.09**	28	20/20
s9234(211)	3	8916	5088	**42.93**	71	20/19
	4	8698	5428	**37.59**	81	20/19
	5	5044	4078	**19.15**	50	20/20
s15850(597)	3	17314	12764	**26.28**	181	100/99
	4	24606	12778	**48.07**	239	100/99
	5	14774	10820	**26.76**	153	100/99
s13207(669)	3	18912	12750	**32.58**	185	100/99
	4	14354	11134	**22.43**	139	100/97
	5	22394	12812	**42.79**	247	100/99
s38584(1452)	3	51086	32868	**35.66**	475	200/199
	4	68376	32804	**52.02**	575	200/199
	5	88704	33014	**62.78**	596	200/200
s38417(1636)	3	58294	33166	**43.11**	445	200/199
	4	100906	34516	**65.79**	592	200/200
	5	79730	32972	**58.65**	536	200/200
s35932(1728)	3	72208	34826	**51.77**	628	200/200
	4	70540	32370	**54.11**	505	200/199
	5	79288	33002	**58.38**	552	200/200
			average	**35.38**		

3D Relaxation and the local refinement *3D Planarization* without adding the TSV cost, respectively. Both solutions fulfill the related TSV constraint. The fifth column reports the reduction rate from the constraint solving to local refinement during the second stage. This case has the same initial solution with that without imposing TSV constraint. Differently, the local search first derives the solution under the limitation of TSV, and then finds the local optimized solution. The sixth column shows the TSV usage in the initial ordering; the seventh column reports the TSV usage in the wire-minimization ordering and the limitation of TSV.

Table 4.2 shows a good reduction of the total TSV cost due to the loose TSV constraint in the big circuits, such as s38584, s38417, and s35932. At the same time, the 4-layer s38417 has the best optimization rate 65.79% from the solution after constraint solving due to the huge change in the initial solution. For all circuits, the local refinement technique achieves an average of 35.38% on the wire cost reduction but consumes more time for iterations during the stage-2 algorithm.

Table 4.3 shows the results of comparing the total stitching wire cost without imposing any TSV constraint for multi-layer circuits. The third and fourth columns show the total wire cost in μm after performing the GA-based method and the proposed algorithm, respectively; the fifth column shows that improvement ratio of our approach to the GA-based approach; the sixth and seventh columns report runtime in seconds for our algorithm and GA-based method, respectively, and the last column shows the speed-up of our algorithm to the GA-based approach. Therefore, the runtime of the proposed algorithm is proportional to the number of iterations used to perform the local refinement technique and the circuit size.

Although the proposed algorithm results in the slightly worse total cost on the small circuit s1423 and s9234, it can have the comparable or even better results than the GA-based approach on all other bigger circuits. Moreover, the proposed algorithm can compute results of the total wire cost in less than one second and runs at least 2-order faster than the GA-based approach.

Table 4.4 reports the results of comparing the wire cost with the constraint on the number of TSVs in use for all 3D circuits. Each column has the same meaning as those in Table 4.3 except that the second column specifies the maximum number of TSVs that can be

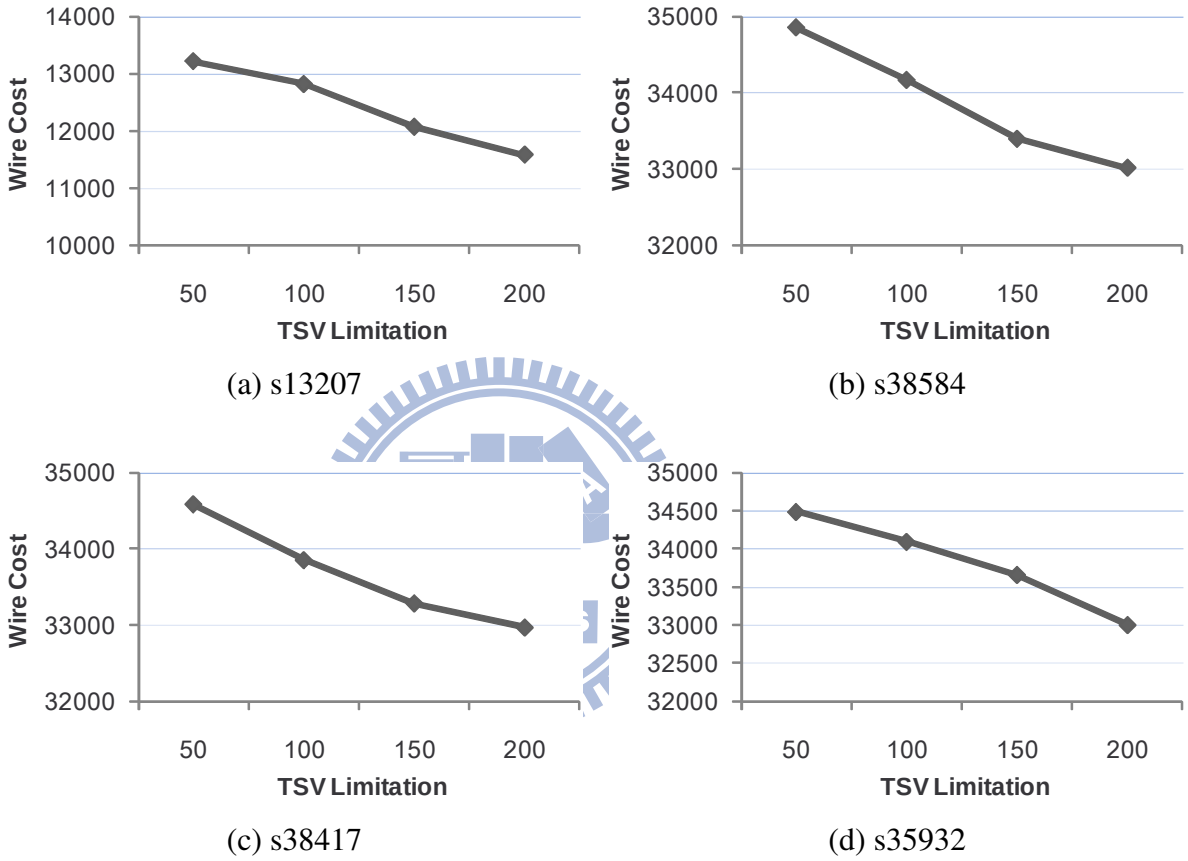
Table 4.3: Comparisons of wire cost and runtime for wire minimization without the TSV constraint

circuit (#FF)	#layer	len-GA (μm)	len-FP (μm)	over. (%)	time-GA (sec)	time-FP (sec)	speedup (X)
s1423(74)	3	1226	1324	**7.99**	41.5	<0.01	>**40000**
	4	1284	1386	**7.94**	62.9	<0.01	>**60000**
	5	1162	1190	**2.41**	62.3	<0.01	>**60000**
s5378(179)	3	3388	3246	**-4.19**	252.2	0.06	**4203**
	4	3270	3190	**-2.45**	285.4	0.02	**14270**
	5	3050	2990	**-1.97**	289.7	0.04	**7242**
s9234(211)	3	4442	4582	**3.15**	479.7	0.07	**6853**
	4	4470	4598	**2.86**	457.9	0.07	**6542**
	5	4020	3944	**-1.89**	427.2	0.06	**7120**
s15850(597)	3	12168	11714	**-3.73**	3248.9	1.8	**1815**
	4	11746	11500	**-2.09**	4330.1	1.6	**2776**
	5	10658	10360	**-2.80**	5211.7	1.6	**3237**
s13207(669)	3	12276	11972	**-2.48**	5205.6	2.4	**2151**
	4	11226	10928	**-2.65**	8721.8	2.1	**4173**
	5	11862	11200	**-5.58**	7471.2	1.8	**4151**
s38584(1452)	3	30432	29344	**-3.58**	19467.8	27.2	**716**
	4	30266	28792	**-4.87**	24305.8	26.9	**904**
	5	30112	28842	**-4.22**	25740.0	24.9	**1033**
s38417(1636)	3	32286	30564	**-5.33**	54083.5	29.9	**1812**
	4	32320	30072	**-6.96**	51510.6	30.6	**1681**
	5	31082	28798	**-7.35**	69316.5	30.5	**2273**
s35932(1728)	3	33010	30698	**-7.00**	68546.3	37.7	**1820**
	4	31454	29514	**-6.17**	98843.8	39.4	**2507**
	5	31088	29146	**-6.25**	107629.0	45.4	**2370**

Table 4.4: Comparisons of wire cost and runtime for wire minimization with the TSV constraint

circuit (#FF)	TSV const.	#layer	len-GA (μm)	len-FP (μm)	over. (%)	time-GA (sec)	time-FP (sec)	speedup (X)
s1423 (74)	20	3	1272	1336	**5.03**	34.8	<0.01	>**30000**
		4	1378	1466	**6.39**	51.0	0.01	**5096**
		5	1152	1222	**6.08**	38.8	0.01	**3877**
s5378 (179)	20	3	3580	3484	**-2.68**	245.0	0.16	**1531**
		4	3298	3256	**-1.27**	237.1	0.11	**2155**
		5	3138	3054	**-2.68**	173.5	0.09	**1928**
s9234 (211)	20	3	5310	5088	**-4.18**	341.3	0.4	**922**
		4	5182	5428	**4.75**	319.0	0.4	**725**
		5	4166	4078	**-2.11**	307.1	0.3	**1228**
s15850 (597)	100	3	12636	12764	**1.01**	2961.1	5.8	**514**
		4	12180	12778	**4.91**	3867.6	9.3	**415**
		5	11094	10820	**-2.47**	4577.4	4.3	**1067**
s13207 (669)	100	3	12646	12750	**0.82**	4763.1	8.7	**551**
		4	11484	11134	**-3.05**	7295.1	4.1	**1771**
		5	12884	12812	**-0.56**	5300.3	9.3	**572**
s38584 (1452)	200	3	32464	32868	**1.24**	15692.4	109.1	**144**
		4	32906	32804	**-0.31**	20845.2	132.0	**158**
		5	33500	33014	**-1.45**	17467.6	141.6	**123**
s38417 (1636)	200	3	34566	33166	**-4.05**	47988.8	122.0	**393**
		4	34752	34516	**-0.68**	41204.7	182.0	**226**
		5	33744	32972	**-2.29**	55407.5	164.3	**337**
s35932 (1728)	200	3	35748	34826	**-2.58**	62614.4	218.3	**287**
		4	33864	32370	**-4.41**	74544.8	172.2	**433**
		5	33418	33002	**-1.24**	75214.4	186.5	**403**

used. Note that benchmark circuits of different scales are imposed with different numbers of TSVs in use, say from 20 to 200. Similarly, the GA-based approach can obtain good total wire cost on smaller circuits, but not on bigger circuits. Although the runtime used for relaxing the TSV number slows down the overall performance, our algorithm still can run at least 2-order faster than the GA-based approach.



(a) s13207 (b) s38584

(c) s38417 (d) s35932

Figure 4.2: TSV impact on the stitching-wire cost for four circuits partitioned into 5 layers

Figure 4.2 studies the TSV impact on the stitching wire cost for four circuits s13207, s38584, s38417, and s35932 partitioned into 5 layers. The x-axis represents the different limitations of TSVs in use, which ranges from 50 to 200; the y-axis represents the stitching wire cost in μm obtained from the proposed algorithm. Four figures illustrated as Figure 4.2 have the same trend that the curve line has the negative slope. Therefore, Figure 4.2 shows that the shorter total wire cost, the more the TSV cost for multi-layer circuits.

Table 4.5: Performance for power reduction without the TSV constraint

circuit(#FF)	stage 1	stage 2	red. (%)	initial / final peak trans.
s1423(74)	9.22E+04	9.15E+04	**0.77**	14/16
s5378(179)	1.23E+06	1.22E+06	**1.08**	57/56
s9234(211)	1.86E+06	1.84E+06	**0.80**	47/47
s15850(597)	1.32E+07	1.31E+07	**0.53**	32/32
s13207(669)	1.60E+07	1.59E+07	**0.78**	48/48
s38584(1452)	1.05E+08	1.03E+08	**1.04**	63/42
s38417(1636)	7.97E+07	7.94E+07	**0.47**	39/47
s35932(1728)	8.83E+06	8.76E+06	**0.76**	4/8
		average	**0.78**	

Moreover, we get the wire-cost reduction rates 12.38%, 5.28%, 4.66%, and 4.31% from 50-TSV to 200-TSV circuits s13207, s38584, s38417, and s35932. The data show that the circuit has more scan cells, the reduction rate is lower, which shows that the reduction rate is bigger when the TSV number is closer to the TSV number used in the optimized solution.

4.2 Reducing Scan-induced Power

Table 4.5 shows the performance of our algorithm without the limitation of TSVs for all circuits. The first column also shows the same information in the wire-minimization part, but we don't show the multi-layer information since the same total weighted transitions are obtained from the multi-layer circuits without the TSV constraints. The second and third columns show the total weighted transition of the initial ordering and the new ordering, respectively; the fourth column shows the reduction rate of total weighted transition from the initial ordering to the new one; the fifth and sixth columns show the peak transition in the initial ordering and the reordering after performing the local refinement technique *3D Planarization*, respectively.

From this table, the technique *3D Planarization* reports a small reduction rate of 0.78%

Table 4.6: Performance for power reduction with the TSV constraint

circuit (#FF)	#layer	stage 2a	stage 2b	red. (%)	initial #TSV	target #TSV / final #TSV
s1423(74)	3	1.00E+05	9.54E+04	**5.02**	62	20/20
	4	1.08E+05	9.77E+04	**9.93**	80	20/20
	5	1.09E+05	9.88E+04	**9.31**	118	20/20
s5378(179)	3	1.38E+06	1.28E+06	**7.26**	122	20/20
	4	1.44E+06	1.28E+06	**11.15**	180	20/20
	5	1.44E+06	1.29E+06	**10.15**	236	20/20
s9234(211)	3	2.26E+06	2.02E+06	**10.63**	200	20/20
	4	2.31E+06	2.02E+06	**12.34**	296	20/20
	5	2.36E+06	2.04E+06	**13.50**	390	20/20
s15850(597)	3	1.46E+07	1.34E+07	**8.25**	358	100/100
	4	1.49E+07	1.37E+07	**8.56**	546	100/100
	5	1.51E+07	1.37E+07	**8.73**	872	100/100
s13207(669)	3	1.84E+07	1.68E+07	**9.13**	450	100/100
	4	1.82E+07	1.66E+07	**9.26**	688	100/100
	5	1.94E+07	1.69E+07	**13.07**	828	100/100
s38584(1452)	3	1.17E+08	1.08E+08	**7.59**	914	200/200
	4	1.20E+08	1.09E+08	**9.64**	1444	200/200
	5	1.23E+08	1.09E+08	**10.98**	1746	200/200
s38417(1636)	3	9.28E+07	8.28E+07	**10.77**	1400	200/200
	4	9.62E+07	8.35E+07	**13.17**	1926	200/200
	5	9.64E+07	8.40E+07	**12.94**	2562	200/200
s35932(1728)	3	1.40E+07	1.01E+07	**27.64**	1430	200/200
	4	1.53E+07	1.04E+07	**31.91**	2230	200/200
	5	1.76E+07	1.10E+07	**37.40**	2818	200/200
			average	**12.85**		

averagely over the initial solution on all circuits since we find a good solution after performing the *Multiple Fragment Heuristic*. Although some peak transitions after performing the refinement technique are worse than that in the initial solution, no big differences occur in all multi-layer circuit.

Table 4.6 shows the performance of our algorithm with the TSV constraint. In this case, the information for the number of layer for all circuits are shown. The third and fourth columns show the total weighted transition after performing the constraint solver *3D Relaxation* and the local refinement *3D Planarization* without adding the TSV cost, respectively. The fifth column also reports the reduction rate from the constraint solving to local refinement during the second stage. The sixth column shows the TSV usage in the initial ordering; the seventh column reports the TSV usage in the power-reduction ordering and the limitation of TSV.

Table 4.6 also shows a good reduction of the total TSV cost due to the loose TSV constraint in the big circuits, especially in s35932. Therefore, the 5-layer s35932 has the best optimization rate 37.40% from the solution after constraint solving due to the huge change in the initial solution. Consequently, the local refinement technique obtains the average reduction rate 12.85% on the total power cost.

Table 4.7 reports the results of comparing the power cost without imposing any TSV constraint for all circuits. The second and third columns show the total weighted transition after performing the GA-based method and the proposed algorithm, respectively; the fourth column also shows that improvement ratio of our approach to the GA-based approach; the fifth column shows the number of peak transition after performing the GA-based method and the proposed algorithm; the sixth and seventh columns report runtime in seconds for our algorithm and GA-based method, respectively, and the last column shows the speed-up of our algorithm to the GA-based approach. Therefore, the runtime is related to the number of iterations used to perform the local refinement technique, look-up table construction, and the circuit size.

In this table, the differences in the peak transition between the GA-based method and the proposed algorithm results are small. Then, the proposed algorithm results have the comparable or even better results than the GA-based approach on all circuits. Moreover, the proposed algorithm can compute results of the total power cost in less than one second

Table 4.7: Comparisons of power cost and runtime for power reduction without the TSV constraint

circuit (#FF)	TWT -GA	TWT -FP	over. (%)	peak trans. -GA / FP	time-GA (sec)	time-FP (sec)	speedup (X)
s1423(74)	9.36E+04	9.15E+04	**-2.24**	27/16	68.8	0.08	**860**
s5378(179)	1.22E+06	1.22E+06	**0.16**	60/56	474.0	0.8	**564**
s9234(211)	1.83E+06	1.84E+06	**0.60**	52/47	777.9	1.2	**671**
s15850(597)	1.34E+07	1.31E+07	**-2.47**	48/32	10313.7	8.4	**1223**
s13207(669)	1.60E+07	1.59E+07	**-0.99**	54/48	13234.7	12.7	**1039**
s38584(1452)	1.12E+08	1.03E+08	**-7.36**	61/42	91982.7	85.0	**1082**
s38417(1636)	8.63E+07	7.94E+07	**-8.00**	43/47	128165.0	117.5	**1091**
s35932(1728)	9.56E+06	8.76E+06	**-8.36**	3/8	199112.0	70.9	**2810**

Table 4.8: Comparisons of power cost and runtime for power reduction with the TSV constraint

circuit (#FF)	TSV constr.	#layer	TWT -GA	TWT -FP	over. (%)	peak trans. -GA / FP	time-GA (sec)	time-FP (sec)	speedup (X)
s38584(1452)	200	3	1.16E+08	1.08E+08	**-7.00**	66/69	65920.9	349.3	**189**
		4	1.17E+08	1.09E+08	**-6.68**	70/61	56979.5	410.4	**139**
		5	1.16E+08	1.09E+08	**-5.99**	57/72	62367.5	444.9	**140**
s38417(1636)	200	3	8.86E+07	8.28E+07	**-6.48**	42/44	111766.0	486.5	**230**
		4	8.76E+07	8.35E+07	**-4.61**	44/38	106233.0	565.5	**188**
		5	8.94E+07	8.40E+07	**-6.12**	36/40	92206.0	602.6	**153**
s35932(1728)	200	3	1.05E+07	1.01E+07	**-3.49**	7/5	188079.0	516.7	**364**
		4	1.08E+07	1.04E+07	**-3.70**	6/5	162181.0	604.2	**268**
		5	1.11E+07	1.10E+07	**-1.00**	7/6	145956.0	693.3	**211**

and still runs at least 2-order faster than the GA-based approach.

Table 4.8 reports the results of comparing the power cost with the constraint on the number of TSVs in use for three big circuits s38584, s38417, and s35932. Each column has the same meaning as those in Table 4.7 except that the second column specifies the limitation of TSV in use. Similarly, different circuits are imposed with different numbers of TSVs in use, ranges from 20 to 200. Since used for relaxing the TSV number, the runtime is slower than the power-reduction problem without the TSV constraints. Consequently, our algorithm still can run at least 2-order faster than the GA-based approach.

Chapter 5

Conclusion

The scan-chain reordering problem is reviewed and modified for TSV-constrained 3D IC designs. A fast scan-chain reordering algorithm is developed and consists of two stages: (1) initial solution computation and (2) local refinement & constraint solving. To avoid the high complexity of *3D optimization*, we convert such problem into a TSP problem and use one of the state-of-the-art algorithms named *Multiple Fragment Heuristic* combined with a dynamic closest-pair data structure *FastPair* to derive a good initial solution quickly. Then two local refinement techniques *3D Planarization* and *3D Relaxation* are also proposed to minimize the wire cost or reduce the scan-induced power and to relax the use of TSVs, respectively. Experimental results show that the proposed algorithm can achieve comparable ($<$3%), or even better performance than that from a GA-based approach and runs at least 2-order faster with (without) considering the TSV constraint.

Bibliography

[1] C. Ababei, Y. Feng, B. Goplen, H. Mogal, T. Zhang, K. Bazargan, and S. Sapatnekar. Placement and routing in 3d integrated circuits. *Design Test of Computers, IEEE*, 22(6):520 – 531, nov.-dec. 2005.

[2] J. L. Bentley. Experiments on traveling salesman heuristics. In *Proc. Symp. on Discrete Algorithms (SODA)*, pages 91–99, 1990.

[3] Y. Bonhomme, P. Girard, L. Guiller, C. Landrault, S. Pravossoudovitch, and A. Virazel. Design of routing-constrained low power scan chains. In *Proc. Design, Automation and Test in Europe Conf. (DATE)*, page 10062, 2004.

[4] C.A. Bower, D. Malta, D. Temple, J.E. Robinson, P.R. Coffinan, M.R. Skokan, and T.B. Welch. High density vertical interconnects for 3-d integration of silicon integrated circuits. In *Proc. Electronic Components and Technology Conf. (ECTC)*, page 5 pp., 2006.

[5] W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer, and P. D. Franzon. Demystifying 3d ics: the pros and cons of going vertical. *Design Test of Computers, IEEE*, 22(6):498 – 510, nov.-dec. 2005.

[6] D. Eppstein. Fast hierarchical clustering and other applications of dynamic closest pairs. *J. Exp. Algorithmics*, 5:1, 2000.

[7] C. Giri, S.K. Roy, B. Banerjee, and H. Rahaman. Scan chain design targeting dual power and delay optimization for 3d integrated circuit. In *Proc. Advances in Computing, Control, Telecommunication Technologies Conf. (ACT)*, pages 845 –849, 2009.

[8] P. Gupta, A.B. Kahng, and S. Mantik. Routing-aware scan chain ordering. In *Proc. Asia and South Pacific Conf. on Design Automation (ASP-DAC)*, pages 857 – 862, 2003.

[9] M. Hirech, J. Beausang, and X. Gu. A new approach to scan chain reordering using physical design information. In *Proc. Int'l Test Conf. (ITC)*, pages 348 –355, 1998.

[10] X.-L. Huang and J.-l. Huang. A routability constrained scan chain ordering technique for test power reduction. In *Proc. Asia and South Pacific Conf. on Design Automation (ASP-DAC)*, page 5 pp., 2006.

[11] P. Jain, T.-H. Kim, J. Keane, and C.-H. Kim. A multi-story power delivery technique for 3d integrated circuits. In *Proc. Int'l Symp. on Low Power Electronics and Design (ISLPED)*, pages 57–62, 2008.

[12] J. W. Joyner, P. Zarkesh-Ha, J. A. Davis, and J. D. Meindl. A three-dimensional stochastic wire-length distribution for variable separation of strata. In *Proc. Int'l Interconnect Technology Conf. (IITC)*, pages 126 –128, 2000.

[13] J.W. Joyner and J.D. Meindl. Opportunities for reduced power dissipation using three-dimensional integration. In *Proc. Int'l Interconnect Technology Conf. (IITC)*, pages 148 – 150, 2002.

[14] J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, V. Narayanan, M. S. Yousif, and C. R. Das. A novel dimensionally-decomposed router for on-chip communication in 3d architectures. In *Proc. Int'l Symp. on Computer Architecture (ISCA)*, pages 138–149, 2007.

[15] S. Makar. A layout-based approach for ordering scan chain flip-flops. In *Proc. Int'l Test Conf. (ITC)*, pages 341 –347, 1998.

[16] B.B. Paul, R. Mukhopadhyay, and I.S. Gupta. Genetic algorithm based scan chain optimization and test power reduction using physical information. In *Proc. TENCON Conf.*, pages 1 –4, 2006.

[17] V. F. Pavlidis and E.G. Friedman. *Three-dimensional integrated circuit design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008.

[18] R. Sankaralingam, R.R. Oruganti, and N.A. Touba. Static compaction techniques to control scan vector power dissipation. In *Proc. VLSI Test Symp. (VTS)*, pages 35 –40, 2000.

[19] M. Sunohara, T. Tokunaga, T. Kurihara, and M. Higashi. Silicon interposer with tsvs (through silicon vias) and fine multilayer wiring. In *Proc. Electronic Components and Technology Conf. (ECTC)*, pages 847 –852, 2008.

[20] A.W. Topol, D.C. La Tulipe, L. Shi, S.M. Alam, D.J. Frank, S.E. Steen, J. Vichiconti, D. Posillico, M. Cobb, S. Medd, J. Patel, S. Goma, D. DiMilia, M.T. Robson, E. Duch, M. Farinelli, C. Wang, R.A. Conti, D.M. Canaperi, L. Deligianni, A. Kumar, K.T. Kwietniak, C. D'Emic, J. Ott, A.M. Young, K.W. Guarini, and M. Ieong. Enabling soi-based assembly technology for three-dimensional (3d) integrated circuits (ics). In *Proc. Electron Devices Meeting Conf. (IEDM)*, pages 352 –355, 2005.

[21] X. Wu, P. Falkenstern, K. Chakrabarty, and Y. Xie. Scan-chain design and optimization for three-dimensional integrated circuits. *J. Emerg. Technol. Comput. Syst.*, 5(2):1–26, 2009.

[22] Y.-Z. Wu and M.C.-T. Chao. Scan-chain reordering for minimizing scan-shift power based on non-specified test cubes. In *Proc. VLSI Test Symp. (VTS)*, pages 147 –154, 2008.

[23] Y. Xie and Y. Ma. Design space exploration for 3d integrated circuits. In *Proc. Int'l Conf. on Solid-State and Integrated-Circuit Technology (ICSICT)*, pages 2317 –2320, 2008.