# Pre-bond TSV Test Optimization and Stacking Yield Improvement for 3D ICs

by

Bei Zhang

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
December 13, 2014

Keywords: 3D IC, compound yield, cost analysis, pre-bond through silicon via (TSV) probing, sector symmetry and cut, test sessions, wafer-on-wafer stacking

Copyright 2014 by Bei Zhang

Approved by

Vishwani D. Agrawal, Chair, James J. Danaher Professor of Electrical and Computer Engineering
Adit Singh, James B. Davis Professor of Electrical and Computer Engineering
Victor P. Nelson, Professor of Electrical and Computer Engineering

Abstract

Through silicon via (TSV) based three-dimensional IC (3D IC) exhibits various advantages over traditional two-dimensional IC (2D IC), including heterogeneous integration, reduced delay and power dissipation, compact device dimension, etc. However, to commercialize 3D IC products, still, many challenges exist. In this dissertation, we focus on conquering two of these challenges. The first challenge is to reduce pre-bond faulty TSV diagnosis time. The second challenge is to improve the compound yield and reduce cost of wafer-on-wafer stacked 3D ICs. Novel ideas are proposed and are demonstrated to be good solutions for these two challenges.

Pre-bond TSV testing and defect identification is extremely important for yield assurance of 3D stacked devices. In this dissertation, we proposed a three-step optimization method named "SOS3" to greatly reduce pre-bond TSV test time without losing the capability of identifying certain number of faulty TSVs. The three steps of optimization are as follows. First, an ILP (integer linear programming) model is proposed to generate near-optimal set of test sessions for pre-bond TSV diagnosis. The sessions generated by our ILP model identify defective TSVs in a TSV network with the same capability as that of other available heuristic methods, but with consistently reduced test time. Second, an iterative greedy procedure to sort the order of test sessions is proposed. Third, a fast TSV identification algorithm is proposed to actually diagnoses the faulty TSVs based on given test sessions. Extensive experiments are done for various TSV networks and the results show SOS3 as a framework greatly speeds up the pre-bond TSV test. SOS3 provides useful known-good-die information for 3D die-on-die, die-on-wafer, and wafer-on-wafer stacking.

Wafer-on-wafer stacking offers practical advantages over die-on-die and die-on-wafer stacking in 3D IC fabrication, but it suffers from low compound yield. To improve the

yield, a novel manipulation scheme of wafer named $n$-sector symmetry and cut (SSC$n$) is also proposed in this dissertation. In this method, wafers with rotational symmetry are cut into $n$ identical sectors, where $n$ is a suitably chosen integer. The sectors are then used to replenish repositories. The SSC$n$ method is combined with best-pair matching algorithm for compound yield evaluation. Simulation of wafers with nine different defect distributions shows that previously known plain rotation of wafers offers only a trivial benefits in yield. A cut number four is optimal for most of the defect models. The SSC4 provides significantly higher yield and the advantage becomes more obvious with increase of the repository size and the number of stacked layers. Cost model of SSC$n$ is analyzed and the cost-effectiveness of SSC4 is established. Observations made are: 1) Cost benefits of SSC4 become larger as the manufacturing overhead of SSC4 become smaller, 2) cost improvement of SSC4 over conventional *basic* method increases as the number of stacked layers increases and 3) for most defect models, SSC4 largely reduces the cost even when manufacturing overhead of SSC4 is considered to be very large.

# Acknowledgments

There are many people to whom I want to say a thousand thanks. First, I would like to thank my advisor Professor Vishwani D. Agrawal. When I got stucked in my research, he is always there for guidance and more importantly, encouragement. No matter how simple the question is, he always answers with extreme patience and points me to many useful references. He always puts himself in the students' position, and give them lots of care not only in academic research but also in their ordinary lives.

I would also like to thank Dr. Adit Singh and Dr. Victor Nelson for serving as my committee members. Dr Singh's VLSI Testing course serves as the starting point of my PhD research. From Dr. Nelson's Computer-Aided Design Course, I learned how to use various useful tools like DFTAdvisor, Fastscan, IC station, etc. I would like to thank Prof. Xiao Qin for agreeing to be my external reader.

I would also like to thank my colleagues in my department, especially, Chao Han, Baohu Li, Guangjie Huang, Yingsong Huang, Jiao Yu, Hua Mu, and Xing Wu, Tiantian Xie, etc. It is them who make my life in Auburn more joyful.

Above all, I would like to thank my parents for their constant support. Without them, I wouldn't even have the chance to study in the US.

Finally, I must acknowledge that the research presented in this dissertation, is supported in part by the National Science Foundation Grants CCF-1116213, IIP-0738088 and IIP-1266036.

Table of Contents

Abstract . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . ii
Acknowledgments . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . iv
List of Figures . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . viii
List of Tables . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . xi
1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
1.1 Various 3D Technologies . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
1.2 Fabrication of TSV-based 3D Stacked ICs . . . . . . . . . . . . . . . . . . . 3
1.2.1 TSV Fabrication Process, Characteristics, and Possible Defects . . . . 3
1.2.2 3D IC Stacking Orientations and Stacking Scales . . . . . . . . . . . 7
1.3 3D IC Testing Issues and Emerging Solutions . . . . . . . . . . . . . . . . . 11
1.3.1 2D Versus 3D Test Moments . . . . . . . . . . . . . . . . . . . . . . 11
1.3.2 Pre-bond TSV Test . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
1.3.3 Emerging Test Standard for 3D ICs . . . . . . . . . . . . . . . . . . 16
1.4 Other Challenges of 3D Stacking Technology . . . . . . . . . . . . . . . . . 19
1.5 Organization of Dissertaion . . . . . . . . . . . . . . . . . . . . . . . . . . 20
2 Test Session Optimization for Pre-bond TSV Probing in 3D Stacked ICs . . . . 22
2.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22
2.2 ILP Model for Test Session Generation with Specified Identification Capability 24
2.3 Experimental Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29
2.4 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 32
3 An Optimal Probing Method of Pre-Bond TSV Fault Identification in 3D Stacked ICs . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34
3.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34

3.2 A Dynamically Optimized TSV Identification Algorithm . . . . . . . . . . . 35
3.3 Experimental Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 36
3.4 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 38
4 SOS3: Three-Step Optimization of Pre-Bond TSV Test for 3D Stacked ICs . . . 39
4.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 39
4.2 Probabilistic Analysis of Number of Faulty TSVs Within a TSV Network . . 39
4.3 An Iterative Greedy Procedure for Test Session Scheduling . . . . . . . . . . 41
4.4 A Three-step Test Time Optimization Simulator . . . . . . . . . . . . . . . . 45
4.5 Experimental Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 45
4.6 Conclusion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 49
5 A Novel Wafer Manipulation Method for Yield Improvement and Cost Reduction of 3D Wafer-on-Wafer Stacked ICs . . . . . . . . . . . . . . . . . . . . . . . 51
5.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 51
5.2 Background and Motivation . . . . . . . . . . . . . . . . . . . . . . . . . 53
5.3 Preliminaries . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56
5.3.1 Defect Distributions on a Wafer . . . . . . . . . . . . . . . . . . . . . 56
5.3.2 Wafers with Rotational Symmetry . . . . . . . . . . . . . . . . . . . . 58
5.3.3 Running Repository Based Best-pair Matching Algorithm . . . . . . . 58
5.3.4 Matching Criteria . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 59
5.3.5 A Hybrid Wafer-on-Wafer Stacking Procedure . . . . . . . . . . . . . 60
5.4 Sector Symmetry and Cut for Yield Improvement . . . . . . . . . . . . . . . 60
5.4.1 Wafers Cut into Sectors . . . . . . . . . . . . . . . . . . . . . . . . . 61
5.4.2 Sector Symmetry and Cut . . . . . . . . . . . . . . . . . . . . . . . . 62
5.4.3 Discussion on the Number of Cuts . . . . . . . . . . . . . . . . . . . . 63
5.4.4 Summmary . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 69
5.5 Experimental Results . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 69
5.5.1 Experimental Setup . . . . . . . . . . . . . . . . . . . . . . . . . . . . 69

5.5.2 Comparison of Various Stacking Procedures for Different Defect Distributions . . . 71
5.5.3 Impact of Number of Stacked Layers on Compound Yield . . . 74
5.5.4 Impact of Production Size on Compound Yield . . . 76
5.6 Cost Effectiveness of Sector Symmetry and Cut Method . . . 78
5.7 Conclusion . . . 84
6 Conclusions and Future Work . . . 85
Bibliography . . . 88
A Impact of Number of Cuts on Final Production Size of Good 3D ICs . . . 95

List of Figures

1.1 TSV process sequence showing etching, CVD oxide, PVD Ti/Cu, Cu electroplating and CMP for 10x100 µm vias [18]. . . . . . . . . . . . . . . . . . . . . . . . . 3

1.2 *RC* circuit model of defect-free blind and open-sleeve TSVs. . . . . . . . . . . . 6

1.3 *RC* model of resistance defective TSV and capacitance defective TSVs. . . . . . 8

1.4 Illustration of different die stacking orientations. . . . . . . . . . . . . . . . . . . . 8

1.5 Illustration of Die-on-Die stacking. . . . . . . . . . . . . . . . . . . . . . . . . . . 9

1.6 Illustration of Die-on-Wafer stacking. . . . . . . . . . . . . . . . . . . . . . . . . 10

1.7 Illustration of Wafer-on-Wafer stacking. . . . . . . . . . . . . . . . . . . . . . . . 10

1.8 Illustration of test moments of 2D and 3D ICs. . . . . . . . . . . . . . . . . . . . 12

1.9 Illustration of pre-bond TSV probing. . . . . . . . . . . . . . . . . . . . . . . . . 14

1.10 Circuit model for pre-bond TSV probing. . . . . . . . . . . . . . . . . . . . . . . 15

1.11 Conceptual illustration of IEEE P1838 standard architecture and wrapper for a 3-die stack with flip chip bumps on the bottom die [41]. . . . . . . . . . . . . . 17

1.12 Various test moments supported by P1838 standard. . . . . . . . . . . . . . . . 19

2.1 ILP model 1 for finding near-optimal test sessions with specified identification capability. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29

2.2 Test time comparison between ILP model 1 and heuristic [39] for a 20-TSV network. 30

2.3 Test time comparison between ILP model 1 and heuristic [39] for resolution constraint $r = 3$. . . . . 31

2.4 Comparison of number of sessions between ILP model 1 and heuristic [39] for resolution constraint $r = 4$. . . . . 32

3.1 A dynamically optimized TSV identification algorithm. . . . . 35

4.1 Pseudo-code for iterative test session sorting. . . . . 44

4.2 Three-step test time optimization simulator. . . . . 46

5.1 Wafer-on-wafer stacking procedures. . . . . 55

5.2 Gray maps showing the yield level distribution on the wafer. . . . . 57

5.3 Wafer maps showing rotational symmetry. . . . . 58

5.4 A conventional wafer cut into four sectors. . . . . 61

5.5 Cutting a rotationally symmetric wafer into identical subwafers. . . . . 62

5.6 Illustration of die loss for cutting the wafer into 6 sectors. . . . . 63

5.7 Two different ways of placing dies on a rotationally symmetric wafer. . . . . 64

5.8 Calculation of $DPW1$ for sector placement method 1. . . . . 65

5.9 Calculation of $DPW2$ for sector placement method 2. . . . . 66

5.10 Calculation of $DPW2$ of 3-cuts for sector placement method 2. . . . . 67

5.11 $DPW1$ and $DPW2$ versus number of cuts for placement methods 1 and 2. . . . . 68

5.12 Process flow of sector symmetry and cut method. . . . . . . . . . . . . . . . . . . 70

5.13 Yield improvement by various stacking procedures for different defect distribution patterns of Figure 5.2. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 72

5.14 Normalized yield for various stacking methods versus number of stacked layers for different defect distribution patterns of Figure 5.2. . . . . . . . . . . . . . . . 75

5.15 Yield reduction for various defect distributions (Figure 5.2) as production size increases. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 77

A.1 Exploring the impact of number $n$ of cuts on final production size of good 3D ICs produced by the sector symmetry and cut (SSC$n$) procedure. . . . . . . . . 96

List of Tables

2.1 Capacitor charging time of parallel TSV test [49]. . . . . . . . . . . . . . . . . . . 23

3.1 Exhaustive and dynamically optimized (Figure 3.1) application of TSV test sessions constructed by ILP model 1. . . . . . . . . . . . . . . . . . . . . . . . . . 37

4.1 Probability of different number of failing TSVs $\phi$ within a 15-TSV network. . . 41

4.2 Expectation of number of tested sessions, defect clustering coefficient $\alpha = 1$, data shows (sessions for SOS2, sessions for SOS3, reduction by SOS3). . . . . . . . . 48

4.3 Expectation of test time ($\mu$s), defect clustering coefficient $\alpha = 1$, data shows (test time for SOS2, test time for SOS3, reduction by SOS3). . . . . . . . . . . . . . 48

5.1 Geometrical parameters for dies per wafer (DPW) calculation. . . . . . . . . . . 64

5.2 Wafer manipulation methods. . . . . . . . . . . . . . . . . . . . . . . . . . . . 70

5.3 Cost improvement percentage for SSC4 over *basic* under various defect distributions (Figure 5.2) and for number of staking layers ($l$) ranging from 2 to 6. . . . 82

# Chapter 1

# Introduction

## 1.1 Various 3D Technologies

Currently, there are several 3D chip integration technologies. Examples are monolithic 3D, system in package (SIP), package on package (POP), and 3D stacked integrated chip (IC) technology. By allowing higher component density, these 3D techniques are mainly favored in applications with small footprint requirements, such as mobile phones, digital cameras, etc.

In monolithic manufacturing, multiple device layers are grown on the same wafer. After the first device layer and the corresponding interconnects are finished, a dielectric layer such as SiO2 is deposited. The isolation layer is then polished to allow the growth of the second device layer. This process is repeated so that multiple layers can grow in a serial manner. Communications between different device layers are provided by vias etched through the dielectric layer. Monolithic 3D IC provides high via density, and possibly smaller mask count. The biggest obstacle to achieve monolithic 3D devices is that the thermal processing of upper silicon layers can disturb the already processed devices and interconnects underneath.

The system in package (SIP) technique is also known as chip stack MCM (multi-chip module). In this technique, multiple chips are stacked vertically and enclosed in a single package. Communications between internal dies are provided by wire bonds. Communication to the outside world can be either provided by wires or flip chip bumps. In package on package technique (POP), multiple packaged chips are stacked vertically. The signal routing between packaged chips is provided by a standard interface. Within each packaged chip, wire bond is always used to connect the IO pad on the die to the package solder balls. Both SIP and POP enable heterogeneous system integration, which means that dies in the stack

may have different functions and may even be fabricated by different vendors. Dies within the stack can be optimized according to their own technologies [13, 30, 33]. Both SIP and POP also offer benefit of smaller footprint. However, both techniques are based on wiring interconnects, which is power consuming and incurs large delay.

The current industry trend is in favor of 3D stacked IC technology. To achieve higher levels of integration, multiple dies of active electronic component are stacked vertically in a 3D IC. We call a die within a 3D stacked IC a *layer*. Connections between layers are provided by *through silicon vias* (TSVs) [6, 18, 32, 45]. TSVs are short and reduce the need for long interconnects as required on planar ICs, thus reducing the delay and power consumption [13, 63, 64]. 3D stacked IC also offers heterogeneous integration and smaller device footprint, which is desirable in hand-held devices. Though challenges remain, the 3D stacked IC is a very hot topic in recent ten years. Several experimental chips and commercial chips are emerging successively.

The earliest experimental 3D stacked chip is a 3D version of the Pentium 4 CPU presented by Intel in 2004 [7]. This chip contains two dies stacked face-to-face. By arranging functional blocks manually in these two dies, the 3D version offers 15% performance improvement and also 15% less power consumption than the 2D version. In 2007, Intel introduced the Teraflops research chip [8]. This is an experimental 80-core design with stacked memory. By implementing a TSV-based memory bus, the total bandwidth of the chip reaches 1 TB per second while consuming much less power than traditional I/O approach. The first academic 3D stacked processor was presented in 2008 at the university of Rochester [50]. After that, two more 3D-stacked-IC-based multi-core designs were presented at the International Solid-State Circuits Conference in 2012 [20, 29]. These two chips utilize Tezzazon's FaStack technology and are fabricated by GlobalFoundry with 130 nm process. The above mentioned chips are mostly experimental chips and thus not involved in volume production. In July 23, 2013, Xilinx announced the world's first commercial heterogeneous stacked 3D IC, i.e.

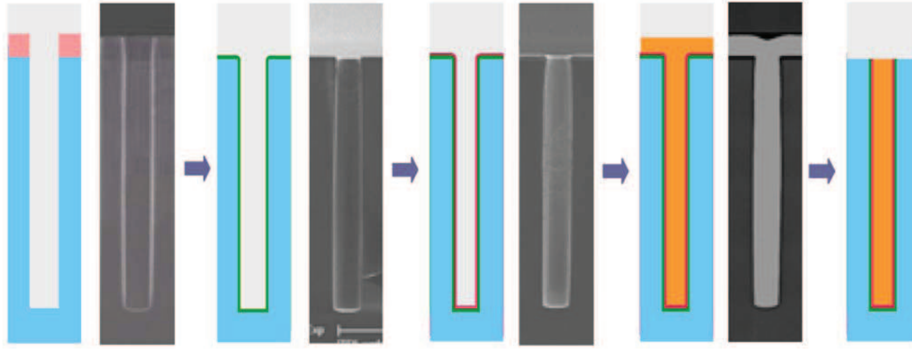


Figure 1.1: TSV process sequence showing etching, CVD oxide, PVD Ti/Cu, Cu electroplating and CMP for 10x100 µm vias [18].

the Virtex-7 H80T FPGA [1]. Two weeks later, Samsung Electronics announced the industry's first 3D stacked NAND flash memory which offers much higher device density than any existing NAND flash technology [2]. As researchers continue pushing the 3D stacking technology forward, more and more commercialized 3D products are expected to emerge in the near future.

This dissertation focuses on 3D stacking technology. For the rest of this work, 3D stacked IC and 3D stacking technology are called 3D IC or 3D technology for simplicity.

## 1.2 Fabrication of TSV-based 3D Stacked ICs

### 1.2.1 TSV Fabrication Process, Characteristics, and Possible Defects

TSVs act as the media to transport power supply and signals among stacks of a 3D stacked IC. Because of its essential role in 3D IC, its fabrication is critical. Figure 1.1 illustrates the five fabrication steps of TSVs [18].

The first step of TSV fabrication is via drilling. In this step, vias are always etched by DRIE (deep reactive ion etching) technique [26]. During etching, slightly tapered side wall (typically desired to be in the range of 83 to 89 degrees [38]) is always preferred to improve the step coverage of the following deposition of $SiO_2$ insulation layer, the barrier layer, and the seed layer. The second step is insulation layer deposition. In this step, either SACVD (semi-atmosphere chemical vapor deposition) or low-temperature PECVD (plasma enhanced

chemical vapor deposition) is used [6]. The third step is barrier and seed layer deposition which is always achieved through PVD (physical vapor deposition) process [6]. The fourth step is via filling. Periodic pulse reverse (PPR) current plating is demonstrated to be the most popular and successful way in via filling [31]. Different materials can be used for via filling, such as copper, tungsten, doped polysilicon [6]. The most popular material is copper due to high conducting property and its compatibility with the conventional interconnect processing [6]. For vias with large size, poly-silicon is preferred because deposition of poly-silicon is much slower than copper which results in a denser deposition that can stand higher stress. The final step of fabrication is CMP (chemical mechanical polishing [26]) which is utilized to thin the wafer so as to expose the buried TSV tips for subsequent die stacking.

When to fabricate TSVs in the complete fabrication flow of a layer? Well, there can be three different schemes.

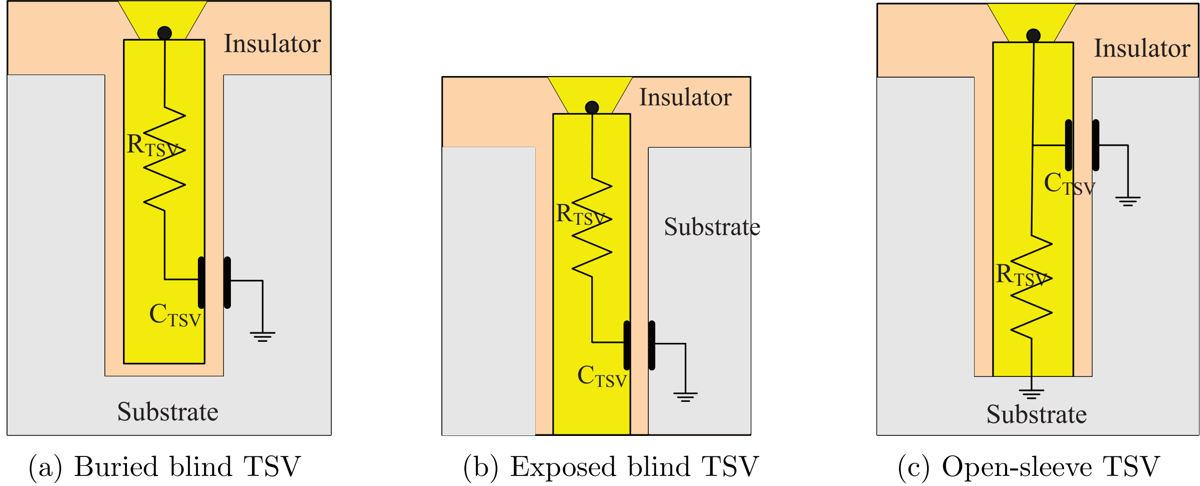
1. "Via first before FEOL" which means TSVs are formed before the FEOL (front-end of the line) [31][44]. The front-end-of-line is the first portion of IC fabrication where the individual devices (transistors, capacitors, resistors, etc.) are patterned in the semi-conductor. FEOL generally covers everything up to (but not including) the deposition of metal interconnect layers. Because there will be further high-temperature (over 1000 degrees) CMOS fabrication processing, the filling material should have the ability to withstand high temperature. From this aspect of concern, the filling material is always chosen as poly-silicon. Refilling poly-silicon requires narrow feature, so this process requires the width of the via to be typically smaller than 5 µm. The wafer needs to be finally thinned to about 150 µm, which means the etched down via needs to have depth with at least 150 µm. This makes the high aspect ratio of the TSV (150:5 or 30:1), and thus brings more challenges in fabrication. The advantage of this scheme is that no barrier and seed layer is needed, and the isolation layer can be easily achieved by traditional oxidation process.

2. "Via first after BEOL" (also called "Via last" in [31]) which means TSVs are formed after the back-end of the line (BEOL) of IC fabrication. The back end of line (BEOL) is the second portion of IC fabrication where the individual devices (transistors, capacitors, resistors, etc.) get interconnected with wiring on the wafer. In this scheme, TSVs are formed after the completion of the CMOS chip but before wafer thinning. Because the CMOS device is completed and the passivation layer is already formed, no further high-thermal requirement process will be needed. We can fill the via with copper, which brings better electrical and thermal properties compared to poly-silicon. Since there are no narrow-feature requirements anymore, the TSV can be made with relatively lower aspect ratio (ranging from 3:1 to 7:1 [44]).

3. "Via last after BEOL" which means TSVs are formed after the IC is fabricated and the wafer is thinned [44]. In this scheme, the wafer is already bonded with adhesive onto the wafer carrier. To protect the bonded wafer and the adhesive, the temperature needs to be specially controlled to be not above 200 degrees. Also during fabrication of TSVs, chemical materials need to be selected such that they wouldn't (or slowly) attack the IC layer.

Generally speaking, fabricating TSVs in different schemes may require different techniques to be used in the five steps of TSV fabrication mentioned above.

Before bonding, only one end of the TSV is connected to active circuitry. If the other end of a TSV is completely insulated by surrounding oxide before thinning or exposed after thinning, then the TSV is called a *blind TSV* [11, 41]. If the other end is surrounded by and shorted to bulk silicon before thinning, then the TSV is called an *open-sleeve TSV* [11, 41]. The characteristics of blind and open-sleeve TSVs are different and thus require different testing strategies. Figure 1.2 shows the $RC$ circuit model of a blind and an open-sleeve TSV, respectively. Note that for simplicity the barrier and seed layer are not shown in these figures. The resistance $R$ of an experimental copper TSV with 2-5 µm diameter and 5 µm



(a) Buried blind TSV (b) Exposed blind TSV (c) Open-sleeve TSV

Figure 1.2: *RC* circuit model of defect-free blind and open-sleeve TSVs.

height is 80-200 μΩ [41].The capacitance $C$ of an experimental copper TSV with 1-10 μm diameter and 30-100 μm height is 10-200 fF [41].

Defects can be introduced during the manufacturing process of TSVs (pre-bond defects) or during the die assembly process (post-bond defects). Thus, testing of TSVs is important as a single irreparable defective TSV may cause the entire stack to fail. TSVs can be tested before die bonding (pre-bond) or after die bonding (post-bond). Post-bond testing targets defects caused by TSV misalignment, mechanical stress, thermal issues, etc. Post-bond TSV test has been extensively studied [25, 34, 36, 46] and TSVs after bonding are basically treated as wires. Post-bond TSV testing can also be easily conducted by employing the developing IEEE P1838 standard [34, 36, 41]. The IEEE P1838 standard is based on existing IEEE 1149.1 standard and IEEE 1500 standard. The purpose of developing this standard is to provide a standard test architecture for 3D IC where desired test access to different layers within the stack can be achieved.

Pre-bond TSV test is much more challenging. Also, the majority of TSV defects can happen before bonding. For example, reference [10] illustrates 11 possible defect types of TSVs where 6 of them occur before bonding. We focus on introducing pre-bond TSV defects for the rest of this section.

According to [10], pre-bond TSV defects include:

1) Micro-void in the TSV pillar, caused by electronic migration and will increase the TSV resistance.

2) Complete breakage in TSV pillar, caused by improper handling of thinned wafers and will result in an open path within the TSV.

3) TSV pillar filling failure, i.e., the TSV pillar is not fully filled. For example, only the left half cylinder is filled by copper. This defect can happen if the seed layer is not deposited uniformly in the third step of TSV fabrication process. A partially filled TSV pillar has larger resistance than the nominal value.

4) Impurity (like foreign particle) between TSV and the micro-bump. This will increase the contact resistance between TSV and its micro-bump.

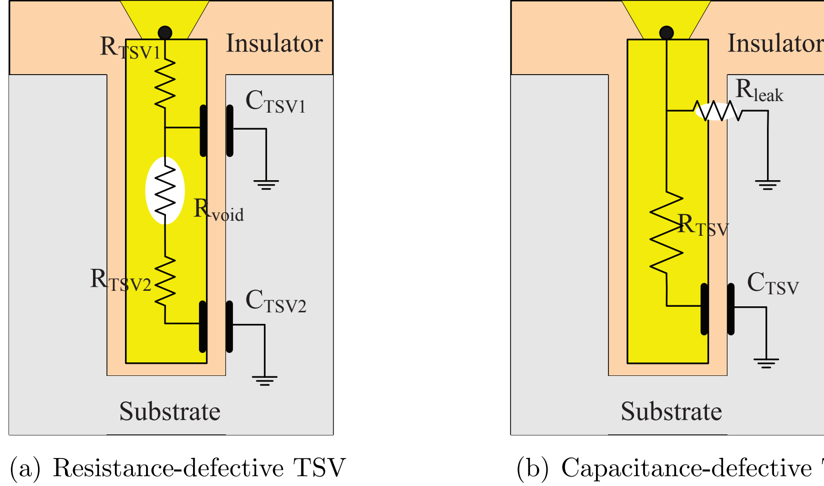
5) TSV pillar is delaminated from the substrate, which may be caused by thermal stress during the fabrication process. This defect will form an open path between the TSV pillar and the microbump.

6) Pinhole within the insulation layer, which forms a conducting path from TSV to the substrate. This leakage path largely increases the capacitance between the TSV and the substrate.

Of these six defects, five of them are resistive in nature, i.e., increasing the TSV resistance. Only the last one, i.e., the pinhole defect results in capacitance changes. Figure 1.3(a) shows the *RC* circuit model for the resistance defective TSVs. Figure 1.3(b) shows the *RC* model of the capacitance defective TSV with pinhole defect [41].

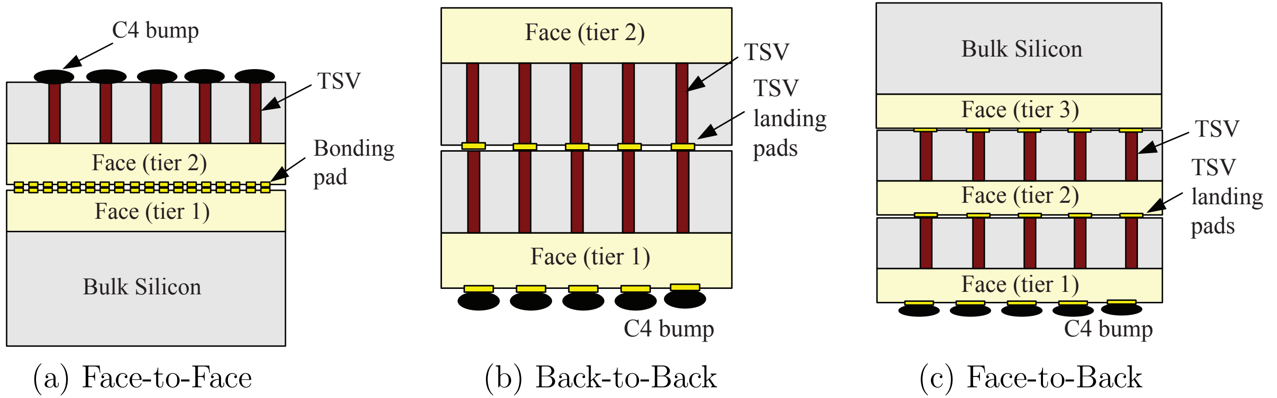
### 1.2.2 3D IC Stacking Orientations and Stacking Scales

To stack a die on another die in 3D IC fabrication, the stacking orientation can be of multiple options. We can stack two dies face-to-face, back-to-back, or face-to-back. Here, *face* refers to the front side of a die containing active circuitry. *Back* refers to the back side




(a) Resistance-defective TSV (b) Capacitance-defective TSV

Figure 1.3: *RC* model of resistance defective TSV and capacitance defective TSVs.




(a) Face-to-Face (b) Back-to-Back (c) Face-to-Back

Figure 1.4: Illustration of different die stacking orientations.

of the die containing bulk silicon. TSVs are buried in the back side and can be exposed by wafer grinding.

Figure 1.4 illustrates these three different stacking orientations. In the figure, we assume the fabricated 3D IC has a flip-chip package, i.e., the communications to the outside world are provided by the flip-chip C4 bumps located at the face side of the bottom die.

Face-to-face stacking offers the highest via density between two dies since the face-side bonding pads can be much smaller than TSV landing pads. Compared to face-to-face stacking, the via density of back-to-back stacking is much smaller. Moreover, instead of fabricating TSVs in only one layer, the back-to-back stacking requires TSVs to be fabricated in both layers. Fabrication of additional TSVs incurs higher cost and increases the risk

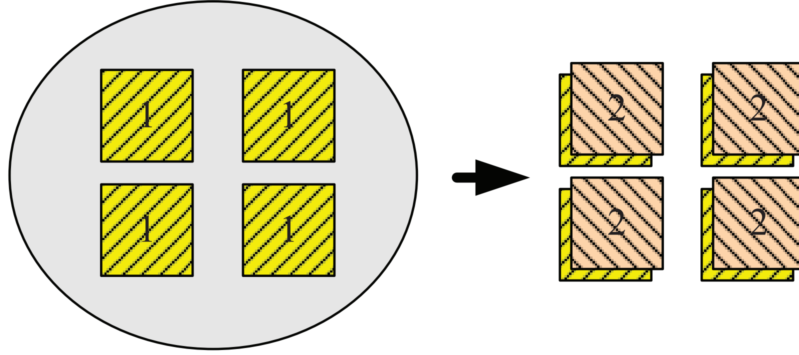


Figure 1.5: Illustration of Die-on-Die stacking.

of having defective TSVs. Thus, back-to-back stacking is not likely to be used in practice. Neither face-to-face nor back-to-back methods scale well to stacks with more than two layers. However, in face-to-back stacking, additional layers can be added to the stack in a repeated way, i.e., the exposed TSV tips on the back side of the lower layer are bonded to dedicated TSV landing pads on the face side of a layer higher in the stack. Thus, any number of layers can be accommodated in face-to-back stacking. Because of this flexibility of multiple layer stacking, face-to-back stacking is most likely to be utilized in practice. In Figure 1.4, we show copper-to-copper direct bonding between two layers. Sometimes microbumps are attached to TSV landing pads for bonding purposes.

To fabricate a 3D IC, the stacking scales can vary. There are three difference stacking scales, namely, die-on-die stacking, die-on-wafer stacking, and wafer-on-wafer stacking. For the later two stacking scales, wafer dicing is required to obtain individual 3D stacks. Figures 1.5, 1.6, and 1.7 illustrate these 3 stacking scales, respectively. In these figures, "1"s and "2"s represent dies on different wafers. For all 3 stacking scales, pre-bond wafer test is necessary to provide known-good-die (KGD) information. The KGD information helps improve the yield by avoiding stacking a bad die on a good die (or good stack) and wasting the good die (or good stack). Without pre-bond test, yield of 3D IC can decrease exponentially with the number of stacked layers, and 3D IC will be difficult to commercialize.

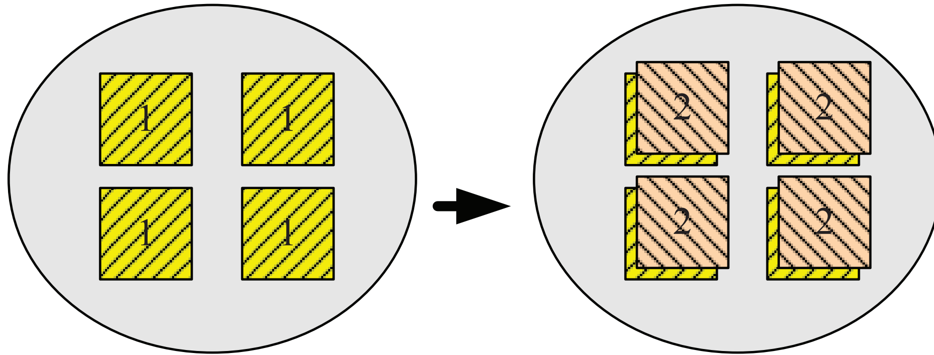



Figure 1.6: Illustration of Die-on-Wafer stacking.

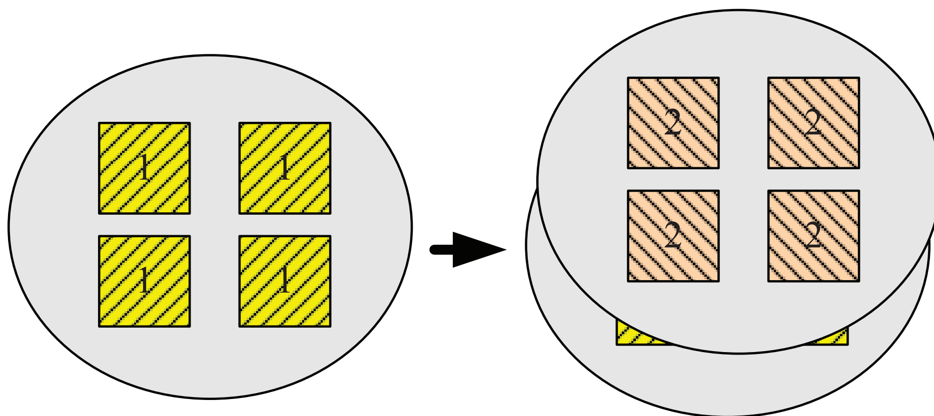



Figure 1.7: Illustration of Wafer-on-Wafer stacking.

Among these three stacking scales, wafer-on-wafer stacking is very attractive. It offers the highest production throughput since each stacking produces a large number of stacked ICs. Wafer-on-wafer stacking also avoids the costly process of pick-and-place required for the other two stacking scales. Other advantages of wafer-on-wafer stacking include smaller die sizes, thinner wafers, and higher TSV density [47, 54, 65]. Although the other two stacking scales offer higher final yield, they are harder to handle, stack, and process, besides being more expensive [17, 51]. The biggest drawback of wafer-on-wafer stacking is that even with KGD information sometimes we cannot avoid stacking a good die on a bad die. Thus, the yield of wafer-on-wafer stacking is a bottle neck. Nonetheless, this weakness can be compensated by developing advanced wafer matching algorithms or advanced wafer manipulation methods. More details of wafer-on-wafer stacking will be illustrated in section 5.

## 1.3 3D IC Testing Issues and Emerging Solutions

### 1.3.1 2D Versus 3D Test Moments

There are two test moments of conventional 2D chips. The first is wafer probing test. The second is final test after chip assembly and packaging. 3D IC testing is more complex with more test moments. Figure 1.8 illustrates the various test moments of 2D and 3D chips, respectively.

Seen from Figure 1.8(b), for a 3D IC containing $n$ layers, there can be $2n$ test moments.

1) $n$ pre-bond test moments. Pre-bond test is necessary as it helps avoid the situation where a single bad die pollutes a complete 3D stack.

2) $n-2$ partial stack test moments (also called mid-bond test moments).

3) One post-bond test moment of the complete stack before assembly and packaging. Post-bond test avoids packaging a bad stack.

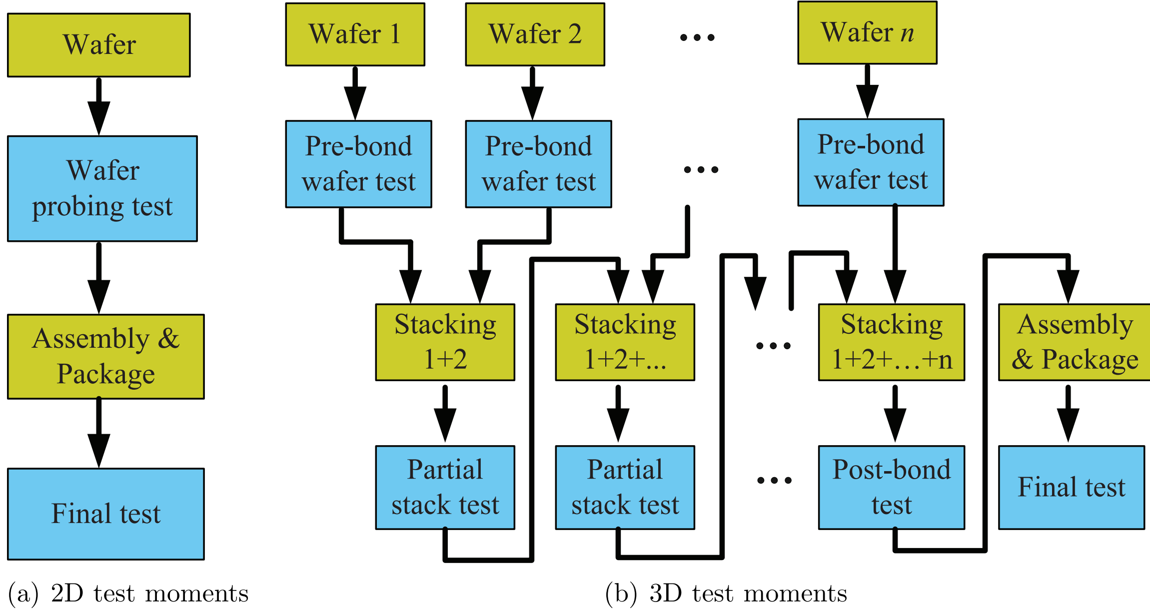



Figure 1.8: Illustration of test moments of 2D and 3D ICs.

4) One final test moment of the packaged 3D IC. Final test assures the quality of 3D chips shipped to customers.

Each of these $2n$ test moments includes two test conditions, i.e. the TSV test and the circuitry test. Thus in total, there are $2 \times 2n = 4n$ test conditions for a $n$-layer 3D IC.

### 1.3.2 Pre-bond TSV Test

As mentioned in section 1.3.1, pre-bond testing contains circuit level testing and TSV testing. Pre-bond circuit level testing can be handled by the IEEE P1838 standard as will be illustrated in the next section. Moreover, the IEEE P1838 standard facilitates TSV test and circuit test after die bonding as well. Among the $4n$ test conditions of a 3D IC, all except the $n$ pre-bond TSV test conditions can be handled by the developing standard. Thus, in this section, we focus on the more challenging problem of pre-bond TSV testing.

Pre-bond TSV test targets defects arising during wafer manufacturing, such as a void within a TSV, a complete break in a TSV, a pinhole creating a leakage path between TSV and substrate, etc. Pre-bond TSV testing is important as it helps identify defective dies early

in the process and avoid situations where one single bad die causes entire 3D stack to be discarded. It is also necessary in providing known good die (KGD) information for die-to-die or die-to-wafer fabrication process. Even for wafer-on-wafer stacking, pre-bond TSV test helps in better wafer matching and thus improves the yield [47, 51, 52, 54, 58, 60, 65, 75, 80].

The pre-bond TSV testing is challenging mainly because before-bonding a TSV is single ended, i.e., one of its ends is not connected to any circuitry. For pre-bond TSV test, we can test on a still-thick wafer. In that case, the TSVs are deeply buried in the wafer substrate without any test access. This requires special per-TSV DFT circuits (e.g., BIST) to test the TSVs with only single-sided access. Several built-in self-test (BIST) techniques have been proposed for buried TSVs, such as, the use of a voltage division circuit to measure the leakage resistance of TSVs and detect pinhole defects [12], or a DRAM and ROM-like test to determine the $RC$ time constant and resistance of blind TSVs and open-sleeve TSVs, respectively [11]. Ring Oscillators have been widely used to characterize the propagation delay of TSVs and thus diagnose possible resistive open or leakage defects [15, 73]. All BIST approaches require dedicated circuits to be added for each individual TSV, and the area overhead is huge since there can be tens of thousands of TSVs on a chip [28, 67]. Moreover, the BIST circuits themselves suffer from process variation, which may render them ineffective. An alternative is to test thinned wafers where TSV tips are exposed. This requires special facilities to probe thinned wafers (about 50 µm thick) without damaging them. However, the relatively large pitch (40 µm) of current probing technology [53, 69] prohibits individual TSV probing with a realistic pitch of 10 µm [28, 43].

A pre-bond TSV probing method has been recently proposed [40] where several TSVs are contacted by a single probe needle defining a *TSV network*. The number of TSVs within a network is typically less than 20 and depends on the relative diameter of the probe needle and the pitch of TSVs [28, 41, 43, 53, 69]. TSV parametric test can be conducted by adding an active driver in the probe needle and forming a charge sharing circuit between single (or multiple) TSV(s) and the probe needle. This probing method offers robustness to process

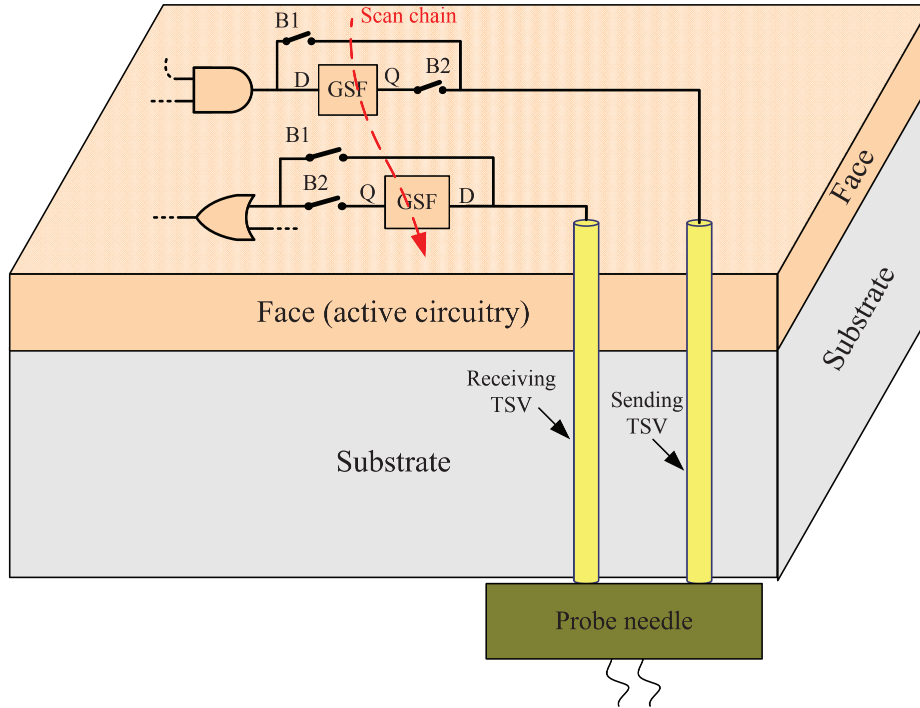



Figure 1.9: Illustration of pre-bond TSV probing.

variation, requires less hardware overhead, provides accurate measurement of TSV resistance, and has many more benefits as explained in the original proposal [39, 40, 41]. One possible concern for current utilization of this technique is that the planarity of TSV tips may not be consistent and could prevent a large probe needle from making good electrical contacts with all TSVs in a network, simultaneously. However, the great benefits offered by the probing method serves as a driving force for both foundry and test equipment manufacturer to work together and find a solution.

Figure 1.9 illustrates how a probe needle contacts two TSVs on the backside of a thinned die. As can be seen, gated scan flip-flops (GSF) are inserted between system logic and TSVs which is in accordance with the developing IEEE P1838 standard [36, 41]. Among the two TSVs, the left side TSV is a receiving TSV which receives a signal from the other die and drives the on-chip logic while the right side one is a sending TSV which is driven by the on-chip logic and sends a signal to the other die. In the normal mode, all GSFs are made transparent by opening B2 and closing B1 switches by a “bypass” signal. Then, the receiving

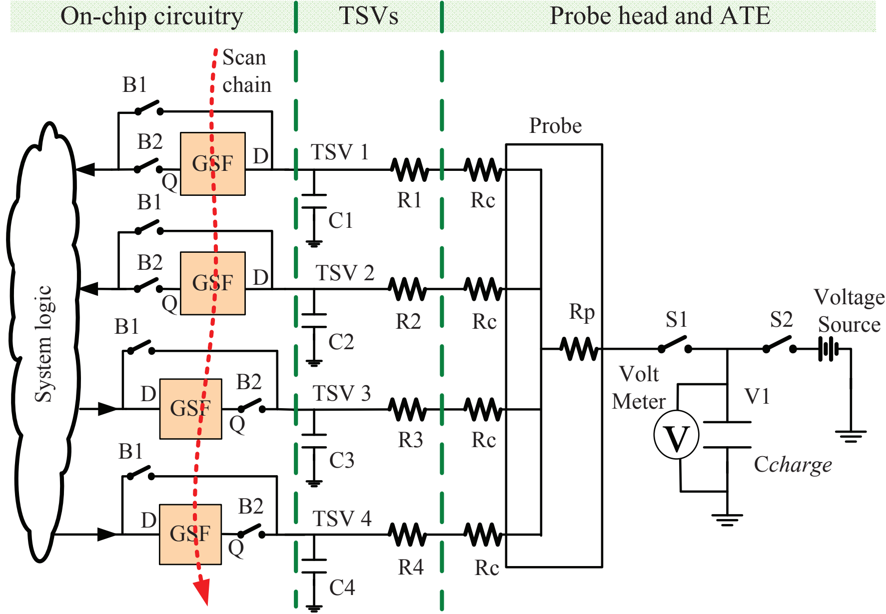



Figure 1.10: Circuit model for pre-bond TSV probing.

TSV in Figure 1.9 feeds a signal from the other die to the input of an OR gate, and the sending TSV is driven by the output of an AND gate on the current die.

Figure 1.10 shows the circuit model of the probing test set up [40, 41] for an example 4-TSV network. TSV $i$ is represented by its resistance $R_i$ and capacitance $C_i$. $R_c$ represents the contact resistance between TSV and the probe. $R_p$ represents probe needle resistance. A gated scan flip-flop is inserted between TSV $i$ and the system logic. All GSFs can be loaded or read out through a boundary scan mechanism. In the normal mode, all GSFs are made transparent. In the pre-bond TSV test mode, all GSFs drive respective TSVs. In Figure 1.10, TSVs 1 and 2 are receiving TSVs and TSVs 3 and 4 are sending TSVs. A GSF in scan mode drives a receiving TSV during pre-bond TSV probing when both B1 and B2 switches are closed. A GSF drives a sending TSV when B1 is opened and B2 is closed.

The above technique facilitates both TSV resistance measurement and capacitance measurement. For example, pre-bond TSV resistance measurement starts by scanning in all GSFs with “1.” $C_{charge}$ and all TSVs are then discharged through the probe needle. By configuring the switches of a GSF, a charge sharing circuit is constructed between that GSF and $C_{charge}$ through its corresponding TSV (either sending or receiving TSV). The charging

rate of $C_{charge}$ is compared to a calibrated curve of a good TSV to determine the resistance of the TSV under test. The discharging and charging process continues for each TSV, which can be completed quickly by sequentially configuring the switches of one GSF at a time so that there is always one GSF driving one TSV. The probe needle remains incontact with the corresponding TSV network, and the charging and discharging process continues until either all TSVs within the network are identified or a certain number of TSVs with resistive defects are pinpointed within the network. All TSV networks are tested in two groups. Networks in a group are tested simultaneously by a probe head containing a large number of needles, each making contact with a single network. Once all contacted networks are tested, the probe head is lifted and repositioned to test the remaining group [40, 41]. Thus, all TSVs on chip can be tested by just two touch downs [40, 41]

### 1.3.3 Emerging Test Standard for 3D ICs

In order to provide a standard test access of 3D stacked ICs, the IEEE P1838 work-group is developing test access architecture and die wrappers specific for 3D IC [34, 36, 41]. This standard assumes the circuit is partitioned into modules and thus enables modular testing of a 3D IC. Modular testing refers to an approach in which the various modules constituting an IC product can be tested separately. The various modules of a 3D IC include each die, its embedded cores, TSVs between two dies, and external pins mounted on printed circuit board (PCB). By enabling modular testing, scheduling of pre-bond test, mid-bond test, post-bond test, and final test can be completely flexible. This flexibility makes it easier to find the most cost-effective test flow for a specific 3D IC manufacturing process.

A conceptual example of a 3-die stack wrapped in accordance to P1838 test standard is shown in Figure 1.11 [41]. Based on the standard, a die wrapper is added to each die in the stack, which is the additional DFT architecture that most characterizes the P1838 standard. The die wrapper can be compatible with the IEEE 1500 standard. Note that for the bottom die, besides the IEEE 1500 die wrapper, there is also an IEEE 1149.1 TAP

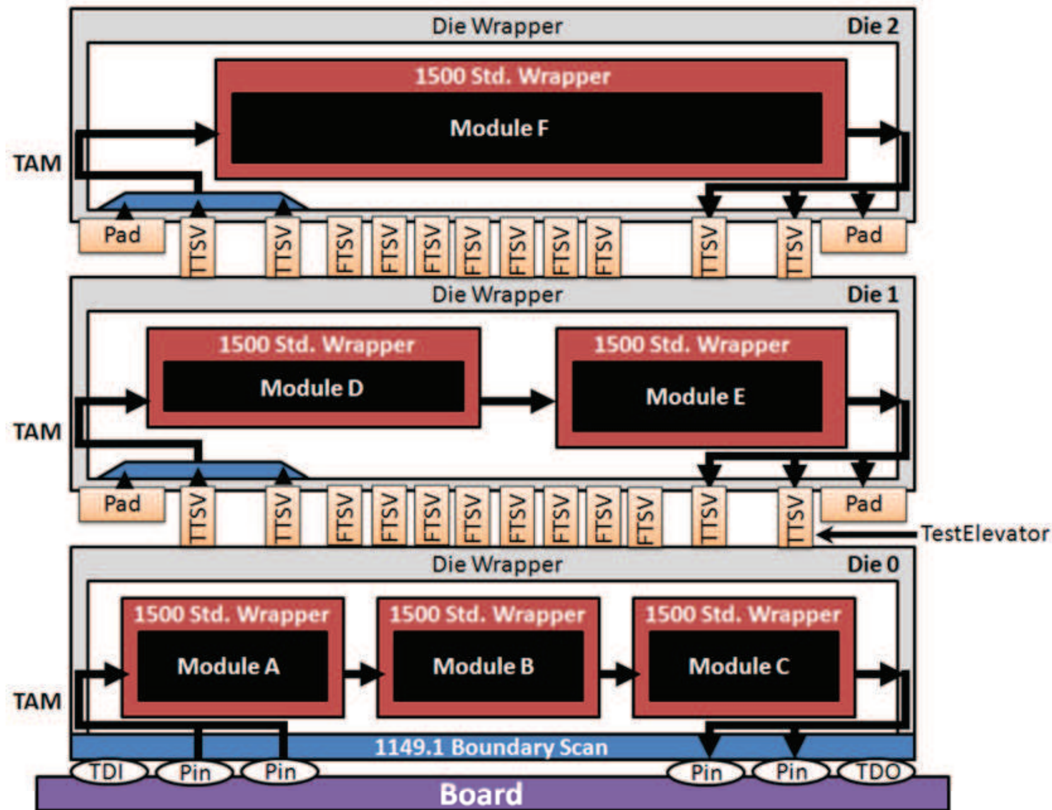



Figure 1.11: Conceptual illustration of IEEE P1838 standard architecture and wrapper for a 3-die stack with flip chip bumps on the bottom die [41].

controller to enable board level testing after the 3D IC is mounted on a PCB. The wrapper instruction registers of all 1500 die wrappers are controlled by the bottom IEEE 1149.1 TAP controller. Each die may contain several embedded cores; these cores are wrapped by IEEE 1500 wrappers as well. There exists a hierarchical relationship between a die wrapper and its embedded core wrappers so that the test mode of a core wrapper can be controlled by the up-level die wrapper [34, 41].

Seen from Figure 1.11, there are large probe pads added on each die except for the bottom die. These pads enables individual contact to a probe needle. The probe pads, together with the die wrapper, enable pre-bond circuit level testing for each non-bottom die. For the bottom die, there are already large flip chip bumps which directly enables pre-bond probing; thus no large pads are needed.

In this paragraph, we introduced how the P1838 standard supports mid-bond testing. If an upper die is bonded to a lower die, e.g. die 2 is bonded to die 1 in Figure 1.11, the probe pads of die 2 are not available anymore. To test die 2, instead of applying test stimuli through die 2 pads, dedicated test TSVs (TTSVs in Figure 1.11, also called test elevators) are used to carry data from die 1 pads all the way up to die 2 and also route test data back to the pads of die 1. The arrows in Figure 1.11 shows the data flow direction throughout the stack. Note that to enable circuitry testing of die 2, the die 1 wrapper needs to be set in *Bypass* mode so that test data can bypass die 1 and be routed to die 2 directly. To test die 1 in this mid-bond phase, the die 1 wrapper needs to be set in *Test Turn* mode, which means the test data will be directly applied to die 1 without moving upward to die 2. The P1838 wrapper also enables mid-bond TSV testing. By setting the die wrappers of both die 1 and die 2 to *Extest* mode, all TTSVs as well as functional TSVs (FTSVs in Figure 1.11) between these 2 dies can also be tested to detect any TSV defects occur during bonding. Specifically, test data is shifted in the wrapper boundary registers (WBR) of both die 1 and die 2, forming a daisy chain. From WBRs, the data is then applied to the TSVs between these two dies and responses are latched in WBRs. The latched responses are then shifted out to the dedicated probing pads on die 1 for faulty TSV diagnosis. To sum up, the TTSVs, probe pads, altogether with the die wrappers enables both mid-bond circuitry testing and TSV testing.

After all dies are stacked, post-bond test is conducted. Post-bond circuitry test and TSV test resembles mid-bond test. For example, to test die 1 in Figure 1.11, the die 0 wrapper needs to be set in *Bypass* mode, and die 1 wrapper needs to be set in *Test Turn* mode. The only difference between post-bond and mid-bond test is that for post-bond testing test data must come from and also be routed down to the flip chip bumps on the bottom die. This is not always the case for mid-bond testing unless the bottom die is also bonded during the mid-bond phase. Final test after packaging is conducted in exactly the same way as post-bond test.

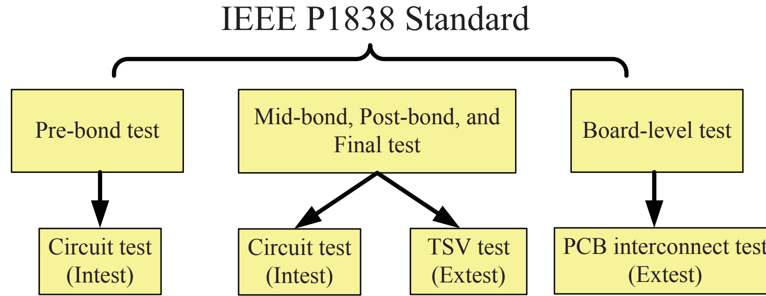



Figure 1.12: Various test moments supported by P1838 standard.

Figure 1.12 shows all possible test moments supported by the P1838 standard. Note again that the P1838 standard cannot directly support pre-bond TSV test. The probing technique mentioned in section 1.3.2 provides a practical way for pre-bond TSV testing. It is compatible with P1838 standard since it utilizes the wrapper boundary registers (WBRs) within the die wrapper. The TSV probing technique in section 1.3.2 served as the basis for our discussions in chapters 2, 3 and 4.

## 1.4 Other Challenges of 3D Stacking Technology

There are still many challenges to commercialize 3D IC technology. For wafer-on-wafer stacking, a wafer needs to be thinned down to be bonded to another wafer. New intra-die defects may occur during grinding, alignment, and bonding processes, which are unique to 3D IC fabrication. Thus new fault models and test patterns need to be created accordingly to identify the new defects [30, 37].

Resolving the thermal issues of 3D IC design is another challenge [13, 30, 37]. A typical 3D IC consists of multiple dies and has smaller footprint than the 2D version. Thus the area for heat radiation becomes smaller for a 3D IC. The heat accumulation inside the stack may severely increase circuit delay and thus outweighs the benefit of small interconnect delay a 3D IC offers.

Test access mechanism design and test scheduling has been extensively studied for 2D SOCs [21, 35, 62]. However, for 3D IC containing core-based dies, new constraints need to be considered during the test architecture design. One important constraint is that the number

of TSVs used for test access (TTSVs) should not exceed a certain value, because TSVs have area overhead and are potential source of defects [42, 68].

As mentioned, for an $n$-tier 3D IC, there are $2n$ test moments. Each test moment can include just TSV test, or just circuit test, or both of them, or none of them. This makes the search space for most cost-effective test flow overwhelming. For a specific manufacturing process, it would be very challenging but important to not only decide which test moments need to be included in the entire test flow but also to decide what should be tested (TSVs, or circuit, or both, or none) for each test moment. Finding an optimal test flow helps greatly reduce the cost of 3D ICs, and progresses are making in this area [23, 58, 59, 65].

Since the number of TSVs on-chip can reach tens of thousands [28, 67], the DFT engineers need to keep in mind that there are probably several faulty TSVs after die bonding. To avoid discarding the stack with faulty TSVs, implementing an efficient TSV redundancy architecture to replace faulty TSVs with redundant good ones is important and also challenging [24, 27, 72, 81]. There are lots of rules to follow when designing the TSV redundancy architecture. First, the total number of redundant TSVs cannot be too large since the keep-out zone of a TSV occupies a large silicon area and is thus costly. Second, the associated circuit overhead for faulty TSVs replacement and signal rerouting should be low within the architecture. Third, the architecture should put higher priority on rescuing signal TSVs instead of power or clock TSVs. TSVs for power delivery and clock distribution always form a connected graph and are more immune to defects [30]. Last but not least, the architecture needs to be able to replace faulty TSVs, which are likely to be clustered in local silicon area [27, 81].

## 1.5 Organization of Dissertaion

The rest of this dissertation is organized as follows. Chapters 2, 3, and 4 focus on minimizing the identification time of faulty TSVs during pre-bond TSV probing. To be more specific, chapter 2 addresses the problem of test session generation for pre-bond TSV probing.

An ILP (integer linear programming) based method for near-optimal test session generation is proposed. Chapter 3 proposes a fast TSV identification algorithm which identifies faulty TSVs based on the generated test sessions. In chapter 4, we first propose an iterative greedy procedure to sort the order of test sessions. Then, this iterative session sorting procedure is combined with the ILP method in chapter 2 and the fast TSV identification algorithm in chapter 3. The combination of these 3 pieces of work form a framework called "3-Step test time Optimization Simulator (SOS3)" which greatly speeds up pre-bond faulty TSV identification process. Chapter 5 introduces a novel wafer manipulation method called "Sector Symmetry and Cut $n$ (SSCn)" which improves the compound yield and reduces the cost of wafer-on-wafer stacking. The work of chapter 2 to chapter 4 serves as the basis of chapter 5 since it helps provide known good die (KGD) information for wafer matching in chapter 5. Finally, chapter 6 concludes this dissertation.

Chapter 2

Test Session Optimization for Pre-bond TSV Probing in 3D Stacked ICs

## 2.1 Introduction

The bulk of TSV defects that can be tested before bonding result in increase of TSV resistance [10], so pinpointing TSV resistive defects is very important. Considering the relative sizes of typical test probes and TSVs, an earlier proposal [40] uses a large probe needle with an active driver to contact multiple TSVs at a time, as illustrated in section 1.3.2.

In section 1.3.2, it is stated that TSV before bonding can be tested individually using the probing technique by configuring one GSF to drive one TSV at a time. However, parallel TSV test can also be conducted by configuring multiple GSFs at a time so that multiple GSFs can drive multiple TSVs simultaneously. In parallel testing, $C_{charge}$ is charged faster and the measurement terminates even quicker [40, 41]. However, there is also a constraint, i.e. the number of TSVs tested in parallel cannot exceed a constant "$r$" due to minimum measurement resolution requirement [40, 41]. This resolution of measurement refers to the minimum change in TSV resistance that can be detected by the probing technique and it is adversely affected by the number of TSVs tested in parallel. We call the TSVs tested in parallel within the same TSV network a *test session*. Based on the probing technique in section 1.3.2 [40, 41], any faulty TSV within a session will cause the session test to fail but we cannot tell which TSV(s) is (are) faulty. On the other hand, a good parallel test implies that all TSVs within the session are fault-free.

SPICE simulation of a TSV probe setup was done using the PTM (predictive technology model [4]) 45nm technology [39, 40, 41, 49], and the capacitance charging time as a function of the number of TSVs tested in parallel is recorded in Table 2.1. Note that *test time* in this work only refers to the time to charge the capacitor $C_{charge}$, as described in [39].

Table 2.1: Capacitor charging time of parallel TSV test [49].

| Number of TSVs tested in parallel ($q$) | Charging time $t(q)$ ($\mu$s) |
|---|---|
| 1 | 0.80 |
| 2 | 0.53 |
| 3 | 0.42 |
| 4 | 0.38 |

The goal of TSV probing is to identify up to a certain number, $m$, of faulty TSVs within a $T$ TSV network, where $m$ is the number of redundant TSVs in the TSV network being tested. If the number of identified faulty TSVs exceeds $m$, then not all faulty TSVs in the network can be repaired, and the chip would be discarded. Otherwise, the on-chip redundant TSVs are sufficient to replace all identified faulty ones. This goal of TSV probing can be achieved by testing one TSV at a time with highest resistance measurement resolution. However such a high resolution is unnecessary. Besides, the single-TSV session requires longer test time, according to Table 2.1. Moreover, instead of identifying all faulty TSVs, only up to $m$ faulty TSVs need to be pinpointed in a network. Significant test time saving is possible if we test TSVs in parallel, without losing the capability of identifying up to $m$ faulty TSVs, and also guarantee that the size of each test session does not exceed the resolution constraint $r$.

Reference [39] proposes a heuristic to generate such test sessions. However, the results are far from being optimal due to the greedy nature of the heuristic. For example, to pinpoint one faulty TSV in a 6-TSV network with minimum resolution constraint of $r = 4$, the heuristic based sessions are {1,2,3,4}, {1,5,6}, {2,5}, {3,6}, {4} [39]. The total time of these sessions is 2.66 $\mu$s according to Table 2.1 which is not optimal. The optimal sessions for this case would be {1,2,3}, {1,4,5}, {2,4,6}, {3,5,6}, which further reduces the test time by 9.7%. This example motivates us to find a way to generate optimal set of test sessions with minimum test time.

In this chapter, we address the problem of test session generation for pre-bond TSV probing, with focus on minimizing the identification time of faulty TSVs. We propose an

integer linear programming (ILP) based method for session generation that can greatly reduce test time compared to the heuristic method [39]. For example, to locate up to 2 faulty TSVs in a 11-TSV network, our method can reduce test time by 38.2% over the heuristic method [39].

We define the following terminology.

1) *TSV network.* A TSV network is formed by all the TSVs that are simultaneously contacted to the same probe needle.

2) *Test session.* During pre-bond TSV probing, TSVs tested in parallel within the same TSV network form a test session.

3) *Session size.* Session size $q$ is defined as the number of TSVs within a session.

4) *Resolution constraint.* Resolution constraint $r$ indicates that the session size should never exceed $r$.

5) *Test time of a session.* Test time of a session in this paper only refers to the charging time of $C_{charge}$. It is related to the session size (refer to Table 2.1).

6) *Maximum number of faulty TSVs to identify within a network.* This number $m$ equals to the number of redundant TSVs in the TSV network being tested.

## 2.2 ILP Model for Test Session Generation with Specified Identification Capability

In this section, we propose an integer linear programming (ILP) model (named ILP model 1) to find a near-optimal set of test sessions. The problem is formulated as follows:

**Problem 2.2.1.** Given the test time $t(q)$ for test session size $q$, $q \in [1, r]$, and the maximum number $m$ of faulty TSVs within a $T$-TSV network, determine a series of test sessions (each of size less than $r$) so that up to $m$ faulty TSVs can be uniquely identified and the total test time is minimized.

A sufficient condition to solve Problem 2.2.1 is as follows.

**Condition 2.2.1.** If each TSV ($TSV_i$) is put in $m$+1 test sessions, $S_1, S_2, \cdots, S_{m+1}$, and the intersection of any two out of these $m$+1 sessions contains only $TSV_i$, i.e., $S_j \bigcap S_k = TSV_i$ for $j \neq k \in [1, m{+}1]$, then up to $m$ faulty TSVs within the network can be uniquely identified.

We refer to those $m+1$ sessions satisfying condition 1 as $m+1$ unique test sessions for $TSV_i$. A unique test session for $TSV_i$ is defined as a session whose intersection with any other session containing $TSV_i$ consists of only $TSV_i$. We prove the sufficiency of Condition 2.2.1 by first stating the following theorem.

**Theorem 2.2.1.** Given that there are no more than $m$ faulty TSVs within a network. If a good TSV, $TSV_i$, belongs to $m + 1$ unique test sessions, then we can find at least one out of these $m + 1$ sessions which consists of only good TSVs.

We prove Theorem 2.2.1 by contraposition.

**Proof of theorem 2.2.1.** Given there are up to $m$ faulty TSVs within a network. Suppose each unique test session for $TSV_i$ contains at least one faulty TSV. Because of the "unique" identity of these $m + 1$ sessions, the faulty TSVs within each session should be different. We conclude that there will be at least $m + 1$ faulty TSVs within the network, which is obviously a contradiction to the given condition that says there are at most $m$ faulty TSVs. In other words, at least one unique session for $TSV_i$ would contain only good TSVs.

Next, we prove Condition 2.2.1 is a sufficient condition for solving Problem 2.2.1.

**Proof of sufficiency.** According to Theorem 2.2.1, Condition 2.2.1 will guarantee that for any good TSV (let's say $TSV_i$) there will be at least one unique session (let's say session $S_j$) consists of only good TSVs. The testing result of $S_j$ would suggest that all TSVs within $S_j$ to be fault-free. Thus, we uniquely identify $TSV_i$ as a good TSV. If all good TSVs are identified, then all defective TSVs are too.

**Proof of non-necessity.** The non-necessity of Condition 2.2.1 is proved by considering a simple example. In sequential testing, all TSVs can be uniquely identified but each TSV resides in only one session.

The ILP model proposed in this section is based on Condition 2.2.1. We first summarize all general constraints for our ILP model as follows:

1. **C1.** Each TSV should reside in at least $m + 1$ sessions.
2. **C2.** The size of a test session ranges anywhere from 0 (empty session) to $r$.
3. **C3.** We suppose any non-empty session is a unique session for any TSV within it.

An upper bound $N_{up}$ on total number of sessions can be calculated as,

$$N_{up} = \left\lceil \frac{t(1)}{t(r)} \cdot T \right\rceil \tag{2.1}$$

where $r$ is resolution constraint, $t(1)$ and $t(r)$ are test times for sessions with sizes 1 and $r$, respectively. If the total number of sessions $N$ is larger than $N_{up}$, then even if all sessions have size $r$ the total test time is still larger than that of sequential testing. This upper bound constrains the maximum number of sessions produced by the ILP model presented next.

A binary variable $x_{ij}$ $(1 \leq i \leq T, 1 \leq j \leq N_{up})$ is defined as follows:

$$x_{ij} = \begin{cases} 1 & \text{if } TSV_i \text{ is assigned to session } S_j \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

From **C1**, we have

$$\sum_{j=1}^{N_{up}} x_{ij} \geq m + 1 \tag{2.3}$$

We define an integer variable $L_j$, which denotes the size of session $S_j$,

$$L_j = \sum_{i=1}^{T} x_{ij} \tag{2.4}$$

From **C2**, we have

$$0 \leq L_j \leq r \tag{2.5}$$

Based on **C3**, if $\exists\ x_{ij} \cdot x_{ik} = 1$ for any $i$ and any $j \neq k \in [1, N_{up}]$, then $\sum_{i=1}^{T} x_{ij} \cdot x_{ik} = 1$. **C3** also implies that the intersection of any two different sessions $S_j$ and $S_k$ should contain no more than a single TSV, thus,

$$\left| S_j \bigcap S_k \right| = \sum_{i=1}^{T} x_{ij} \cdot x_{ik} \leq 1 \tag{2.6}$$

Constraint (2.6) is obviously nonlinear. To linearize it, we further introduce a binary variable $z_{ijk} = x_{ij} \cdot x_{ik}$ and two more linear constraints:

$$x_{ij} + x_{ik} - z_{ijk} \leq 1 \tag{2.7}$$

$$x_{ij} + x_{ik} - 2 \cdot z_{ijk} \geq 0 \tag{2.8}$$

According to these constraints if $x_{ij} = 0$, then $z_{ijk} \leq \frac{x_{ik}}{2}$. Since both $x_{ik}$ and $z_{ijk}$ are binary variables, $z_{ijk}$ has to be zero. If $x_{ij} = 1$, $x_{ik} \leq z_{ijk} \leq 0.5 + 0.5x_{ik}$, and we conclude, $z_{ijk} = x_{ik}$. Thus, constraints (2.7) and (2.8) guarantee $z_{ijk} = x_{ij} \cdot x_{ik}$. With $z_{ijk}$, constraint (2.6) becomes

$$\sum_{i=1}^{T} z_{ijk} \leq 1 \tag{2.9}$$

The objective of the ILP model is to minimize the total test time of all sessions:

$$\text{Minimize} \sum_{j=1}^{N_{up}} t(L_j) \tag{2.10}$$

Both $L_j$ and $t(L_j)$ are variables, we need to linearize the objective function so that any commercial ILP solver can be used. A new binary variable $\delta_{jq}$ ($1 \leq j \leq N_{up}$, $0 \leq q \leq r$) is

introduced:

$$\delta_{jq} = \begin{cases} 1 & \text{if session } S_j \text{ contains } q \text{ TSVs} \\ 0 & \text{otherwise} \end{cases} \tag{2.11}$$

With equation (2.11), $t(L_j) = \sum_{q=0}^{r} \delta_{jq} \cdot t(q)$. In addition, the following two constraints should be included in the model:

$$L_j = \sum_{q=0}^{r} q \cdot \delta_{jq} \tag{2.12}$$

which indicates that a session can have 0 to $r$ TSVs. Also,

$$\sum_{q=0}^{r} \delta_{jq} = 1 \tag{2.13}$$

indicating that the size of a session should be unique. For every session $S_j$, there will be exactly one value of $q$ for which $\delta_{jq} = 1$. Therefore, $\delta_{jq}$ determines the size of $S_j$. We can rewrite the objective function as

$$\text{Minimize} \sum_{j=1}^{N_{up}} \sum_{q=0}^{r} \delta_{jq} \cdot t(q) \tag{2.14}$$

The testing time for different session size $t(q)$ is a constant obtained from SPICE simulation as shown in Table 2.1. Thus, the objective function is linearized.

The complete ILP model is summarized in Figure 2.1. Both the number of variables and number of constraints (a measure of the complexity of the problem) of the ILP model is $O(N_{up}^2 T)$. For the example of Section 2.1, the ILP model produces 4 optimal test sessions in less than 3 seconds. Note that a globally optimal set of sessions is not guaranteed by this ILP model since the model is based on Condition 2.2.1 which is a non-necessary condition for solving Problem 2.2.1.

Obviously, the sessions generated by ILP model 1 guarantee that all TSVs can be identified if the total number of faulty TSVs in a TSV network does not exceed $m$. If there

Objective:

$$\text{Minimize} \sum_{j=1}^{N_{up}} \sum_{q=0}^{r} \delta_{jq} \cdot t(q)$$

Subject to:

1) $x_{ij} + x_{ik} - z_{ijk} \leq 1; \quad \forall i \in [1, T], \forall j \neq k \in [1, N_{up}]$

2) $x_{ij} + x_{ik} - 2z_{ijk} \geq 0; \quad \forall i \in [1, T], \forall j \neq k \in [1, N_{up}]$

3) $\sum_{i=1}^{T} z_{ijk} \leq 1; \ \forall j \neq k \in [1, N_{up}]$

4) $L_j = \sum_{i=1}^{T} x_{ij}; \ \forall j \in [1, N_{up}]$

5) $0 \leq L_j \leq r; \ \forall j \in [1, N_{up}]$

6) $\sum_{j=1}^{N_{up}} x_{ij} \geq m + 1; \ \forall i \in [1, T]$

7) $L_j = \sum_{q=0}^{r} \delta_{jq} \cdot q; \ \forall j \in [1, N_{up}]$

8) $\sum_{q=0}^{r} \delta_{jq} = 1; \ \forall j \in [1, N_{up}]$

Figure 2.1: ILP model 1 for finding near-optimal test sessions with specified identification capability.

are more than $m$ faulty TSVs within a network, by testing the sessions produced by ILP model 1, two possible situations may occur. First, not all TSVs can be uniquely identified as either being good or bad. Second, all TSVs are identified but the number of faulty TSVs is larger than $m$. In both situations, we conclude that there are more than $m$ faulty TSVs within the network and the chip can be discarded. Therefore, the sessions provided by ILP model 1 can always help make the right decision to either replace the identified bad TSVs or discard the chip as having too many faulty TSVs within a local silicon area.

## 2.3 Experimental Results

In this section, we compare the total test time and total number of sessions for the proposed ILP model 1, the heuristic method [39], and sequential TSV testing. Sequential TSV testing refers to testing each TSV within a network sequentially and individually. For ease of comparison, same as [39], we just simply assume all the generated sessions need to be tested. The *relative test time ratio* in this section refers to the ratio of total test time

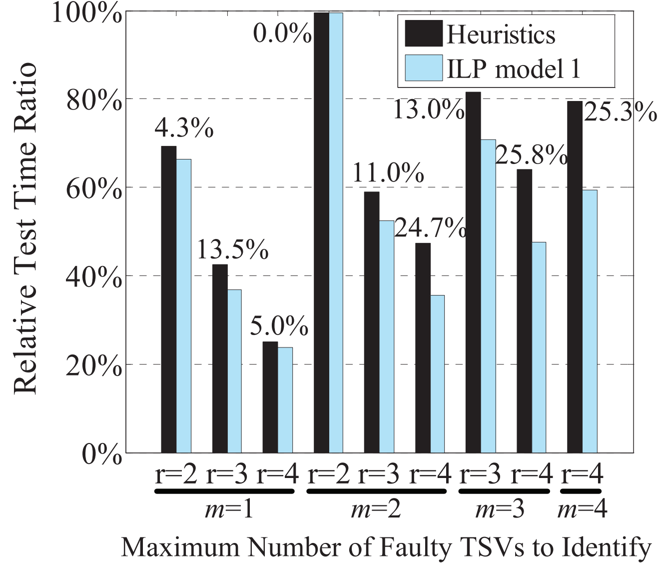



Figure 2.2: Test time comparison between ILP model 1 and heuristic [39] for a 20-TSV network.

to the total time of sequential TSV testing. Note again the *test time* in this work considers only the time to charge the capacitor and does not account for the time to move the probe card [39]. In all the figures, we only show the cases where relative test time ratio is smaller than 100%. For the cases not shown, sequential testing each TSV is more time-efficient than the other two methods. The ILP model 1 is solved using a commercial solver named CPLEX from IBM [3]. In all situations, the solver outputs the results in less than 40 seconds.

Figure 2.2 shows the relative test time ratio for both ILP model 1 and heuristic method considering different resolution constraint $r \in [2, 4]$ and different value of $m \in [1, 4]$ in a 20-TSV network. Note the results of ILP model 1 is always less than the results of heuristics. The number corresponding to each bar pair represents the relative test time improvement of ILP model 1 over heuristic. As can be seen from Figure 2.2, for a given $r$, the relative test time ratio for both methods increases as $m$ increases since pinpointing larger number of defective TSVs requires more sessions and takes longer time. For the same $m$, larger $r$ offers smaller test time ratio because now a session can hold more TSVs with less test time. Moreover, the total number of sessions generally decreases for larger $r$. Also, note that ILP model 1 always helps reduce test time further compared to the heuristic method, and

the improvement generally increases as $m$ increases. This demonstrates that for small $m$, the heuristic may behave well but as $m$ increases, outputs of the heuristic method deviate farther away from the optimal results. The largest improvement of ILP model 1 over heuristic method reaches 25.8% which happens when $T = 20$, $m = 3$ and $r = 4$.

Figure 2.3 examines the impact of number of TSVs within a network $T$ on relative test time ratio. We simulate 4 networks of different sizes with $m \in [1, 4]$ and $r$ fixed at 3. As can be seen the ILP model 1 reduces test time compared to heuristics regardless of the value of $T$. This relative improvement can reach as large as 38.2% as shown on top of the bar pair corresponding to $T = 11$, $m = 2$. It is also interesting to observe that the relative test time ratio of ILP model 1 remains pretty consistent across different values of $T$ for a given $m$. While for the heuristics, the test time ratio varies a lot for different network sizes. For example, when $m = 2$, the relative test time ratio of heuristics for an 11-TSV network is much larger than those for other networks. The unstable performance of the heuristic method is mainly due to its greedy nature in generating test sessions. These observations suggest that the proposed ILP model 1 is more robust across variations of TSV network parameters, and thus could eliminate the need for redesign and optimization of each individual TSV network on chip as required in the heuristic method [39].

We show the total number of test sessions generated by the two methods for 4 different networks in Figure 2.4 where $r$ is fixed at 4 and $m \in [1, 4]$. The number on top of each

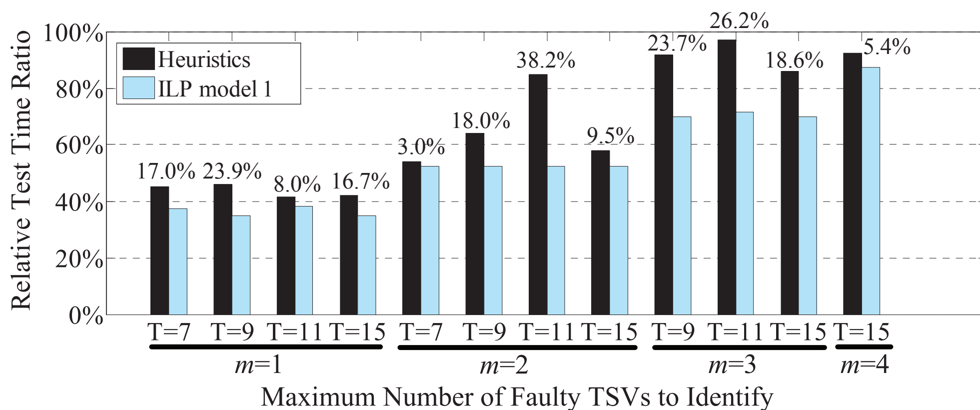



Figure 2.3: Test time comparison between ILP model 1 and heuristic [39] for resolution constraint $r = 3$.

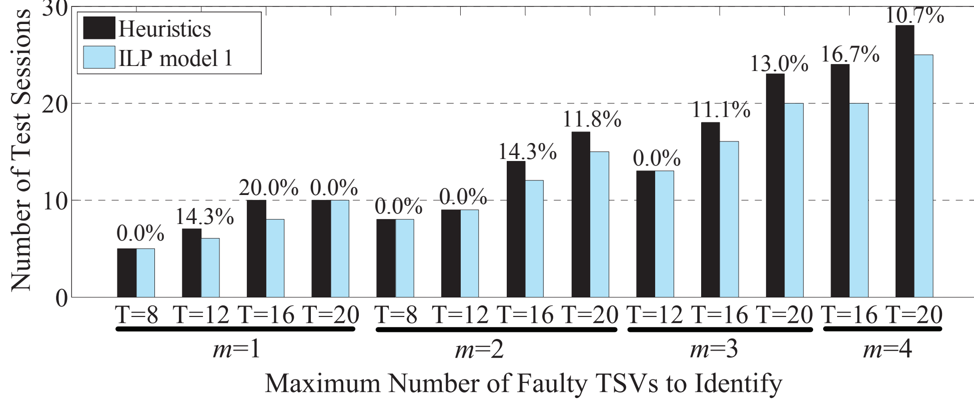



Figure 2.4: Comparison of number of sessions between ILP model 1 and heuristic [39] for resolution constraint $r = 4$.

bar pair represents the relative reduction of total sessions produced by ILP model 1 versus heuristics. As expected, generally a smaller number of sessions is produced for smaller $m$. For a larger TSV network, it is possible to reduce test time with number of test sessions larger than the total number of TSVs (i.e., $T$). As can be seen, the ILP model 1 sometimes produces the same number of sessions as the heuristic. However, our experimental results demonstrate that though the number of sessions produced by both methods can be the same, the sessions themselves are different. Sessions produced by ILP model 1 is guaranteed to be more time-efficient. For example, for $T = 8, m = 1, r = 4$ and $T = 8, m = 2, r = 4$, ILP model 1 produces same number of sessions as the heuristic but with test time reduced by 5.8% and 12.5%, respectively. In most cases, ILP model 1 helps reduce total number of sessions compared to the heuristic. For example, for $T = 16, m = 4, r = 4$, 4 out of 24 sessions can be further eliminated representing 16.7% relative reduction.

## 2.4 Conclusion

An ILP based model is proposed to generate near-optimal set of test sessions for pre-bond TSV probing. Extensive experiments demonstrate that ILP model 1 always reduces pre-bond TSV identification time compared to previous heuristic methods. Moreover, the test time reduction of the ILP model remains consistent for various TSV networks, and thus eliminates the need for separately designing and optimizing the test for each TSV network

as required by previous work. The proposed ILP model 1 is expected to significantly reduce pre-bond TSV test cost in real silicon. Our model is based on a sufficient but non-necessary condition for test session generation which still leaves space for future explorations, such as, possibly finding a necessary and sufficient condition to generate globally optimal set of sessions.

Chapter 3

An Optimal Probing Method of Pre-Bond TSV Fault Identification in 3D Stacked ICs

## 3.1 Introduction

For ease of the following illustration, let us first define the following terminologies.

1) *Fault map $\rho$.* Fault map $\rho$ represents positions of all defective TSVs within the network.

2) *Worst fault map.* Worst faulty map for a given TSV network refers to a fault map which takes most sessions to identify.

We define $|\rho|$ as the number of faulty TSVs within a fault map $\rho$. Test sessions generated by ILP model 1 are near-optimal, which guarantee that all fault maps with $|\rho| \leq m$ can be uniquely identified. However, a careful observation [76] shows that for identifying most of the fault maps with $|\rho| \leq m$, only a small portion of the total sessions need to be tested. Let's mention the example in section 2.1 again. For $T = 6$, $m = 1$, $r = 4$, ILP model 1 generated four optimal test sessions which are {1,2,3}, {1,4,5}, {2,4,6}, {3,5,6}. if $TSV_1$ is faulty, all 4 sessions need to be tested to identify it, but if $TSV_6$ is faulty only the first 3 sessions need to be tested to pinpoint it. By testing only 3 out of 4 sessions, test time is further saved by 25%. This observation further motivates us to develop an algorithm to terminate the test as soon as our goal of identification is reached.

In this chapter, we propose a fast dynamically optimized TSV identification algorithm to further speed up pre-bond TSV test from two aspects. First, during the identification process, any "currently unnecessary" session is skipped. Second, TSV test is terminated as soon as either all TSVs have been identified or the number of identified faulty TSV exceeds $m$, as the chip can be discarded due to lack of redundant TSVs and further test is useless.

```
Algorithm  Fast_TSV_Identification (All_sessions, T, m, t)
1.  Good=[ ]; Bad=[ ]; F_C=[ ]; test_time=0; tested_sessions=0;
2.  foreach session in All_sessions
        // Skip any currently unnecessary test session
3.      if (all TSVs in session have been identified) or
           (there is at least one bad TSV in session)
4.          Continue;
5.      test_time+=t(session);  // Test time accumulation
6.      tested_sessions+=1;     // Test session accumulation
        // Handle a passing session
7.      if session is tested as being good
8.          Add all TSVs in session to Good;
9.          foreach FC_session in F_C
10.             Remove any good TSV from FC_session;
11.             if length(FC_session)==1
12.                 Add the TSV in FC_session to Bad;
13.                 Remove the entire FC_session from F_C;
        // Handle a failing session
14.     else if session is tested as being bad
15.         Remove any good TSV from session;
16.         if length(session)==1
17.             Add the TSV in session to Bad;
18.         else
19.             Append session to F_C;
        // Termination conditions
20.     if ((length(Good)+length(Bad))==T or (length(Bad)>=m+1)
21.         Break;
22. Return  test_time, tested_sessions;
```

Figure 3.1: A dynamically optimized TSV identification algorithm.

## 3.2 A Dynamically Optimized TSV Identification Algorithm

The pseudo-code of the algorithm is shown in Figure 3.1, where argument $t$ represents the test time of sessions. Test time of a session in this work only refers to the charging time of $C_{charge}$, and it is related to the session size as seen from Table 2.1 [39, 41, 78]. The algorithm starts by initializing 3 empty lists named "Good", "Bad", and "F_C". The "Good" and "Bad" lists are used to contain the identified good and faulty TSVs, respectively. The faulty candidate list "F_C" is used to contain any failing session. The algorithm enumerates all the sessions generated by [41] or [78] and skips any "currently unnecessary" session, which refers to a session where either all TSVs in the session have been identified so far or there is at least one identified bad TSV in the session. A "currently unnecessary" session does not provide

any information of TSV identification. Although a session may be "currently unnecessary" for identifying some fault maps of a TSV network, it could be essential for identifying other fault maps of the same TSV network. So, none of the "currently unnecessary" sessions can be deleted. If a session is not skipped, it will be tested. If a session passes the test, all TSVs in the session are added to "Good", and we then use "Good" to refine "F_C." Here, the refinement refers to removing any identified good TSV from the targeted session (see line 10 of Figure 3.1). If after refinement any failing session in "F_C" contains only one TSV, that TSV is identified as defective and added to "Bad." If a session fails the test, "Good" is again utilized to refine this failing session (line 15 of Figure 3.1). If the session after refinement contains only one TSV, that TSV is added to "Bad." Otherwise, the refined failing session is appended to "F_C." The above procedure terminates as soon as any condition shown on line 20 in Figure 3.1 is satisfied.

### 3.3 Experimental Results

Table 3.1 shows the results of the proposed algorithm applied to various TSV networks. Column 1 shows parameters $T$ (network size), $m$ (redundant TSVs in network) and $r$ (resolution constraint). Column 2 gives the number of faulty TSVs ($\phi$) within the network. Column 3 shows the total number of sessions and total test time (in $\mu$s) for exhaustive application of sessions optimized by ILP model 1 [78]. The test time calculation is detailed in references [39, 41] and [78]. For a given value of $\phi$, we enumerate all possible fault maps and obtain the test time and number of tested sessions using the proposed algorithm of Figure 3.1. Column 4 shows the average number of tested sessions and average test time for identifying all fault maps containing $\phi$ faulty TSVs. Column 5 shows the relative reduction in column 4 over column 3. Column 6 shows the maximum number of sessions tested and the corresponding test time for identifying a fault map. Column 7 shows the relative reduction in column 6 over column 3.

Table 3.1: Exhaustive and dynamically optimized (Figure 3.1) application of TSV test sessions constructed by ILP model 1.

| Parameters | Number of | Optimum exhaustive | Dynamically optimized test | | | |
|---|---|---|---|---|---|---|
| | faulty TSVs | test [78] | Av. test sessions | Average reduction | Worst case sessions | Worst case reduction |
| $T$, $m$, $r$ | ($\phi$) | (# sessions, time in $\mu$s) | (# used, time in $\mu$s) | (sessions, time) | (# used, time in $\mu$s) | (sessions, time) |
| 8, 2, 3 | 0 | (8, 3.36) | (5.0, 2.10) | (37.5%, 37.5%) | (5, 2.10) | (37.5%, 37.5%) |
| | 1 | | (5.3, 2.25) | (32.8%, 32.8%) | (6, 2.52) | (25.0%, 25.0%) |
| | 2 | | (6.4, 2.71) | (19.1%, 19.1%) | (8, 3.36) | (0.0%, 0.0%) |
| | 3 | | (7.5, 3.17) | (5.3%, 5.3%) | (8, 3.36) | (0.0%, 0.0%) |
| 12, 3, 3 | 0 | (16, 6.72) | (7.0, 2.94) | (56.2%, 56.2%) | (7, 2.94) | (56.2%, 56.2%) |
| | 1 | | (7.5, 3.14) | (53.1%, 53.1%) | (9, 3.78) | (43.7%, 43.7%) |
| | 2 | | (8.7, 3.65) | (45.5%, 45.5%) | (12, 5.04) | (25.0%, 25.0%) |
| | 3 | | (10.3, 4.32) | (35.5%, 35.5%) | (14, 5.88) | (12.5%, 12.4%) |
| | 4 | | (11.8, 4.97) | (25.9%, 25.9%) | (16, 6.72) | (0.0%, 0.0%) |
| 15, 4, 3 | 0 | (25, 10.50) | (8.0, 3.36) | (68.0%, 68.0%) | (8, 3.36) | (68.0%, 68.0%) |
| | 1 | | (9.6, 4.03) | (61.6%, 61.6%) | (14, 5.88) | (44.0%, 44.0%) |
| | 2 | | (11.1, 4.68) | (55.3%, 55.3%) | (17, 7.14) | (32.0%, 32.0%) |
| | 3 | | (12.6, 5.33) | (49.2%, 49.2%) | (20, 8.40) | (20.0%, 20.0%) |
| | 4 | | (14.3, 6.03) | (42.5%, 42.5%) | (23, 9.66) | (8.0%, 8.0%) |
| | 5 | | (15.8, 6.66) | (36.5%, 36.5%) | (25, 10.50) | (0.0%, 0.0%) |
| 20, 4, 4 | 0 | (25, 9.50) | (9.0, 3.42) | (64.0%, 63.9%) | (9, 3.42) | (64.0%, 63.9%) |
| | 1 | | (10.8, 4.10) | (56.8%, 56.7%) | (15, 5.69) | (40.0%, 39.9%) |
| | 2 | | (12.3, 4.68) | (50.6%, 50.6%) | (18, 6.83) | (28.0%, 27.9%) |
| | 3 | | (13.9, 5.31) | (44.0%, 44.0%) | (21, 7.97) | (16.0%, 15.9%) |
| | 4 | | (15.1, 5.76) | (39.3%, 39.3%) | (24, 9.11) | (4.0%, 3.9%) |
| | 5 | | (18.0, 6.85) | (27.8%, 27.8%) | (25, 9.49) | (0.0%, 0.0%) |

We made four observations from Table 3.1. First, the average number of tested sessions and average test time is much less than the total number of sessions and total test time for any $\phi \leq m$ (repairable TSV network) or any $\phi > m$ (irrepairable TSV network). For example, the average percentage reduction reaches 68.0% for parameters $T = 15$, $m = 4$, $r = 3$, and $\phi = 0$. On average, the proposed algorithm greatly speeds up the pre-bond TSV identification process. Second, as $\phi$ increases the average percentage reduction decreases. This is expected as pinpointing larger number of faulty TSVs within a TSV network generally requires more sessions to be tested and costs more time. Third, in most cases even the maximum number of tested sessions is less than the total number of sessions. Fourth, as expected, the maximum number of tested sessions increases as $\phi$ increases for a given TSV network. From column 7, reduction in the worst case can be small for large $\phi$, requiring all sessions to identify a fault map. This scenario occurs when fault map contains $m$ or more faulty TSVs. The probability of such large numbers of faulty TSVs within a small localized silicon area may be negligible for a mature manufacturing process. Thus, the worst case percentage test time reduction could be quite significant.

### 3.4 Conclusion

The proposed TSV identification algorithm has two main advantages. First, the average number of tested sessions and test time are guaranteed to be small factions of total sessions and test time. Second, even for the worst fault map, for which most sessions are needed, not all sessions may be used, i.e., time saving can occur even in worst case scenarios. Reducing pre-bond TSV test time reduces pre-bond test cost.

Chapter 4

# SOS3: Three-Step Optimization of Pre-Bond TSV Test for 3D Stacked ICs

## 4.1 Introduction

In real silicon, TSV yield is expected to be more than 99% [5]. We calculated the probability of different numbers of failing TSVs within a TSV network, considering different TSV defect distributions. The results suggest that the probability of $\phi$ faults within a TSV network decreases dramatically as $\phi$ increases. This observation motivates us to emphasize the application order of test sessions so that pre-bond TSV test can be terminated as soon as possible. We make two contributions in this chapter. First, we propose an iterative greedy procedure for session sorting. Second, we combine the iterative greedy procedure with the ILP model 1 in chapter 2 and the TSV identification algorithm in chapter 3 and form a 3-Step test time Optimization Simulator ("SOS3") [77]. SOS3 consists of three steps, namely, ILP-based session generation, iterative greedy procedure for session sorting, and fast TSV identification algorithm for early test termination. Each step provides inputs to the next. In the experimental section, we calculate the test time expectation for various TSV networks. The results demonstrate that the session sorting procedure plays an important role in SOS3, as it helps further reduce test time expectation by as large as 31.8%. We also observe that with SOS3 the expectation of TSV identification time is much less than the total time of testing all sessions.

## 4.2 Probabilistic Analysis of Number of Faulty TSVs Within a TSV Network

In this section, we analyze the probability of different numbers of faulty TSVs within a network. TSV defect distributions can be broadly classified as two types, namely independent

defect distribution [81] and clustered defect distribution [57, 81]. For independent TSV defect distribution, the failing probability of a TSV is independent from each other. And the probability of $\phi$ faulty TSVs within a $T$-TSV network can be calculated as:

$$P(\phi) = \begin{pmatrix} T \\ \phi \end{pmatrix} p^{\phi}(1-p)^{T-\phi} \tag{4.1}$$

where $p$ is average TSV failing probability.

Defects clustering effect tries to model the scenario where the presence of a defective TSV increases the probability of more defects in close vicinity [57, 81]. Reference [81] formulates this clustering effect by considering 1) a defect cluster center [57, 81] consists of one single defective TSV, 2) the failing rate of $TSV_i$ is inversely proportional to the distance from the existing cluster center. This formulation is shown in equation 4.2.

$$p(TSV_i) = p \cdot (1 + (\frac{1}{d_{ic}})^{\alpha}) \tag{4.2}$$

where $p(TSV_i)$ represents the failing probability of $TSV_i$, $d_{ic}$ represents the distance between $TSV_i$ and the cluster center, and $\alpha$ is the clustering coefficient. A larger value of $\alpha$ implies less clustering. As $\alpha \to \infty$, the defect distribution becomes independent defect distribution.

The clustered model needs to take the TSV location information into account. Since the number of TSVs within a network is typically less than 20, we consider each TSV network as a 5-by-5 matrix. The value 5 is chosen based on the ratio of the pitch of current probe needle and the pitch of realistic TSVs [43, 53]. We randomly put $T$ TSVs on the integral coordinates of the matrix to obtain the location information of each TSV. After that, we employ equation 4.2 to analyze the probability of different numbers of defective TSVs ($\phi$) within a network. As in [81], we assume each TSV network has only one defect cluster and defect clusters within different networks do not interfere with each other.

Table 4.1: Probability of different number of failing TSVs $\phi$ within a 15-TSV network.

| **Defect distribution** | **TSV yield** | **Number of faulty TSVs $\phi$** | | | |
|---|---|---|---|---|---|
| | | **0** | **1** | **2** | **$\geq$3** |
| Independent | 99.5% | 92.76% | 6.99% | 0.25% | 0.00% |
| | 99.0% | 86.01% | 13.03% | 0.92% | 0.04% |
| | 98.0% | 73.86% | 22.60% | 3.23% | 0.31% |
| Clustered $\alpha$=1 | 99.5% | 92.76% | 6.70% | 0.35% | 0.19% |
| | 99.0% | 86.01% | 12.07% | 1.26% | 0.66% |
| | 98.0% | 73.86% | 19.55% | 4.16% | 2.43% |
| Clustered $\alpha$=2 | 99.5% | 92.76% | 6.78% | 0.31% | 0.15% |
| | 99.0% | 86.01% | 12.39% | 1.13% | 0.47% |
| | 98.0% | 73.86% | 20.60% | 3.81% | 1.73% |

Table 4.1 shows the probability of different values of $\phi$ for a 15-TSV network. We vary the TSV yield from 98% to 99.5% to accommodate different levels of maturity of the manufacturing processes. The clustering coefficient $\alpha$ is set as 1 and 2, similar to the settings in [81] and [27]. Note the values under clustered defect distribution are averaged results of 100 Monte Carlo runs, with each run randomly placing 15 TSVs on the 5-by-5 matrix. By doing this, we try to simulate all possible TSV placements in real silicon. Three observations are summarized in Table 4.1. First, no matter what defect distribution it is, the probability of $\phi = 0$ is the largest and even much larger than the sum of the rest situations with $\phi > 0$. Second, the sum of probabilities of $\phi = 0$ and $\phi = 1$ is almost 1 in all situations, and the probability of $\phi \geq 3$ is low. Third, as TSV yield decreases, the probability of $\phi = 0$ decreases. Motivated by the above observations, we propose to sort the order of test sessions to reduce the expectation of pre-bond TSV test time, as explained in the next section.

## 4.3 An Iterative Greedy Procedure for Test Session Scheduling

We express the expectation $E(\Gamma)$ of test time ($\Gamma$) of a TSV network as follows:

$$E(\Gamma) = \sum_{\text{Any}\rho} \gamma(\rho) P(\rho) \tag{4.3}$$

where $\gamma(\rho)$ is the identification time to determine $\rho$ using the fast TSV identification algorithm in section 3.2 [76], and $P(\rho)$ is the occurrence probability of $\rho$. Note that $\rho = \emptyset$

or $|\rho| = \phi = 0$ means that all TSVs within the network are fault-free. We formulate two problems to be solved.

**Problem 4.3.1.** Given a series of $N$ test sessions that can uniquely identify up to $m$ faulty TSVs within a TSV network of $T$ TSVs, find an optimal order to apply those sessions so that the expectation of pre-bond TSV test time is minimized for this TSV network.

To solve Problem 4.3.1, a straightforward method is to find all permutations of test sessions, and for each permutation calculate the test time expectation using equation 4.3. The permutation which yields minimum $E(\Gamma)$ would be the selected choice. However, there can be $N!$ permutations and $2^T - 1$ fault maps. So the identification algorithm [76] must be run $N! \cdot (2^T - 1)$ times, which is highly time-consuming even for a small network. Fortunately, we notice that the probability of different numbers of faulty TSVs is inversely proportional to $\phi$. Specifically,

$$\sum_{|\rho|=i} P(\rho) \gg \sum_{|\rho|=j} P(\rho) \text{ for any } i < j \tag{4.4}$$

where $\sum_{|\rho|=i} P(\rho) = P(\phi = i)$ and $\sum_{|\rho|=j} P(\rho) = P(\phi = j)$.

Motivated by the fact that $P(\rho)$ is large for small $|\rho|$ and decreases dramatically as $|\rho|$ increases, if we can reduce $\gamma(\rho)$ for small $|\rho|$ the test time expectation should be greatly reduced. For example, the probability of $P(\rho = \emptyset)$ (all TSVs being good in a network) dominates. In case of $\rho = \emptyset$, all TSVs are identified as good TSVs as long as the already tested sessions covered all TSVs. Based on this observation, Problem 4.3.2 is formulated as follows.

**Problem 4.3.2.** Given $N$ test sessions that can uniquely identify up to $m$ faulty TSVs within a network of $T$ TSVs, select $M$ out of $N$ sessions such that these $M$ sessions cover each TSV at least once and the total test time of the selected $M$ sessions is minimum.

Problem 4.3.2 can be solved by constructing an ILP model (named "ILP model 2" to differentiate it from ILP model 1 in section 2.2). We introduce a variable $P_j$, $j \in [1, N]$,

where

$$P_j = \begin{cases} 1 & \text{if session } S_j \text{ is selected (or picked)} \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

Then, the ILP model 2 is described as follows:

Objective: Minimize $\sum_{j=1}^{N} t(L_j) \cdot P_j$

Subject to constraint: each $TSV_i, i \in [1, T]$, is tested at least once by the selected sessions.

$L_j$ represents the size of session $S_j$, and $t(L_j)$ the test time of $S_j$, which is a constant for a given $L_j$. The numbers of variables and constraints for ILP model 2 are $O(NT)$ and $O(T)$, respectively. In all our experiments, ILP model 2 is solved in 1 second or less. The $M$ sessions covering all TSVs with minimal time will be sorted and tested before the rest of the sessions. This will reduce $\gamma(\rho = \emptyset)$ and thus reduce the test time expectation.

As can be seen from Table 4.1, $P(\phi = 1)$ is also outstanding. If we can further reduce $\gamma(\rho)$ with $|\rho| = 1$, the test time expectation should be further reduced. The $N$ test sessions in Problems 4.3.1 and 4.3.2 can be produced by either the ILP model 1 [78] or the heuristic method [39]. Sessions produced by both methods have characteristics such that if each TSV is covered (or tested) by at least two sessions, any single faulty TSV can be uniquely identified. To reduce $\gamma(\rho)$ with $|\rho| = 1$, we can hold the $M$ sessions produced by ILP model 2, and find $M_1$ sessions from the remaining $N - M$ sessions such that these $M + M_1$ sessions will cover each TSV at lease twice and the test time of the $M_1$ sessions is the minimum. Next we explain how ILP model 2 can be again utilized to find these $M_1$ sessions. We first count the times each TSV is covered by the first $M$ sessions, and put the TSVs which have been covered only once into a list named "$TSV_set$". ILP model 2 can be utilized to find $M_1$ sessions out of the $N - M$ sessions such that each $TSV_i, i \in TSV_set$ is covered (or tested) at least once by these $M_1$ sessions and the total test time of these $M_1$ sessions is the minimum. The produced $M_1$ sessions will be sorted and tested directly after the $M$ sessions.

```
Procedure    Test_session_sorting
  Initialization:
     Original_sessions = All the sessions;
     TSV_set = All the TSVs within network;
     Sorted_sessions = [  ];
     Stop_index = any integer ∈ [1, m+1];
     k=1;
  Iterative Execution:
    while (k <= Stop_index) do
      {
        Step1:  Use ILP model 2 to find a subset of sessions from
                Original_sessions which cover each TSV within
                TSV_set at least once with minimum test time;
        Step2:  Append all sessions produced by Step1 to the end
                of Sorted_sessions;
        Step3:  Remove these sessions produced by Step1 from
                Original_sessions;
        Step4:  Based on Sorted_sessions, calculate the times each
                TSV is covered;
        Step5:  Set TSV_set ← TSVs which have been covered by
                only k times;
        Step6:  k++;
      }
  Return Final Results:
     Append Original_sessions to the end of Sorted_sessions;
     Return Sorted_sessions;
```

Figure 4.1: Pseudo-code for iterative test session sorting.

These $M + M_1$ sessions first guarantee $\gamma(\rho = \emptyset)$ is minimized and based on this premise further minimize $\gamma(\rho)$ with $|\rho| = 1$ (simply represented as $\gamma(|\rho| = 1)$). Similar procedure can be repeated for further reduction of $\gamma(|\rho| = 2)$, $\gamma(|\rho| = 3)$, $\cdots$ , until $\gamma(|\rho| = m)$.

We summarize the overall procedure for session sorting in Figure 4.1. ILP model 2 is iteratively utilized in *Test_session_sorting* procedure with each execution tries to find a subset of sessions from "*Original_sessions*" so as to cover all the TSVs within "*TSV_set*" at least once with minimum time. The greedy nature of our procedure is obvious since it puts higher priority on reducing $\gamma(\rho)$ with smaller $|\rho|$. The run time of the procedure is (almost) equal to the run time of ILP model 2 times how many times ILP model 2 is executed, which is determined by "*Stop_index*" in Figure 4.1. Note "*Stop_index*" can be set as any value from 1 to $m + 1$. When "*Stop_index*" is 1, *Test_session_sorting* will only reduce $\gamma(\rho = \emptyset)$

by finding $M$ sessions which covered all TSVs at least once with minimum time. When "*Stop_index*" is $m+1$, the procedure will first reduce $\gamma(\rho = \emptyset)$, and then reduce $\gamma(|\rho| = 1)$, and then reduce $\gamma(|\rho| = 2)$, $\cdots$, all the way up to reducing $\gamma(|\rho| = m)$.

### 4.4 A Three-step Test Time Optimization Simulator

In this section we proposed a 3-Step test time Optimization Simulator (SOS3). The first step of SOS3 is ILP model 1 [78] introduced in chapter 2 for test session generation. Note that we choose ILP model 1 instead of the heuristic method [39] because test sessions generated by both methods have exactly the same TSV identification capability. However, the ILP model generates fewer sessions and is more time-efficient. The second step of SOS3 is the proposed iterative greedy procedure for session sorting. This procedure accepts the $N$ sessions from step 1 as the inputs and sort the sessions to reduce test time expectation. The last step is the fast TSV identification algorithm [76] introduced in chapter 3. This algorithm takes the sorted list of sessions as the inputs and finishes the identification process as soon as any termination condition happens. By integrating the session sorting procedure and the fast TSV identification algorithm in SOS3, the pre-bond TSV probing can be terminated as soon as possible with largely reduced test time expectation.

Figure 4.2 illustrates the overall diagram of SOS3. The inputs to SOS3 contain three pieces of information: 1) the TSV and TSV network information, 2) the probing technology information, and 3) the on-chip TSV redundancy information. The outputs of SOS3 are: 1) the sorted list of sessions, 2) identified TSVs, 3) test time expectation, and 4) expectation of number of tested sessions.

### 4.5 Experimental Results

In this section we compare both the expectation of test time and number of tested sessions between two different simulators: SOS3 and SOS2 (2-Step test time Optimization Simulator). The only difference between SOS2 and SOS3 is that the iterative session sorting

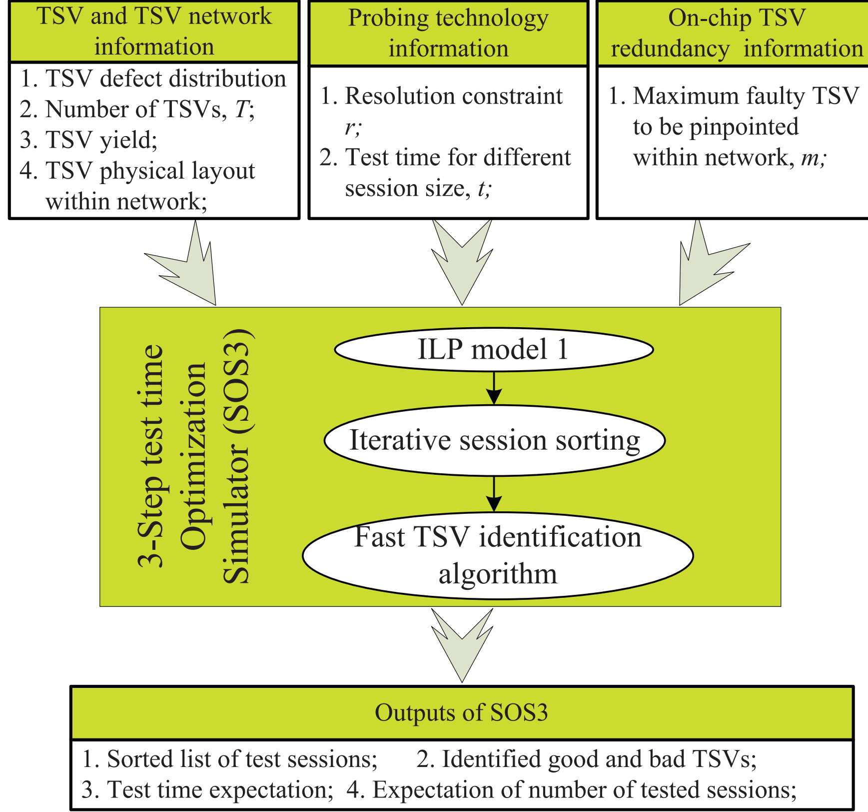



Figure 4.2: Three-step test time optimization simulator.

procedure is eliminated in the former. By comparing these two simulators, we try to illustrate the importance of session sorting for test time reduction. Note we did not compare the test time expectation of SOS3 to that of the heuristic method [39] due to the following two reasons. First, ILP model 1 returns a smaller number of sessions and requires less test time; Second, the session sorting procedure in combination with the identification algorithm helps reduce test time expectation further.

In this section, the expectation of test time $E(\Gamma)$ is estimated as follows.

$$E(\Gamma) = \begin{cases} \sum\limits_{|\rho|<2} \gamma(\rho)P(\rho) + TT \sum\limits_{|\rho|\geq 2} P(\rho) & \text{if } m = 1 \\ \sum\limits_{|\rho|\leq 2} \gamma(\rho)P(\rho) + TT \sum\limits_{|\rho|\geq 3} P(\rho) & \text{if } m \geq 2 \end{cases} \tag{4.6}$$

where $TT$ represents the total time of testing all the sessions produced by ILP model 1 [78].

Similarly the expectation of number of tested sessions $E(S)$ in this section is estimated as follows:

$$E(S) = \begin{cases} \sum\limits_{|\rho|<2} \eta(\rho)P(\rho) + N \sum\limits_{|\rho|\geq 2} P(\rho) & \text{if } m = 1 \\ \sum\limits_{|\rho|\leq 2} \eta(\rho)P(\rho) + N \sum\limits_{|\rho|\geq 3} P(\rho) & \text{if } m \geq 2 \end{cases} \tag{4.7}$$

where $\eta(\rho)$ represents the number of tested sessions for identification of fault map $\rho$ using the TSV identification algorithm [76]. $N$ represents the total number of sessions produced by ILP model 1 [78].

Equations 4.6 and 4.7 are explained as follows. For a TSV network with $m = 1$, we simply assume any fault map with $|\rho| \geq 2$ will cause all sessions to be tested. This is because sessions generated for $m = 1$ are not intended for identifying more than one faulty TSV. Moreover, $P(\phi \geq 2) = \sum\limits_{|\rho|\geq 2} P(\rho)$ is low. Such a low probability has negligible impact on expectation calculation. For TSV networks with $m \geq 2$, we assume all sessions need to be tested to identify fault maps with $|\rho| \geq 3$. This is because it generally takes most of the sessions to identify large number of defective TSVs (like $|\rho| \geq 3$). Moreover, $P(\phi \geq 3) = \sum\limits_{|\rho|\geq 3} P(\rho)$ is relatively low, referring to Table 4.1. Such a low probability has little impact on expectation calculation.

Based on equations 4.6 and 4.7, we compare SOS2 and SOS3 for various values of $T$, $m$, $r$. The commercial ILP solver CPLEX [3] is again used in our experiments. For all simulations, SOS3 and SOS2 provide the outputs in less than one minute. The expectation of number of tested sessions and test time for both SOS2 and SOS3 are shown in Tables 4.2 and 4.3, respectively. We provide an insightful evaluation of SOS3 by varying TSV yield from a low value of 98.0% to a practically expected value of 99.5%. Note we only show the results under clustered defect distribution with $\alpha = 1$, since the results are very similar for the remaining two defect distributions in Table 4.1.

Table 4.2: Expectation of number of tested sessions, defect clustering coefficient $\alpha = 1$, data shows (sessions for SOS2, sessions for SOS3, reduction by SOS3).

| **Parameters** $T$, $m$, $r$ | **Total number of sessions,** $N$ | **Expected number of tested sessions,** $E(S)$ | | |
|---|---|---|---|---|
| | | TSV yield = 99.5% | TSV yield = 99.0% | TSV yield = 98.0% |
| 8, 1, 2 | 8 | (5.0, 4.0, 19.3%) | (5.1, 4.1, 18.7%) | (5.2, 4.3, 17.4%) |
| 8, 2, 3 | 8 | (5.0, 4.0, 19.4%) | (5.0, 4.1, 18.8%) | (5.1, 4.2, 17.6%) |
| 11, 1, 2 | 11 | (8.0, 6.0, 24.2%) | (8.1, 6.2, 23.5%) | (8.2, 6.4, 21.8%) |
| 11, 2, 3 | 11 | (6.0, 4.1, 31.6%) | (6.1, 4.3, 30.0%) | (6.3, 4.6, 26.8%) |
| 15, 2, 2 | 23 | (10.0, 8.1, 19.3%) | (10.2, 8.3, 18.6%) | (10.6, 8.7, 17.0%) |
| 15, 3, 3 | 20 | (7.1, 6.1, 13.5%) | (7.3, 6.4, 12.8%) | (7.8, 6.9, 11.2%) |
| 16, 3, 4 | 16 | (6.1, 5.2, 15.3%) | (6.4, 5.5, 14.0%) | (6.9, 6.1, 11.6%) |
| 16, 4, 4 | 20 | (5.2, 4.3, 17.9%) | (5.6, 4.7, 16.0%) | (6.2, 5.4, 12.8%) |
| 20, 3, 4 | 20 | (9.1, 6.3, 31.1%) | (9.4, 6.7, 28.8%) | (9.9, 7.5, 24.2%) |
| 20, 4, 4 | 25 | (9.2, 6.3, 31.3%) | (9.5, 6.7, 29.1%) | (10.3, 7.8, 24.5%) |

Table 4.3: Expectation of test time ($\mu$s), defect clustering coefficient $\alpha = 1$, data shows (test time for SOS2, test time for SOS3, reduction by SOS3).

| **Parameters** $T$, $m$, $r$ | **Test time of all sessions,** $TT$ | **Expectation of test time,** $E(\Gamma)$ | | |
|---|---|---|---|---|
| | | TSV yield = 99.5% | TSV yield = 99.0% | TSV yield = 98.0% |
| 8, 1, 2 | 4.24 | (2.66, 2.15, 19.3%) | (2.71, 2.20, 18.7%) | (2.76, 2.28, 17.4%) |
| 8, 2, 3 | 3.36 | (2.10, 1.69, 19.4%) | (2.13, 1.73, 18.8%) | (2.15, 1.77, 17.6%) |
| 11, 1, 2 | 5.83 | (4.25, 3.21, 24.2%) | (4.31, 3.29, 23.5%) | (4.36, 3.41, 21.8%) |
| 11, 2, 3 | 4.62 | (2.54, 1.73, 31.6%) | (2.58, 1.81, 30.0%) | (2.64, 1.93, 26.8%) |
| 15, 2, 2 | 12.19 | (5.33, 4.30, 19.3%) | (5.44, 4.42, 18.6%) | (5.61, 4.65, 17.0%) |
| 15, 3, 3 | 8.40 | (2.99, 2.58, 13.5%) | (3.09, 2.69, 12.8%) | (3.27, 2.90, 11.2%) |
| 16, 3, 4 | 6.08 | (2.34, 1.98, 15.3%) | (2.44, 2.10, 14.0%) | (2.63, 2.32, 11.6%) |
| 16, 4, 4 | 7.60 | (1.99, 1.63, 17.9%) | (2.12, 1.78, 16.0%) | (2.38, 2.08, 12.8%) |
| 20, 3, 4 | 7.60 | (3.47, 2.39, 31.1%) | (3.58, 2.55, 28.8%) | (3.79, 2.87, 24.2%) |
| 20, 4, 4 | 9.50 | (3.50, 2.40, 31.3%) | (3.64, 2.58, 29.1%) | (3.93, 2.97, 24.5%) |

The first column of both tables shows various TSV networks with different $T$, $m$, $r$. The second column of Table 4.2 and Table 4.3 show the total number of sessions and total test time of all sessions produced by ILP model 1 [78], respectively. Columns 3, 4 and 5 of both tables consist of elements with each element consisting of 3 values, i.e., (the expected value of SOS2, the expected value of SOS3, and the relative percentage improvement of SOS3 over SOS2). Note that all values are the averaged results of 100 Monte Carlo runs, with each run corresponding to a different TSV placement in a 5-by-5 matrix. By doing this we try to simulate all possible TSV layouts in real silicon.

We can make four observations from Table 4.2 and 4.3. First, both the expectation of number of tested sessions and the expectation of test time are only a small portion of total number of sessions and total test time, respectively. This demonstrates that SOS2 and SOS3 greatly speed up the pre-bond TSV identification process and thus reduce pre-bond TSV test cost. As can be seen from Table 4.2, the total number of sessions can reach as high as 25, but the expected number of tested sessions is always no more than 8.7 for SOS3. For TSV network with $T = 20$, $m = 4$, $r = 4$, the test time expectation of SOS3 is only 31.3% of the total test time, considering 98.0% TSV yield. Second, both the expectation of test time and number of tested sessions increases for lower TSV yield. This is because the probability of having larger numbers of defective TSVs within a network increases as TSV yield decreases. Identifying larger numbers of faulty TSVs typically takes more sessions and longer test time for both SOS2 and SOS3. Third, SOS3 helps further reduce the expectation value compared to SOS2. For example, for $T = 11$, $m = 2$, $r = 3$, SOS3 further reduces test time expectation by 31.6% compared to SOS2 considering 99.5% TSV yield. Fourth, the improvement of SOS3 over SOS2 decreases as TSV yield decreases. This is because the session sorting procedure within SOS3 puts higher priority on reducing the identification time for smaller value of $|\rho|$. To identify fault map with large $|\rho|$ (like $|\rho| \geq 2$), SOS3 does not have much advantage over SOS2. As TSV yield decreases, $P(\phi \geq 2) = \sum_{|\rho| \geq 2} P(\rho)$ increases and the advantage of SOS3 also slightly decreases. However, even the smallest improvement of SOS3 over SOS2 is 11.2% in both tables. The large differences between SOS2 and SOS3 illustrate the significance of session sorting in reducing test time expectation.

### 4.6 Conclusion

In this chapter, a session sorting procedure is proposed to sequence test sessions in such a way that the pre-bond TSV test can terminate as soon as possible for small numbers of faulty TSVs within a network. This session sorting procedure, although greedy in nature, is based on iterative execution of an ILP model. We also propose a 3-step test time optimization

simulator (SOS3) to reduce pre-bond test time expectation. In SOS3, an existing ILP model is first used to generate a series of test sessions with certain identification capability. Then, session sorting is conducted based on the generated sessions. Lastly, an existing fast TSV identification algorithm is used for early test termination. Extensive experimental results for various TSV networks demonstrate the benefit of session sorting on reducing test time expectation. We also show how SOS3 guarantees that the test time expectation is always a small portion of the total time of testing all the sessions. As a framework, SOS3 is expected to greatly reduce pre-bond TSV test cost in real silicon.

Chapter 5

# A Novel Wafer Manipulation Method for Yield Improvement and Cost Reduction of 3D Wafer-on-Wafer Stacked ICs

## 5.1 Introduction

In this chapter, we introduce a novel wafer manipulation method for yield improvement and cost reduction of 3D wafer-on-wafer stacked ICs. The pre-bond TSV testing methods introduced in chapters 2 to 4 serve as the basis and provide useful known-good-die information for wafer matching.

A bottleneck in the wafer-on-wafer stacking is its relatively low compound yield, especially for large number of stacked layers and low wafer yields. Here, compound yield is defined as the final yield of the 3D IC, ignoring any defect induced during fabrication process such as stacking, bonding, interconnect formation, packaging, etc. It is a theoretical value based on the simple assumption that a pre-bond tested good die stacked on another pre-bond tested good die (or good partial stack) would definitely produce a good stack (or good 3D IC). Without any matching, compound yield of randomly stacked wafers will decrease exponentially as more die are stacked, thus dramatically increasing the cost of wafer-on-wafer fabricated 3D ICs. To improve the compound yield two kinds of previous efforts are worth mentioning:

1) Matching algorithms have been proposed so as to select the best matching wafers to stack instead of stacking them randomly [47, 51, 52, 54, 58, 60, 65].

2) Exploiting empirically observed defect distribution models (e.g., wafer maps with radially clustered defects [52]) or special layouts of wafers (such as fabricating wafer

with rotational symmetry so that two wafers can be matched in more ways than one [51, 79, 74, 80]) have been proposed.

Our effort in this chapter has five parts:

1) A hybrid wafer-on-wafer stacking procedure is proposed in Section 5.3.5, which combines advantages from several methods [51, 58, 60]. With a combination of best practices in the existing work, this hybrid procedure serves as a reference for comparing with the novel sector symmetry and cut method, which is the core contribution of this work.

2) A novel manipulation scheme of wafers is introduced. To be specific, wafers fabricated with rotational symmetry are cut into identical sectors (called subwafers). We refer to two subwafers as identical if they have the same die distribution and die orientation. These identical subwafers are then used to replenish the repository. A repository consists of wafers corresponding to a specific layer of a 3D IC stack. The simulation results show that sector symmetry and cut method produces much higher compound yield than that of existing methods [47, 51, 52, 54, 58, 60, 65]. For example, compared with the work in [52, 58, 60], the relative improvement of compound yield can reach as high as 189%.

3) We derive mathematical formulations for die per wafer (DPW) calculation considering rotational symmetric wafers. We demonstrate that larger capability of rotation produces more flexibility of wafer matching, but also increases the number of die lost due to the sector symmetry. Extensive experiments (parts of experimental results are shown in the appendix) have been done to find the optimal number of cuts.

4) We compare the compound yield of different stacking procedures under various defect distribution models.

5) We construct a cost model of SSC$n$ and conduct detailed cost analysis of SSC4. We demonstrate that compared to conventional methods, SSC4 largely reduces the 3D IC cost under various defect distribution models, especially for situations where the number of stacked layers is large.

The rest of this chapter is organized as follows. Section 5.2 introduces the background and motivation for this work. Section 5.3 discusses various aspects of wafer-on-wafer stacking closely related to this work. An original contribution is given in Section 5.4 that proposes a novel wafer manipulation method named *sector symmetry and cut* (SSC$n$). Simulation results are presented in Section 5.5 where yield comparison with related work [52, 58, 60] is given. The cost-effectiveness of SSC$n$ is analyzed in Section 5.6. Section 5.7 concludes this chapter.

## 5.2 Background and Motivation

Since a major bottleneck in wafer-on-wafer stacking is the low compound yield, many researchers have proposed optimal matching algorithms to improve the yield. Smith et al. [54] stack wafers with same or similar wafer maps from different repositories. Reda et al. [47] propose several matching algorithms including a globally greedy matching, an iterative matching heuristic (IMH), and a global optimal matching based on integer linear programming (ILP). Verbree et al. [65] propose an iterative greedy matching algorithm, which applies globally greedy matching to only two repositories at a time. Both proposals [47, 65] are based on static repositories, which means none of the repositories will be replenished until they run out of wafers. Contrary to the static repository scheme, Taouil et al. [58, 60] propose the concept of running repositories. After a wafer leaves a repository, a new wafer immediately enters that repository so the repository will always be full of wafers. This new scheme of replenishing repositories is proven to offer higher compound yield and, more importantly, lower run time complexity than the static repository system.

All of the above methods consider only wafer maps with uniform defect distribution, which is not a good model for describing defect distribution in the real world. It is well known that the defects on wafers are clustered [22, 61, 70, 71]. A clustered defect distribution model was first applied by Singh [52] to wafer-on-wafer stacking. Singh's work shows that with the same stacking procedure, more practical wafer maps generate higher compound yield than wafer maps with uniformly distributed defects. However, the matching algorithm [52] is based on a static repository instead of running repository, which may not fully exploit the advantage of the clustered defect model. Also, in industry, there may be various kinds of defect distribution models on the wafer.

Singh [51] proposes a way to fabricate wafers with rotational symmetry such that two wafers can be matched four ways to find the best match. The rotational symmetry offers a new strategy regardless of what matching algorithm is used. With rotations the repository size is virtually multiplied by the rotation number (4 in this case), which is helpful for the improvement of compound yield. Weaknesses of the contribution [51] include an impractical assumption of uniform defect distribution and the use of only a static repository.

Figure 5.1 shows four different aspects of the stacking procedures: defect distribution models, wafer manipulations, repository replenishment schemes, and matching algorithms. Notice that these aspects are generally independent of each other and any alternative can be generally selected for one aspect without interfering with choices for others. In Figure 5.1, a top to bottom path shows the choices made by a referenced work. For example, the leftmost path [47] means that uses a uniform defect distribution model, takes no action for wafer manipulation, and uses greedy, IMH, and an ILP matching algorithms based on static repository replenishment scheme. Viewed horizontally, Figure 5.1 arranges existing works in an ascending order of their publication date.

To express the motivating factors for the present work, let us examine the two rightmost paths in Figure 5.1:

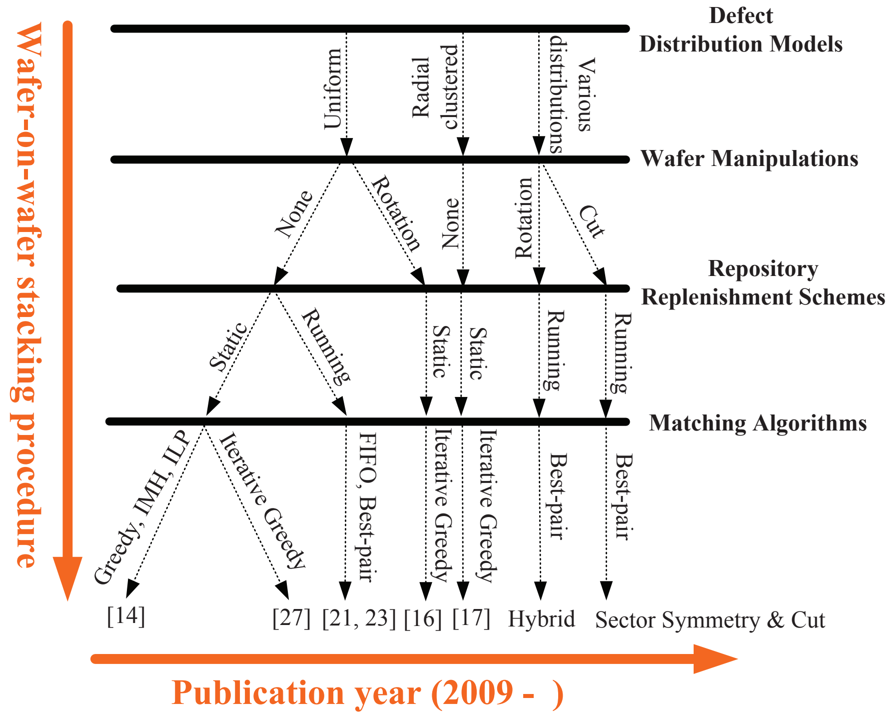



Figure 5.1: Wafer-on-wafer stacking procedures.

1) If we consider the manipulation of wafer rotation [51], and wafer matching based on a running repository scheme [58, 60], the compound yield should be improved. That motivates the *hybrid* scheme.

2) The motivation for a *sector symmetry and cut* strategy comes from the realization that compared with die-on-die and die-on-wafer stacking, the low yield of wafer-on-wafer stacking is due to the restriction that a die on a wafer must be matched with exactly one die on another wafer. In other words, any pair of dies to be bonded together must occupy the same geometric position on respective wafers. If the wafers are cut into smaller parts of identical shape, then the effect of this restriction can be reduced. Moreover, if the wafers being cut are fabricated with rotationally symmetric sectors, then each cut sector (subwafer) will look identical. As will be illustrated in this chapter, greater matching flexibility and higher compound yield can be achieved. Cost analysis of this manipulation method also shows large reduction of 3D IC cost.

## 5.3 Preliminaries

### 5.3.1 Defect Distributions on a Wafer

A negative binomial defect distribution model [9, 48, 55, 56] has been widely used to estimate the die yield as a function of defect density, die area, and a clustering parameter. However, this model does not provide enough information for us to generate wafer maps with certain observed patterns like radial symmetry, periodicity, offset, etc. Because of strict confidentiality in the industry, data on real wafers are not always exposed. This explains why most publications assume a uniform distribution of defects [47, 51, 54, 58, 60, 65]. The problem with this assumption is that the *unrealistic* uniform distribution model always leads to a pessimistic compound yield in the wafer-on-wafer stacking procedure [52].

Based on previous literature [14, 16], nine patterns of wafer maps corresponding to different defect distributions are generated for the yield analysis of wafer-on-wafer stacking procedure. The spatial probability functions of these nine patterns are shown in Figure 5.2 where different gray levels correspond to different levels of yield ranging from zero (black pixel) to one (white pixel).

Briefly, the nine defect distribution patterns are:

- Pattern 1: A large central spot showing the low yield at the center of the wafer.
- Pattern 2: A shifted semi-ring showing higher yield at the lower right corner of the wafer.
- Pattern 3: A slightly shifted small spot.
- Pattern 4: A very thin centered ring showing radial yield degradation.
- Pattern 5: A mixed pattern of repetitive rows and shifted semi-ring.
- Pattern 6: A shifted semi-ring showing higher yield level at the right corner of the wafer.

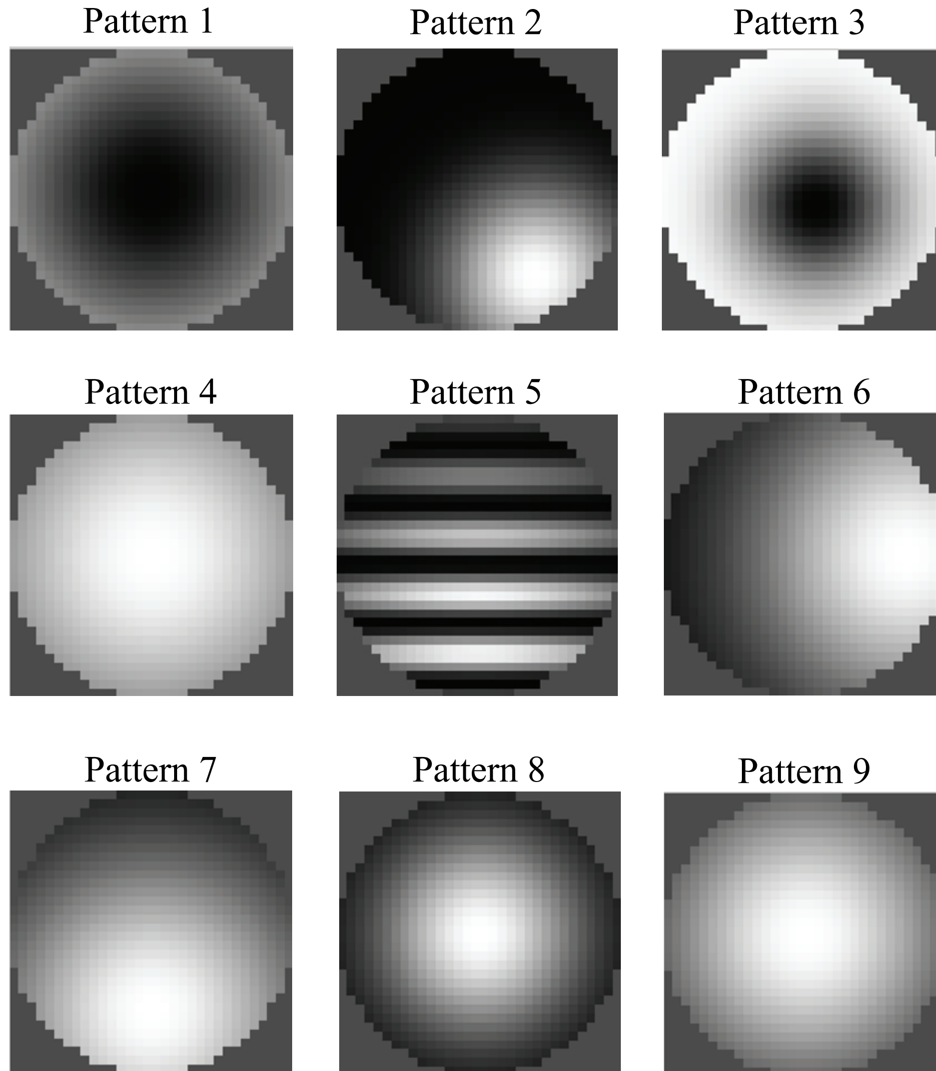



Figure 5.2: Gray maps showing the yield level distribution on the wafer.

- Pattern 7: A shifted semi-ring showing higher yield level at the lower part of the wafer.
- Pattern 8: A thick centered ring showing radial yield degradation.
- Pattern 9: A relatively thin centered ring showing radial yield degradation.

In this paper, experiments are conducted based on the wafer maps generated from Figure 5.2.

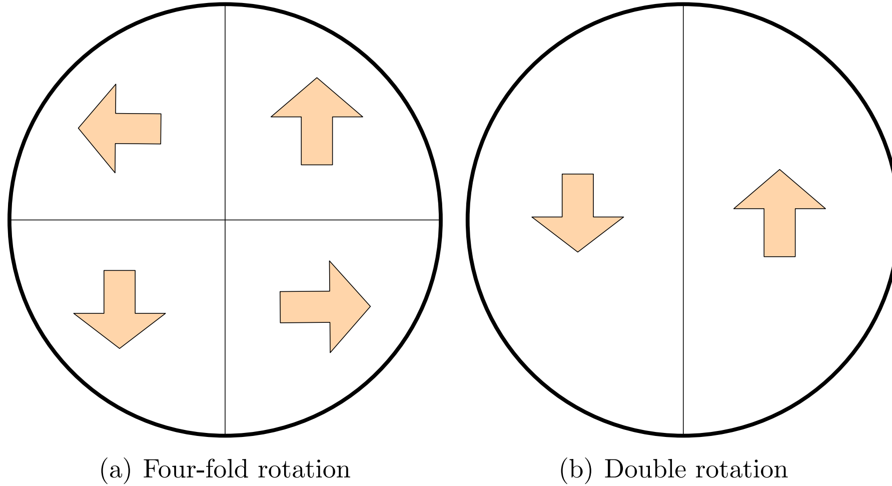



Figure 5.3: Wafer maps showing rotational symmetry.

### 5.3.2 Wafers with Rotational Symmetry

An example of a wafer with rotational symmetry is illustrated in Figure 5.3(a) where the die distribution on the wafer is symmetric with respect to both the horizontal and vertical lines. The die orientation in (a) has 90° difference between adjacent quadrants. If wafers in all repositories have this characteristic, then any pair of wafers drawn from two different repositories can be matched in four ways where one wafer is rotated with respect to the other by 0, 90, 180 or 270 degrees. This virtually enlarges the physical repository size four times. The wafer map introduced in [51] is only capable of such four fold rotation. We also consider wafers capable of two fold rotation. As shown in Figure 5.3(b), the wafer will look identical after each 180° rotation if the die distribution is anti-symmetric across the vertical line, i.e., two halves of the die are oriented with 180° rotation.

### 5.3.3 Running Repository Based Best-pair Matching Algorithm

The running repository scheme is considered in all experiments in this paper since it provably produces higher yield and lower run time complexity than the static repository. Based on such a scheme, the matching algorithm is chosen as the best-pair based algorithm [58, 60] due to its high yield. Thus, wafers from the first two repositories are matched

without any restriction, and the pair producing maximum yield is selected (best-pair match). Then the pair of wafers as a whole is matched with every wafer from the next repository to find the best one (best-one match), and the same process iterates until the last repository. After one complete stack is formed, each repository is replenished immediately. This process is repeated until the production size (total number of stacks fabricated in production) is reached. Note that in the matching algorithm, the matching criterion can produce multiple choices.

### 5.3.4 Matching Criteria

The purpose of wafer matching is to get the maximum final compound yield for a given production size. Given two pre-bond tested wafers, there are basically three criteria to find how well they match [58, 60]: (1) the number of matching good dies (MGD); (2) the number of matching bad dies (MBD); (3) the number of unmatched faulty dies (UFD). An UFD is formed either by a good die overlapping a bad die or a bad die overlapping a good die. Since most publications on wafer matching consider only MGD as the criteria [47, 51, 52, 54, 65] we also use MGD, given that evaluating the best matching criterion is not our focus here.

Wafers are tested prior to bonding. To determine the matching yield of wafer bonding, the state of a tested wafer is represented by a $h \times v$ test matrix of $h$ columns and $v$ rows, where $h$ and $v$ are the maximum number of chips on the wafer along two perpendicular axes termed as horizontal and vertical, respectively. Elements of the test matrix are [0,1] integers. A "1" means a good device and "0" means a bad or non-existing device. Thus, the sum of all elements normalized with respect to the number of device sites on the wafer gives the wafer yield.

When two wafers are stacked, a stacking matrix for the wafer stack is another $h \times v$ matrix whose elements are products of the corresponding elements of test matrices of wafers. The stacking matrix assumes an ideal stacking, i.e., two good devices produce a good stack. It provides the stacking yield in the same way as the test matrix of a wafer gives the wafer

yield. Adding wafers to a partial stack combines test matrices of wafers with the stacking matrix of the previous stack in a similar way. Depending on the manufacturing procedure, whenever a complete or partial stack is tested, the stacking matrix is converted into a test matrix by changing the entries for failed stacks to “0”.

### 5.3.5 A Hybrid Wafer-on-Wafer Stacking Procedure

Based on previous work, we propose a hybrid wafer-on-wafer stacking procedure, which incorporates the rotational symmetry of wafers [51] and running repository based best-pair matching algorithm [58, 60]. This procedure combines the merits of several practices shown in Figure 5.1.

It has been proven [51] that by simple rotation the compound yield can be improved. The reason is quite straightforward: each rotation of a symmetric wafer actually produces a new wafer map, and the repository size is virtually enlarged by as many times as the wafer can be rotated (Figure 5.3). Therefore, we choose a rotationally symmetric wafer in this work. We further select the running repository replenishment scheme and a best-pair matching algorithm to construct a hybrid procedure. We evaluate this hybrid procedure for various defect distribution models.

The initial expectation from this hybrid stacking procedure was that it would produce a considerable compound yield improvement. However, detailed experimental results in Section 5.5 actually show only trivial improvement. Nevertheless, the hybrid procedure serves as a reference for comparison to the work in the next section which is the core contribution of this paper.

## 5.4 Sector Symmetry and Cut for Yield Improvement

The hybrid procedure does not adequately overcome the restrictions of flexibility in matching good dies in wafer stacking. In this section, a novel manipulation scheme of *sector symmetry and cut* is presented to help ease such restrictions.

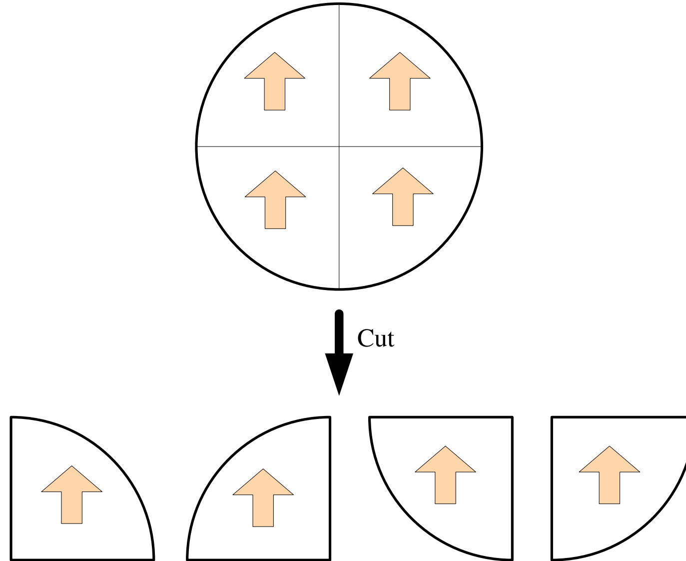



Figure 5.4: A conventional wafer cut into four sectors.

### 5.4.1 Wafers Cut into Sectors

Compared with just rotating the wafer, a more flexible manipulation is to cut each individual wafer into several sectors (called subwafers). If all wafers can be cut to subwafers, then a subwafer can match with any subwafer cut from the same wafer location in another repository. Previously, all subwafers of a wafer were kept together (uncut) during wafer matching. By cutting the wafer, a sector from one wafer can be matched with another sector of another wafer. Figure 5.4 shows four 90° sectors cut from a conventional wafer where the arrow indicates the die orientation within a sector. Similarly, we can cut the wafer into halves (180° sectors) or any number of sectors.

Cutting the wafer into sectors offers an adaptive method between wafer-on-wafer stacking and die-on-die stacking. Comparing with die-on-die stacking, the throughput is largely increased because now each stack produces a sector of 3D ICs. Comparing with wafer-on-wafer-stacking, the yield should be improved because of reduced restrictions in matching sectors rather than matching wafers.

It is quite obvious that extreme cutting (too many sectors) will start losing the advantage because it will get closer to die-on-die stacking, which has highest yield but has a high

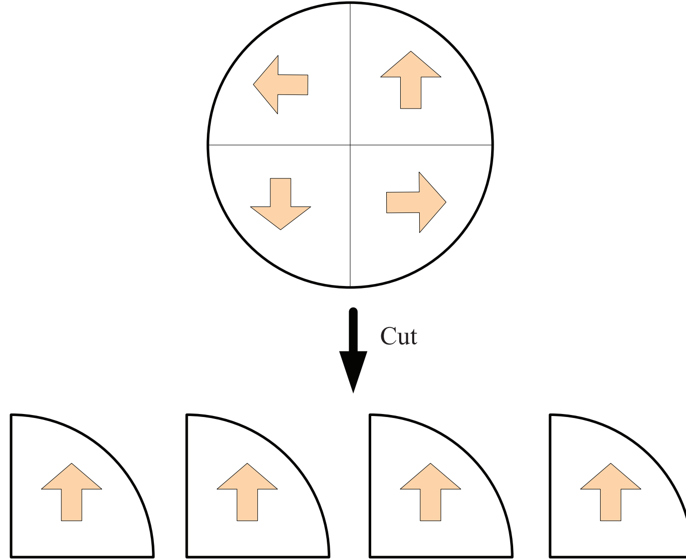



Figure 5.5: Cutting a rotationally symmetric wafer into identical subwafers.

assembly cost. Compared to wafer stacking, a downside of sector stacking is that stacking and bonding of individual sectors requires more effort. Besides, the sector oriented wafer layout causes a loss of chip sites that increases with the number of sectors. With a properly selected sector size, the benefit of matching flexibility, higher yield, and lower cost can outweigh the disadvantages.

### 5.4.2 Sector Symmetry and Cut

After cutting the wafers into subwafers (sectors), each subwafer can only be matched to another subwafer located at the same position within the wafer. For example, the top-left subwafer (second subwafer in Figure 5.4) from repository 1 can only be matched to the top-left subwafer from repository 2. If all subwafers look identical, the restriction due to subwafer location on wafer is eliminated and matching will become more flexible. The idea to obtain identical subwafers from a wafer is straightforward. If subwafers are cut from a wafer fabricated with rotational symmetry, all subwafers will look identical.

Figure 5.5 illustrates the sector symmetry and cut manipulation of the wafer in Figure 5.3(a). Similarly, the wafer can be cut to halves to get two identical subwafers. Now, any subwafer from one repository can be matched to any subwafer from another repository. The

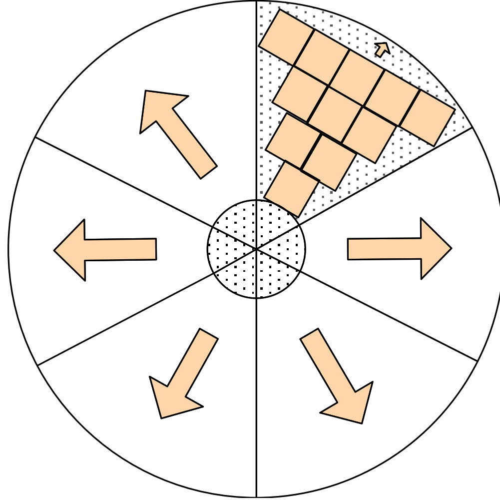


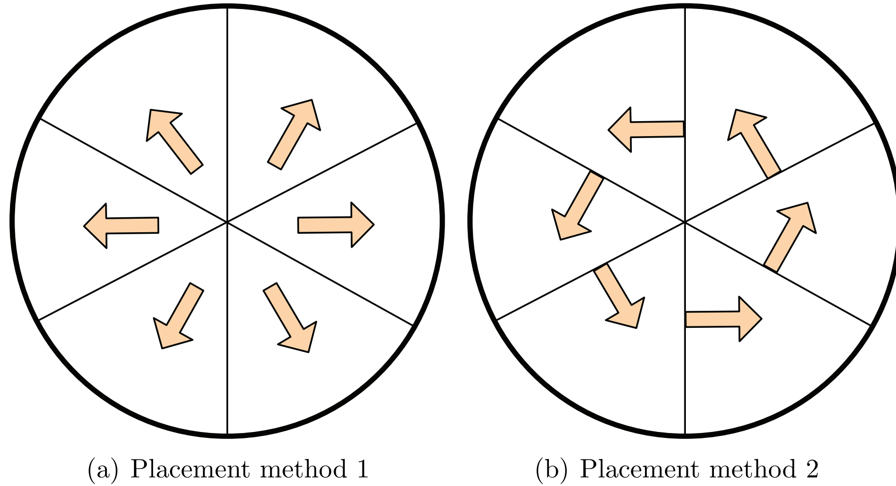
Figure 5.6: Illustration of die loss for cutting the wafer into 6 sectors.

sector symmetry and cut method provides more choices for subwafer stacking in matching algorithms.

### 5.4.3 Discussion on the Number of Cuts

It is natural to think about cutting wafers with rotational symmetry into more sectors than just two or four. However, if a wafer is cut to either three or more than four sectors, new challenges appear. We make two observations. First, dies on the wafer cannot be arranged as compactly, as in the case of two or four sectors. In other words, there will be space wasted at the edges of each sector due to the square or rectangular shape of the chip. Second, cutting a wafer into too many small sectors will generate a circular area of a certain radius, inside which chips cannot be printed, i.e., the area within the circle will be too small to accommodate a complete die.

Figure 5.6 illustrates this point where the wafer is divided into 6 equal sectors. The dotted areas indicate where there is not enough space to accommodate a full die. These areas are either at the edge of the sector or near the center of the wafer. The dotted central area forms a small circle where no single die can be placed within a sector.



(a) Placement method 1 (b) Placement method 2

Figure 5.7: Two different ways of placing dies on a rotationally symmetric wafer.

Table 5.1: Geometrical parameters for dies per wafer (DPW) calculation.

| **Variable** | **Definition** |
|---|---|
| $r$ | Radius of wafer excluding edge clearance |
| $cut_{num}$ | Number of cuts per wafer |
| $H$ | Height of die |
| $L$ | Length or width of die |
| $\alpha$ | Angle of sector |

Thus, cutting a wafer into sectors when the number of sectors is neither two nor four will waste some wafer area and reduce the number of dies per wafer (DPW). Correspondingly, the cost of producing a 3D IC will increase, which must be compensated for by the increased stacking yield.

Rotationally symmetric wafers can use two alternative die placements, as illustrated in Figure 5.7. The two placements yield different DPW. Geometrical parameters used for computing DPW are defined in Table 5.1. Note the vertical and horizontal spacings between dies on the wafer are already included in the die height $H$ and die width $L$.

Figure 5.8 shows a sector with die orientation of Figure 5.7(a). We call this placement method 1. The number of rows $N1_1$ of die that can be placed below the dotted line in Figure 5.8 is computed as,

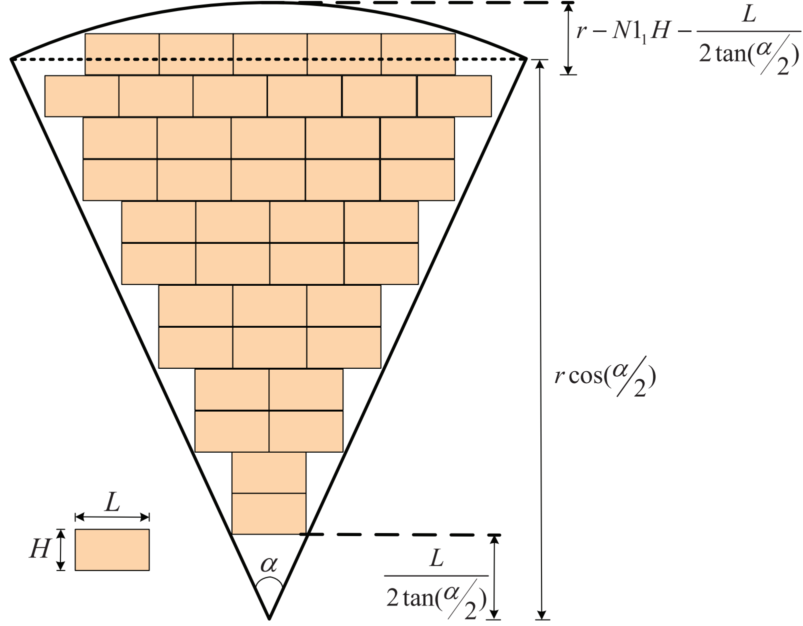



Figure 5.8: Calculation of $DPW1$ for sector placement method 1.

$$N1_1 = \left\lfloor \frac{rcos\frac{\alpha}{2} - \frac{L}{2tan\frac{\alpha}{2}}}{H} \right\rfloor \tag{5.1}$$

Note that the triangle of height $\frac{L}{2tan\frac{\alpha}{2}}$ part at the tip of the sector cannot hold any die. The number of die per sector $DPS1_1$ in $N1_1$ rows is obtained as,

$$DPS1_1 = \sum_{i=1}^{N1_1} \left\lfloor 1 + 2(i-1)\frac{H}{L}tan\frac{\alpha}{2} \right\rfloor \tag{5.2}$$

The number of rows $N1_2$ of die that can be placed above the dotted line in Figure 5.8 is computed as,

$$N1_2 = \left\lfloor \frac{r - N1_1 H - \frac{L}{2tan\frac{\alpha}{2}}}{H} \right\rfloor \tag{5.3}$$

and the number of die per sector $DPS1_2$ accommodated in these $N1_2$ rows is,

$$DPS1_2 = \sum_{i=1}^{N1_2} \left\lfloor \frac{2\sqrt{r^2 - (N1_1 H + iH + \frac{L}{2tan\frac{\alpha}{2}})^2}}{L} \right\rfloor \tag{5.4}$$

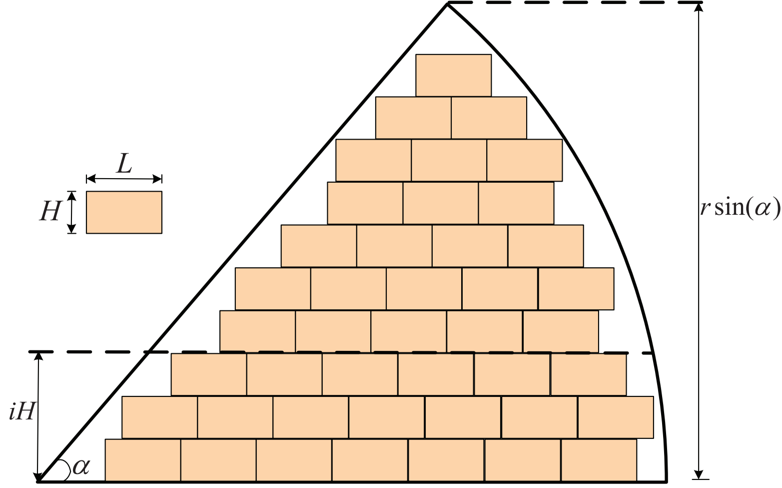



Figure 5.9: Calculation of $DPW2$ for sector placement method 2.

Thus, total number of die per sector $DPS1$ in Figure 5.8 is obtained as,

$$DPS1 = DPS1_1 + DPS1_2 \tag{5.5}$$

Figure 5.9 shows a sector from Figure 5.7(b). We refer to this as placement method 2. The number $N2$ of rows of die that can be placed in this sector is,

$$N2 = \left\lfloor \frac{rsin(\alpha)}{H} \right\rfloor \tag{5.6}$$

A careful examination of method 2 shows that the case for three cuts needs to be examined separately. Die distribution on a sector with two cuts is basically a combination of two sectors from the four cut placement. For four or more cuts, we obtain the number of die per sector as,

$$DPS2 = \sum_{i=1}^{N2} \left\lfloor \frac{\sqrt{r^2 - (iH)^2} - \frac{iH}{tan(\alpha)}}{L} \right\rfloor \tag{5.7}$$

Figure 5.10 shows the die placement of 3-cuts in placement method 2. $N2_1$ and $N2_2$ are the numbers of rows of die that can be placed below and above the dotted line, respectively, in Figure 5.10. Numbers of die per sector $DPS2_1$ and $DPS2_2$ for these sections are computed as follows:

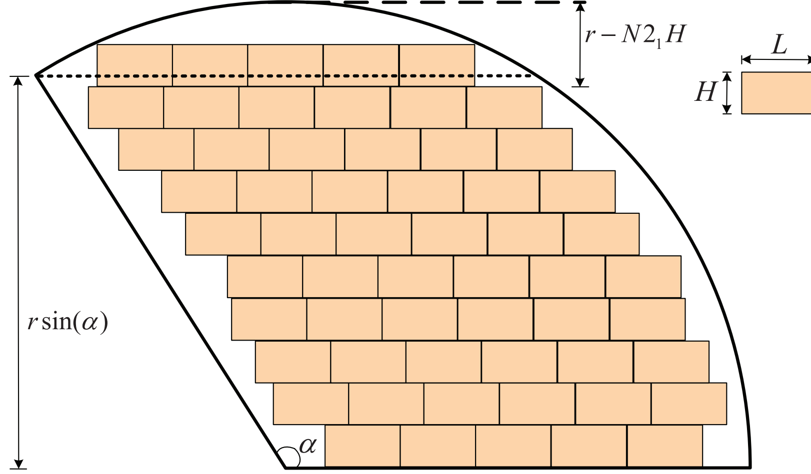



Figure 5.10: Calculation of $DPW2$ of 3-cuts for sector placement method 2.

$$N2_1 = \left\lfloor \frac{rsin(\alpha)}{H} \right\rfloor \tag{5.8}$$

$$DPS2_1 = \sum_{i=1}^{N2_1} \left\lfloor \frac{\sqrt{r^2 - (iH)^2} - \frac{(i-1)H}{cot(\alpha)}}{L} \right\rfloor \tag{5.9}$$

$$N2_2 = \left\lfloor \frac{r - N2_1 H}{H} \right\rfloor \tag{5.10}$$

$$DPS2_2 = \sum_{i=1}^{N2_2} \left\lfloor \frac{2\sqrt{r^2 - [(N2_1 + i)H]^2}}{L} \right\rfloor \tag{5.11}$$

Thus, the total number of die per sector $DPS2$ for method 2 in Figure 5.10 is obtained as,

$$DPS2 = DPS2_1 + DPS2_2 \tag{5.12}$$

The number of die per wafer $DPWq$ for $cut_{num}$ cuts, where $q = 1$ or 2, refers to the placement method 1 or 2, is calculated as,

$$DPWq = DPSq \times cut_{num} \tag{5.13}$$

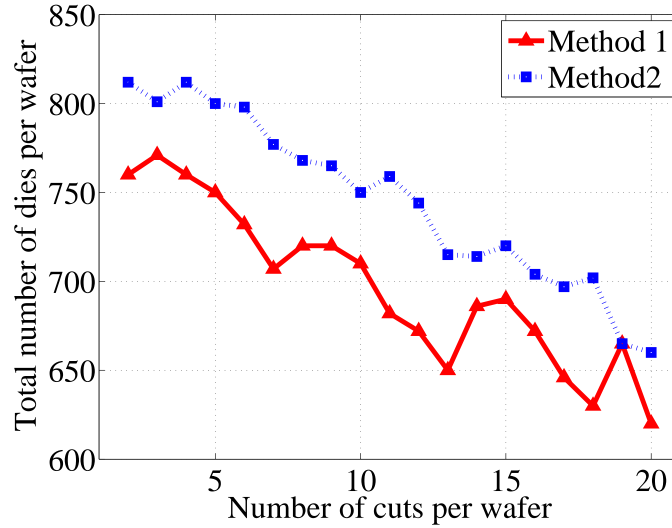



Figure 5.11: $DPW1$ and $DPW2$ versus number of cuts for placement methods 1 and 2.

We consider 8 inch wafers with 5-mm edge clearance and square die of size 31.8 mm $\times$ 31.8 mm. A die spacing of 0.04 mm is assumed. For the selected wafer size and die area, the number of die per wafer is 812 for normal wafers. This number is obtained by using equation 5.7 and 5.13 for 4 cuts in placement method 2.

Figure 5.11 compares the two placement methods for various numbers of cuts. We see a general trend that as the number of cuts increases (larger capability of rotation) the DPW decreases. Also, placement method 2 always outperforms method 1 from the DPW point of view. Actually, through many experiments considering different wafer sizes, die sizes, and chip aspect ratios, we find placement method 2 outperforms method 1 most of the time. That is why we consider placement method 2 in this work. Note that DPW for 2-cuts and 4-cuts with placement method 2 have the DPW of a conventional wafer without cutting. Equations 5.1 to 5.13 are derived for calculating DPW of rotational symmetry wafers. However, like previous work on DPW calculation [19, 66], they can also be applied to DPW calculation of conventional wafers.

### 5.4.4 Summmary

Figure 5.12 shows the complete stacking procedure of the sector symmetry and cut method applied to an example of three stacking levels. Initially all repositories are filled with subwafers. For a given repository size $k$, there will be either $2k$ or $4k$ subwafers within each repository, depending on whether a wafer is cut into two or four pieces. The best-pair match between the first two repositories and the best match for the rest of the repositories are conducted afterwards. Consider for now that the matching is with respect to subwafers instead of wafers. For each repository replenishment, there is a back-up wafer which is cut and rotated. As one subwafer leaves a repository, a new subwafer from the back-up wafer will replenish the repository, immediately. Once the back-up wafer is used up, a new back-up wafer will replace it. Since running repository based best-pair matching algorithm is used in Figure 5.12, the run time complexity is $O(cut_{num} \times k \times p \times n)$ [58, 60] where $cut_{num}$, $k$, $p$ and $n$ are number of cuts, repository size, production size and number of stacked layers, respectively.

We have done extensive Monte Carlo experiments based on different defect models, wafer sizes, and die sizes. The results show that in most cases 4-cuts yield the maximum number of good 3D ICs compared to other numbers of cuts. So a rule-of-thumb is to cut wafer into 4 quadrants. Part of our experimental results are shown in the appendix to illustrate this point. In this work, we emphasize the significance of wafer cut methodology, and only consider cutting wafers into two or four sectors (where no die loss occurs) in the next section. Five types of manipulations of wafers are summarized in Table 5.2.

## 5.5 Experimental Results

### 5.5.1 Experimental Setup

The same wafer and die as in Section 5.4.3 were used in this experiment. Figure 5.2 is used to generate the nine different patterns of wafer maps. If not specified explicitly, a

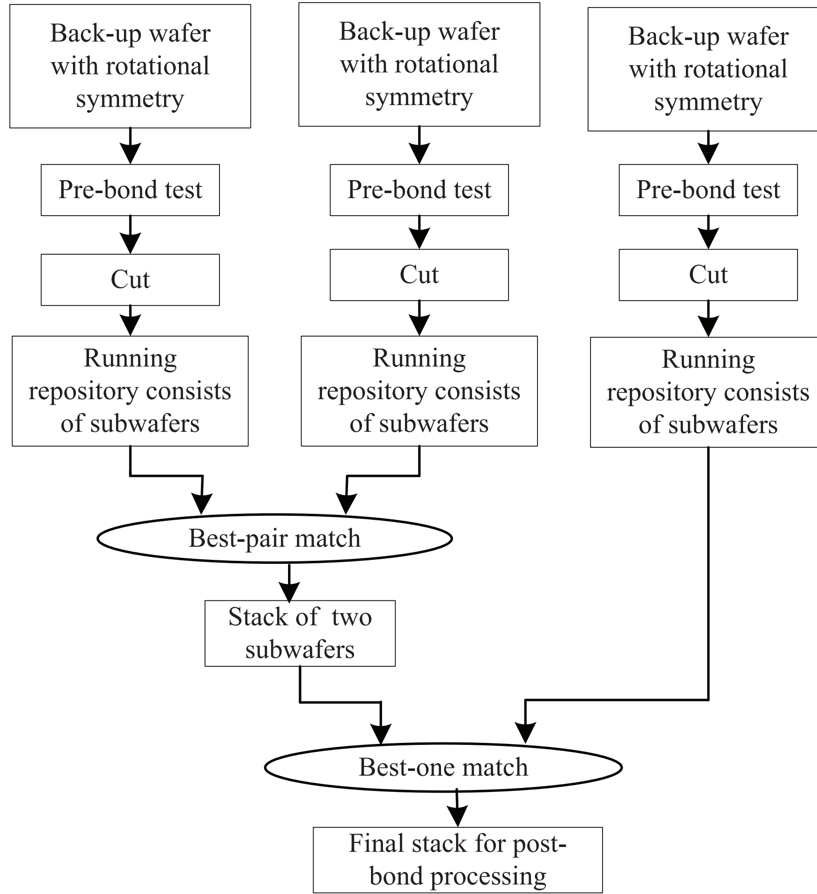



Figure 5.12: Process flow of sector symmetry and cut method.

Table 5.2: Wafer manipulation methods.

| Names | Explanations |
|---|---|
| *Basic* | Wafers without rotational symmetry are matched. |
| *Rotation 4* | Wafers are matched using 4-way rotational symmetry. |
| *Rotation 2* | Wafers are matched using 2-way rotational symmetry. |
| *Sector Symmetry and Cut 4 (SSC4)* | Sectors are matched after 4-way symmetric wafers are cut into 4 sectors. |
| *Sector Symmetry and Cut 2 (SSC2)* | Sectors are matched after 2-way symmetric wafers are cut into 2 sectors. |

default production size of 100,000 3D ICs is targeted in the experiments. All the experiments are repeated 1,000 times and results are averaged to remove noise.

The running repository based best-pair matching algorithm was used [58, 60]. Initially, $k'^2$ ($k' = cut_{num} \times k$) comparisons provide the match information for all wafer (subwafer) pairs from the first two repositories. To speed up the matching algorithm, a heap structure is used to store the match information. Each time a pair of wafers (subwafers) leaves the first two repositories, the corresponding elements are pruned from the heap. As two new wafers (subwafers) enter the first two repositories, their relationships with the existing wafers (subwafers) are constructed and added to the heap. Once the heap is constructed, only $2k'-1$ comparisons are needed each time to replenish the heap.

The five manipulations of Table 5.2 are combined with the running repository based best-pair matching algorithm. The names of these manipulations refer to the complete stacking procedures depending on the context. Recall that the rotation manipulation in Table 5.2 combined with running repository based best pair matching algorithm is the hybrid stacking procedure proposed in Section 5.3.5. Thus, Rotation in this section represents the hybrid procedure.

### 5.5.2 Comparison of Various Stacking Procedures for Different Defect Distributions

In this section we examine the compound yields of final 3D ICs with different stacking procedures under nine different defect distribution models. Initially, the yield of the *basic* procedure with repository size 1 (i.e., random stacking without matching) is calculated for nine types of defect patterns. Subsequently, for each type of pattern, yields for all procedures are normalized with respect to the corresponding random stacking yield. The normalized yield versus repository size for different stacking procedures and defect distributions are shown in Figure 5.13 for three stacked layers.

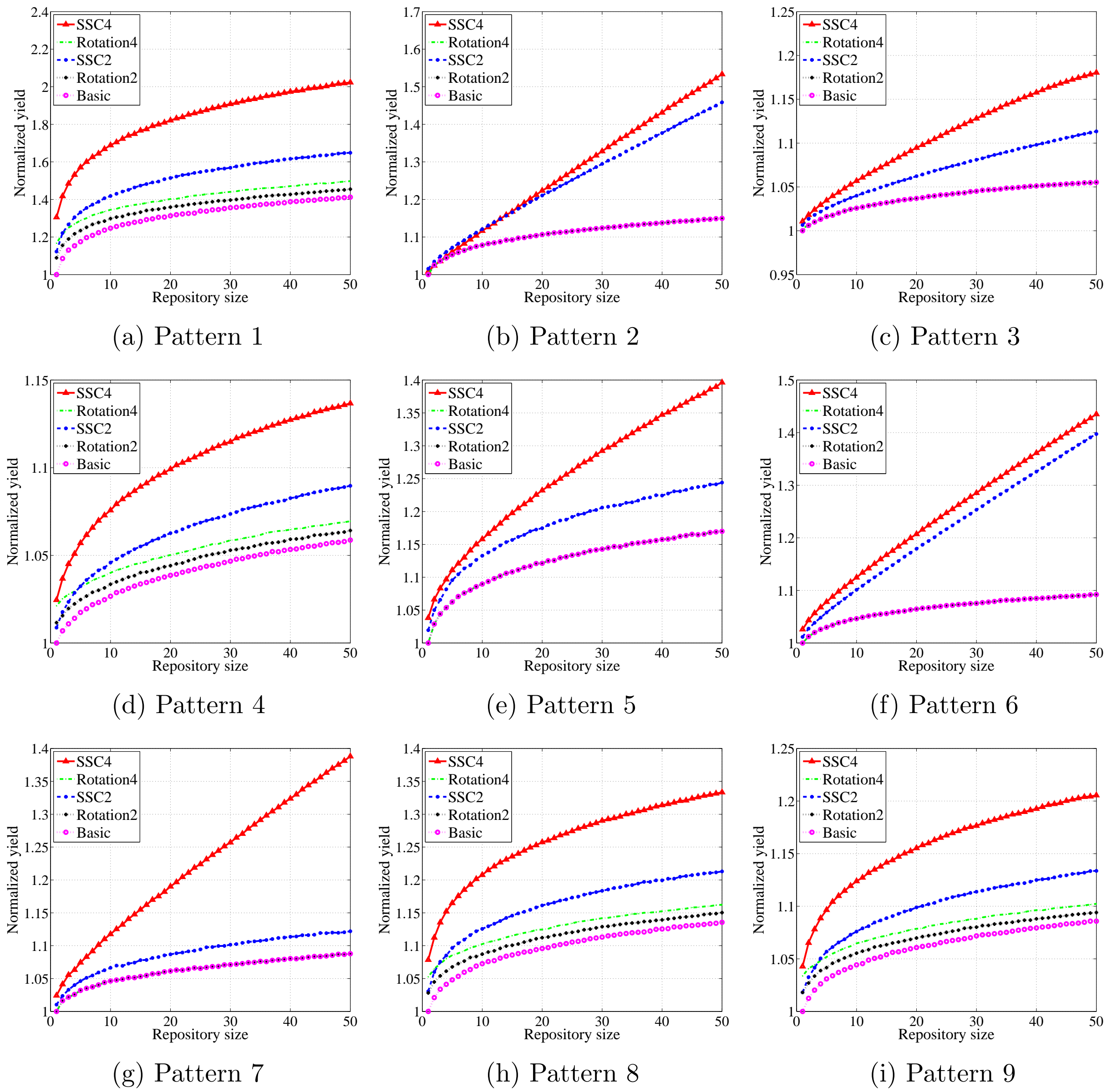



Figure 5.13: Yield improvement by various stacking procedures for different defect distribution patterns of Figure 5.2.

The legends in Figure 5.13 indicate different stacking procedures. For example, ***basic*** (see Table 5.2) means the procedure uses the running repository based best-pair matching algorithm, without any manipulation of wafers. As mentioned in Section 5.5.1, *Rotation* is the hybrid procedure in Figure 5.1.

Next, we compare the performance of different stacking procedures. Regardless of what defect model is used, the yield of the SSC$n$ procedure is always higher than that of others. The reason for superiority of the SSC$n$ procedure is that the restrictions among subwafers are reduced while in Rotation and *basic* all subwafers are bonded together. In SSC$n$, subwafers selected from the same repository are not necessarily from the same wafer. The differences between SSC$n$ and Rotation become more obvious as the repository size grows from 1 to 50. As shown in Figures 5.13, there are up to 50% differences in normalized yield between SSC4 and Rotation 4 when repository size reaches 50.

We evaluate the impact of the number $n$ of cuts on the yield of SSC$n$ procedure. It is obvious from Figure 5.13 that SSC4 always has a higher yield than SSC2. The reason for the yield difference between these two is that in both cases there is no die loss and greater flexibility is provided in SSC4. In SSC4, each wafer is cut into 4 pieces, reducing restrictions between subwafers, and this produces a virtual repository twice the size of the virtual repository of SSC2.

We further evaluate the impact of rotation number $n$ on the yield of proposed hybrid procedure. As can be seen from Figure 5.13, for patterns 1, 4, 8 and 9, the yield of Rotation 4 is better than those for Rotation 2 and *basic*, but the improvement is slight. Why does larger rotation number not help the hybrid procedure significantly? A possible explanation is that under patterns 1, 4, 8 and 9, bad dies are already clustered either at the center or near the edge of wafers, in which case rotating the wafer does little for aligning good dies. For the rest of the patterns, we can see the yield of Rotation and *basic* are the same. To explain this phenomenon, let's re-examine the nine patterns. Of the nine patterns, only four of them (namely, patterns 1, 4, 8, 9) are symmetric about the wafer center while the rest

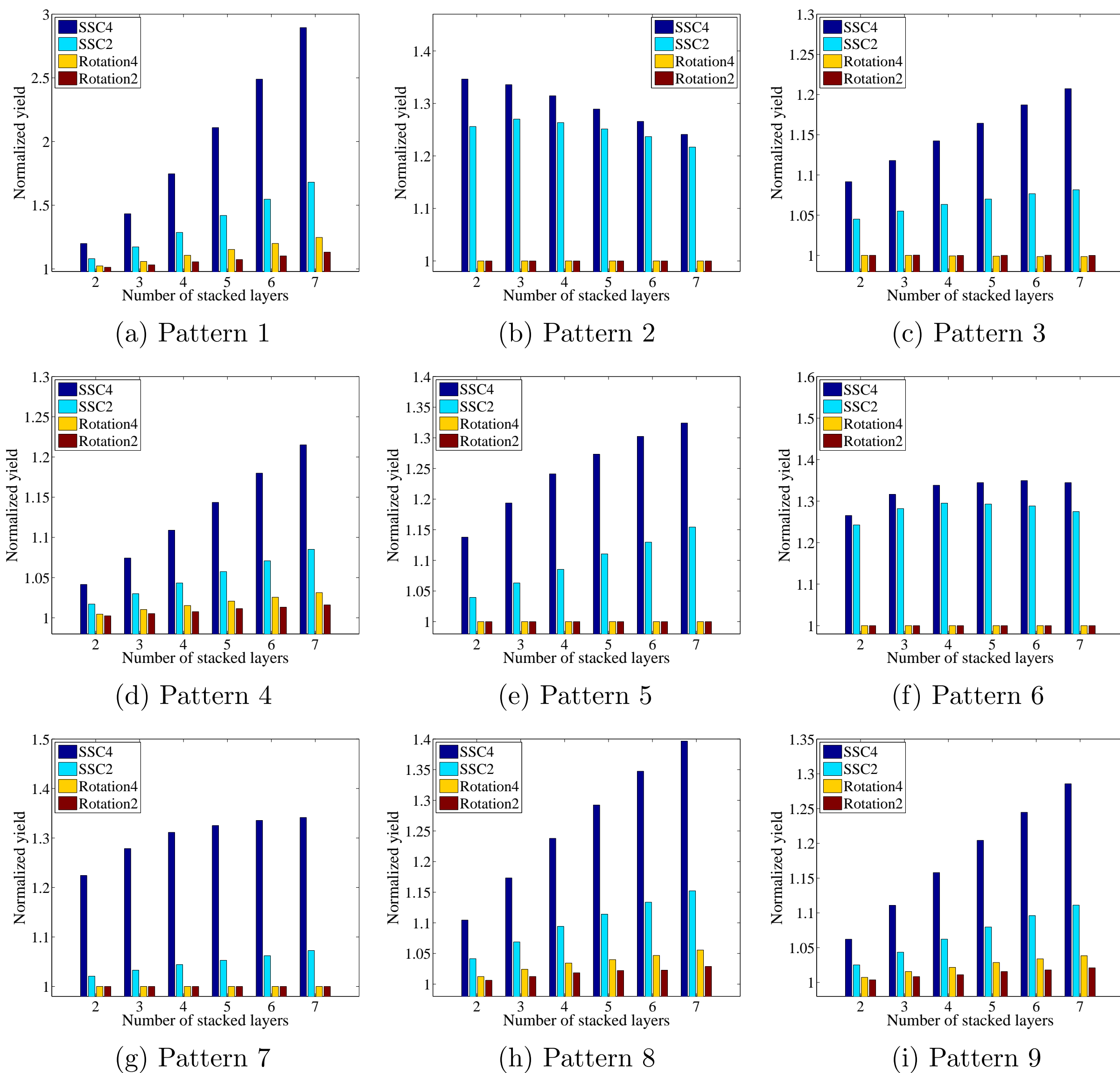
of them are all shifted by some amount. It is obvious that given two wafer maps with the same probabilistically non-symmetric defect distribution, the best way to match them is not to rotate them at all. So even the wafer maps used in our experiments have the capability of four-fold rotation, the rotation method will automatically avoid any rotation. Our observations suggest that for practical wafers with various defect distributions, benefits gained from simple rotation are rather trivial.

Another interesting phenomenon is that the yield for all stacking procedures increases as repository size gets larger. This indicates that a relatively large repository is preferable for yield improvement. The explanation is that larger repository size provides more candidates for matching algorithms thus increasing the compound yield. Considering the extremely small repository with size 1, the wafers are stacked without any freedom for selection. However, larger repository will consume more time in matching algorithms, which correspondingly reduces the throughput.

### 5.5.3 Impact of Number of Stacked Layers on Compound Yield

In this section the impact of number of stacked layers on final compound yield is studied. The experimental results are shown in Figure 5.14, where the $y$ axis indicates normalized yield with respect to the yield of the Basic procedure under the same condition. In Figure 5.14, the repository size is set to 50.

Though not shown in Figure 5.14, the compound yields of all procedures decrease for larger numbers of stacked layers. However, as can be seen, higher improvement is gained for SSC4, SSC2 over Rotation 4, Rotation 2, and *basic*. SSC4 and SSC2 always outperform Rotation 4, Rotation 2 and *basic*, especially for situations where compound yield becomes poorer (Figure 5.14(b) is an exception). For example, in Figure 5.14(a), for 7-level stacks the normalized yield increases from 1.00 for *basic* [52, 58, 60] and 1.25 for Rotation 4 to almost 2.89 for SSC4, which indicates 189% and 131% relative increases, respectively. Note again,




(a) Pattern 1 (b) Pattern 2 (c) Pattern 3

(d) Pattern 4 (e) Pattern 5 (f) Pattern 6

(g) Pattern 7 (h) Pattern 8 (i) Pattern 9

Figure 5.14: Normalized yield for various stacking methods versus number of stacked layers for different defect distribution patterns of Figure 5.2.

compared with *basic* the rotation procedure does not help at all for patterns 2, 3, 5, 6 and 7, regardless of the number of stacked layers.

### 5.5.4 Impact of Production Size on Compound Yield

Since the running repository scheme is utilized in our work, repository pollution is unavoidable [58, 60]. Figure 5.15 shows how the yield decreases as the production size increases for different types of patterns. Note the $x$-axis indicates the number of wafers consumed for a single layer in production. The repository size is set to 25 and the number of stacked layers is selected as 3. Initially, the yield of the SSC4 procedure using only one wafer per repository is pre-calculated for each type of defect distribution pattern. Then for each defect pattern, the yields for all procedures are normalized with respect to the corresponding pre-calculated values.

In Figure 5.15, as the production size increases, the normalized yield for all procedures decreases and finally stabilizes. Interestingly, though yields of SSC4 and SSC2 still outperform Rotation 4 and *basic*, the yield advantages become less obvious for larger production size, especially for patterns with non-symmetric defect probability distributions. One possible explanation is that pollution is more severe for non-symmetric wafer patterns. In the later phase of the manufacturing process, the repository will be always somehow polluted. However, for symmetric defect patterns, a new incoming sector is more likely to match the rest of the unattractive sectors in the repository. For non-symmetric patterns, it is harder to get alignment of good dies. In other words, the compound yield of the selected best pair will be low. That is why the yield benefits of SSC$n$ drops quickly for non-symmetric patterns.

To effectively eliminate pollution and better utilize the SSC$n$ method, a new mechanism to force the unattractive wafers to leave the repository in a timely manner is needed. To our knowledge, no remedy has been proposed. Some possible solutions to reduce pollution could be:

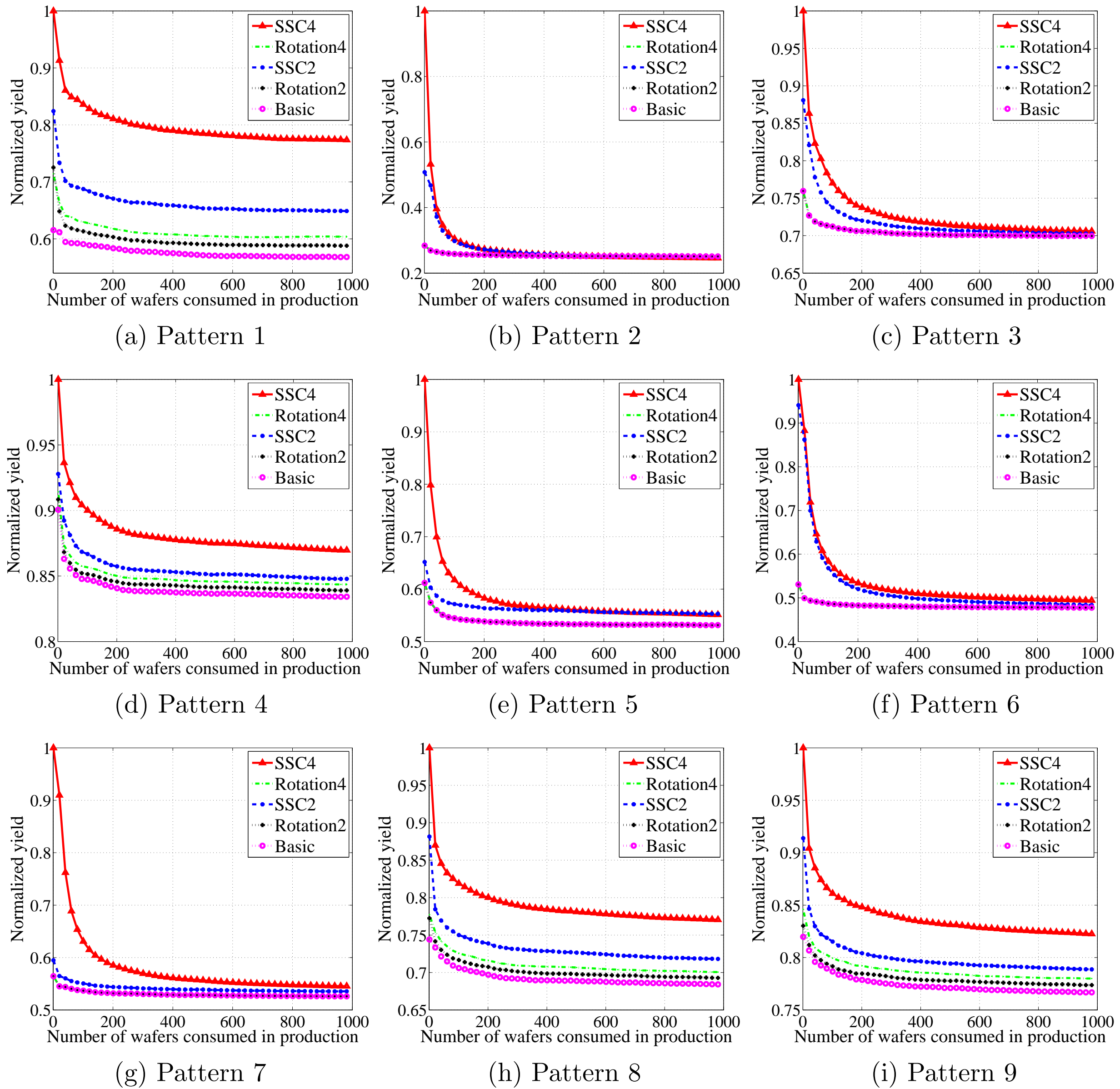



Figure 5.15: Yield reduction for various defect distributions (Figure 5.2) as production size increases.

1) Conduct running repository based matching and static repository based matching, alternatively.

2) Expunge poor wafers (quadrants) from the repository if they have not been used after a certain number of tries. Send them to a die stacking process to make some use of them.

### 5.6 Cost Effectiveness of Sector Symmetry and Cut Method

Previous sections demonstrated the benefits of SSC$n$ method from the aspect of compound yield. However, to decide whether SSC$n$ is applicable, we need to determine the cost-effectiveness of the SSC$n$ method, since SSC$n$ would require extra effort in wafer cutting and bonding, increasing the cost from a manufacturing perspective. The question remains whether the additional cost of wafer cutting and bonding in manufacturing can be compensated for by yield increase? We analyze the cost of a 3D IC in 3 phases: 1) testing, 2) manufacturing, and 3) packaging.

First, we consider the testing cost of a 3D IC. There can be many different kinds of test flows for 3D ICs [59]. We assume an optimized testing flow from [65] to carry out the analysis. This test flow consists of three stages, 1) pre-bond test, 2) post-bond test during which only the newly-formed interconnects are tested, and 3) final test after packaging assumed to cover all interconnects and dies to assure the quality of the final 3D ICs.

Costs of pre-bond test $Cost_{pretest}$, post-bond test $Cost_{postest}$, and final test $Cost_{finaltest}$ are given by equations 5.14 through 5.16. These equations have similar format, i.e., number of items tested multiplied by test cost per item.

$$Cost_{pretest} = DPW \cdot l \cdot t_{die} \tag{5.14}$$

where $t_{die}$ denotes the test cost of a single die. $l$ is the number of stacked layers.

$$Cost_{postest} = DPW \cdot Y(l,k) \cdot (l-1) \cdot t_{int} \tag{5.15}$$

where $Y(l,k)$ is the compound yield for a stack of $l$ layers and a repository size of $k$ wafers. $t_{int}$ is the test cost of the interconnect between two layers.

$$Cost_{finaltest} = DPW \cdot Y(l,k) \cdot Y_{int}^{l-1} \cdot (l \cdot t_{die} + (l-1) \cdot t_{int}) \tag{5.16}$$

where $Y_{int}$ denotes the passing yield of interconnect test between two layers. The total testing cost is:

$$Cost_{test} = Cost_{pretest} + Cost_{postest} + Cost_{finaltest} \tag{5.17}$$

Next, we analyze the impact of wafer cutting on 3D manufacturing cost and find,

$$Cost_{manu} = l \cdot C_w + (l-1) \cdot C_{3D} \cdot (1 + cut_{num} \cdot \beta) \tag{5.18}$$

where $C_w$ is the wafer cost, and $C_{3D}$ [58] is the cost related to 3D stacking process, including through silicon via (TSV), wafer thinning, wafer bonding, etc. A fraction $\beta$ accounts for the overhead of wafer cutting. Larger $\beta$ indicates larger overhead caused by wafer cut and bonding. Note for the *basic* method in which the wafer is not cut before stacking, $cut_{num} = 1$ and $\beta = 0$.

The packaging cost is,

$$Cost_{pack} = DPW \cdot Y(l,k) \cdot Y_{int}^{l-1} \cdot t_{pack} \tag{5.19}$$

where $t_{pack}$ denotes the cost of packaging a 3D IC.

The total cost is the sum of testing cost, manufacturing cost, and packaging cost:

$$Cost_{total} = Cost_{test} + Cost_{manu} + Cost_{pack} \tag{5.20}$$

Notice that the total cost needs to be distributed over all functional 3D ICs. Here, the number of functional 3D ICs after final testing is,

$$Num_{3DIC} = DPW \cdot Y(l,k) \cdot Y_{int}^{l-1} \cdot Y_{die}^{l} \tag{5.21}$$

Therefore, the cost of a single functional 3D IC is,

$$Cost_{3DIC} = \frac{Cost_{total}}{Num_{3DIC}} \tag{5.22}$$

Based on the above cost model, we compare the 3D IC cost of SSC4 and *basic* for nine different wafer patterns. The parameters substituted in the cost model are as follows [9, 58, 65]: $cut_{num} = 4$, $DPW = 812$, $t_{die} = \$0.23$, $t_{int} = \$0.023$, $Y_{die} = 99\%$, $Y_{int} = 97\%$, $C_w = \$1000$, $t_{pack} = \frac{C_w}{DPW} \cdot 0.5$.

Table 5.3 shows the cost analysis results. All positive numbers are in **boldface**, which indicate the percentage of cost improvement with SSC4 over *basic*. Negative numbers indicate cases where cost of SSC4 is higher. We make three observations from Table 5.3. First, given a certain $\beta$ and number of stacking layers ($l$), the cost improvement of SSC4 increases as $\frac{C_{3D}}{C_w}$ decreases. This is because the negative impact of SSC4 on manufacturing cost becomes smaller as $\frac{C_{3D}}{C_w}$ decreases. Second, given a certain $\frac{C_{3D}}{C_w}$ and number of stacking layers, the cost improvement of SSC4 increases as $\beta$ decreases, which is also evident in equation 5.18. These two observations suggest that the cost benefits of SSC4 become larger as the cost overhead of SSC4 become smaller. As the infrastructure of handling sectors of wafers in 3D manufacturing becomes mature, both $\frac{C_{3D}}{C_w}$ and $\beta$ will decrease, and reduced manufacturing overhead of SSC4 can be expected. Third, given certain $\beta$ and $\frac{C_{3D}}{C_w}$, the cost improvement

becomes more significant as the number of layers ($l$) increases. This is because the yield improvement of SSC4 over *basic* becomes much larger when $l$ increases as indicated by Figure 5.14. At large $l$, the final number of good 3D ICs is much larger, thus compensating for the larger manufacturing overhead of SSC4 over *basic*.

As can seen from Table 5.3, for most defect distributions, SSC4 behaves very well even for very large $\frac{C_{3D}}{C_w}$ and $\beta$. Note that $\beta = 0.25$ indicates 100% 3D manufacturing overhead of SSC4. For defect distributions 3, 4, and 9, there is a larger portion of negative numbers when $\frac{C_{3D}}{C_w} = 0.9$, which is of course the worst case condition. But as 3D technology matures, we expect smaller $\frac{C_{3D}}{C_w}$ and $\beta$ in which case SSC4 is more cost-effective.

Table 5.3: Cost improvement percentage for SSC4 over *basic* under various defect distributions (Figure 5.2) and for number of staking layers ($l$) ranging from 2 to 6.

| Defect pattern | Overhead $\beta$ | $\frac{C_{3D}}{C_w} = 0.3$ | | | | | $\frac{C_{3D}}{C_w} = 0.6$ | | | | | $\frac{C_{3D}}{C_w} = 0.9$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $l = 2$ | 3 | 4 | 5 | 6 | 2 | 3 | 4 | 5 | 6 | 2 | 3 | 4 | 5 | 6 |
| Pattern 1 | 0.05 | 14.1 | 27.7 | 40.6 | 50.8 | 58.3 | 12.7 | 26.3 | 39.3 | 49.7 | 57.3 | 11.6 | 25.2 | 38.3 | 48.8 | 56.6 |
| | 0.10 | 12.3 | 25.8 | 38.8 | 49.2 | 56.9 | 9.5 | 22.8 | 36.2 | 46.9 | 54.9 | 7.2 | 20.6 | 34.2 | 45.3 | 53.5 |
| | 0.15 | 10.5 | 23.8 | 37.0 | 47.6 | 55.5 | 6.3 | 19.4 | 33.1 | 44.2 | 52.6 | 2.8 | 15.9 | 30.1 | 41.7 | 50.4 |
| | 0.20 | 8.8 | 21.8 | 35.1 | 46.0 | 54.1 | 3.0 | 15.9 | 29.9 | 41.5 | 50.2 | −1.7 | 11.3 | 26.0 | 38.1 | 47.3 |
| | 0.25 | 7.0 | 19.8 | 33.3 | 44.4 | 52.7 | −0.2 | 12.4 | 26.8 | 38.8 | 47.8 | −6.1 | 6.7 | 21.8 | 34.6 | 44.2 |
| Pattern 2 | 0.05 | 22.8 | 22.3 | 21.1 | 19.5 | 18.0 | 21.7 | 20.8 | 19.4 | 17.7 | 16.1 | 20.7 | 19.6 | 18.1 | 16.4 | 14.7 |
| | 0.10 | 21.3 | 20.2 | 18.7 | 16.9 | 15.3 | 18.8 | 17.1 | 15.3 | 13.3 | 11.5 | 16.8 | 14.7 | 12.7 | 10.6 | 8.7 |
| | 0.15 | 19.7 | 18.1 | 16.3 | 14.4 | 12.6 | 16.0 | 13.5 | 11.2 | 8.9 | 6.9 | 12.9 | 9.8 | 7.3 | 4.8 | 2.6 |
| | 0.20 | 18.1 | 16.0 | 13.9 | 11.8 | 9.9 | 13.1 | 9.8 | 7.1 | 4.5 | 2.2 | 9.0 | 4.9 | 1.8 | −1.0 | −3.4 |
| | 0.25 | 16.5 | 13.9 | 11.6 | 9.2 | 7.2 | 10.2 | 6.1 | 3.0 | 0.1 | −2.4 | 5.1 | 0.0 | −3.6 | −6.8 | −9.5 |
| Pattern 3 | 0.05 | 5.4 | 7.1 | 9.0 | 10.5 | 12.1 | 4.0 | 5.5 | 7.1 | 8.7 | 10.3 | 2.9 | 4.2 | 5.7 | 7.3 | 8.9 |
| | 0.10 | 3.6 | 4.8 | 6.3 | 7.8 | 9.4 | 0.9 | 1.4 | 2.6 | 4.0 | 5.5 | −1.5 | −1.3 | −0.2 | 1.1 | 2.6 |
| | 0.15 | 1.9 | 2.5 | 3.8 | 5.1 | 6.6 | −2.3 | −2.7 | −1.8 | −0.7 | 0.7 | −5.9 | −6.8 | −6.2 | −5.1 | −3.7 |
| | 0.20 | 0.1 | 0.2 | 1.2 | 2.4 | 3.8 | −5.5 | −6.8 | −6.3 | −5.4 | −4.0 | −10.2 | −12.3 | −12.1 | −11.3 | −9.9 |
| | 0.25 | −1.6 | −2.1 | −1.4 | −0.3 | 1.1 | −8.7 | −10.9 | −10.8 | −10.0 | −8.8 | −14.6 | −17.9 | −18.1 | −17.4 | −16.2 |
| Pattern 4 | 0.05 | 1.5 | 3.7 | 6.4 | 9.0 | 11.6 | 0.1 | 2.0 | 4.5 | 7.1 | 9.7 | −1.1 | 0.6 | 3.1 | 5.6 | 8.3 |
| | 0.10 | −0.3 | 1.4 | 3.7 | 6.2 | 8.8 | −3.2 | −2.2 | −0.1 | 2.3 | 5.0 | −5.6 | −5.1 | −3.0 | −0.6 | 2.0 |
| | 0.15 | −2.1 | −1.0 | 1.1 | 3.5 | 6.0 | −6.4 | −6.4 | −4.7 | −2.4 | 0.2 | −10.1 | −10.7 | −9.1 | −6.9 | −4.3 |
| | 0.20 | −3.8 | −3.4 | −1.5 | 0.7 | 3.3 | −9.7 | −10.6 | −9.3 | −7.1 | −4.6 | −14.6 | −16.4 | −15.2 | −13.1 | −10.5 |
| | 0.25 | −5.6 | −5.7 | −4.2 | −2.0 | 0.5 | −13.0 | −14.8 | −13.8 | −11.9 | −9.4 | −19.2 | −22.1 | −21.3 | −19.4 | −16.8 |
| Pattern 5 | 0.05 | 9.5 | 13.4 | 16.5 | 18.5 | 20.3 | 8.1 | 11.7 | 14.7 | 16.7 | 18.5 | 6.9 | 10.3 | 13.4 | 15.4 | 17.1 |
| | 0.10 | 7.7 | 11.0 | 14.0 | 15.9 | 17.7 | 4.8 | 7.6 | 10.4 | 12.3 | 14.0 | 2.3 | 4.8 | 7.6 | 9.5 | 11.2 |
| | 0.15 | 5.8 | 8.7 | 11.5 | 13.3 | 15.0 | 1.4 | 3.5 | 6.0 | 7.8 | 9.5 | −2.3 | −0.6 | 1.9 | 3.6 | 5.3 |
| | 0.20 | 4.0 | 6.3 | 9.0 | 10.7 | 12.4 | −2.0 | −0.7 | 1.7 | 3.3 | 5.0 | −6.9 | −6.1 | −3.9 | −2.2 | −0.6 |
| | 0.25 | 2.1 | 4.0 | 6.5 | 8.1 | 9.7 | −5.3 | −4.8 | −2.7 | −1.1 | 0.5 | −11.4 | −11.6 | −9.6 | −8.1 | −6.5 |

Table 5.3 – Continued.

| Defect pattern | Overhead $\beta$ | $\frac{C_{3D}}{C_w} = 0.3$ | | | | | $\frac{C_{3D}}{C_w} = 0.6$ | | | | | $\frac{C_{3D}}{C_w} = 0.9$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $l = 2$ | 3 | 4 | 5 | 6 | 2 | 3 | 4 | 5 | 6 | 2 | 3 | 4 | 5 | 6 |
| Pattern 6 | 0.05 | 17.2 | 20.4 | 21.8 | 22.3 | 22.6 | 16.1 | 19.0 | 20.3 | 20.7 | 21.0 | 15.2 | 17.9 | 19.1 | 19.5 | 19.7 |
| | 0.10 | 15.6 | 18.3 | 19.5 | 19.9 | 20.1 | 13.2 | 15.4 | 16.4 | 16.5 | 16.7 | 11.2 | 13.1 | 13.9 | 14.0 | 14.1 |
| | 0.15 | 14.0 | 16.3 | 17.3 | 17.5 | 17.6 | 10.3 | 11.8 | 12.4 | 12.4 | 12.4 | 7.2 | 8.3 | 8.7 | 8.5 | 8.5 |
| | 0.20 | 12.4 | 14.3 | 15.0 | 15.0 | 15.1 | 7.4 | 8.2 | 8.5 | 8.2 | 8.1 | 3.3 | 3.4 | 3.5 | 3.1 | 2.9 |
| | 0.25 | 10.8 | 12.2 | 12.7 | 12.6 | 12.6 | 4.5 | 4.6 | 4.5 | 4.1 | 3.8 | −0.7 | −1.4 | −1.8 | −2.4 | −2.7 |
| Pattern 7 | 0.05 | 14.5 | 18.1 | 20.2 | 21.1 | 21.8 | 13.4 | 16.7 | 18.6 | 19.5 | 20.1 | 12.4 | 15.6 | 17.5 | 18.2 | 18.8 |
| | 0.10 | 12.9 | 16.0 | 17.9 | 18.6 | 19.2 | 10.4 | 13.0 | 14.6 | 15.3 | 15.8 | 8.4 | 10.6 | 12.2 | 12.7 | 13.2 |
| | 0.15 | 11.3 | 13.9 | 15.6 | 16.2 | 16.7 | 7.5 | 9.3 | 10.6 | 11.1 | 11.5 | 4.4 | 5.7 | 6.9 | 7.2 | 7.5 |
| | 0.20 | 9.7 | 11.8 | 13.3 | 13.8 | 14.2 | 4.6 | 5.6 | 6.7 | 6.9 | 7.2 | 0.3 | 0.7 | 1.6 | 1.7 | 1.9 |
| | 0.25 | 8.0 | 9.7 | 11.0 | 11.3 | 11.7 | 1.6 | 2.0 | 2.7 | 2.7 | 2.9 | −3.7 | −4.2 | −3.7 | −3.8 | −3.8 |
| Pattern 8 | 0.05 | 6.9 | 11.8 | 16.2 | 19.6 | 22.8 | 5.4 | 10.1 | 14.4 | 17.8 | 21.1 | 4.2 | 8.7 | 13.1 | 16.5 | 19.8 |
| | 0.10 | 5.0 | 9.4 | 13.7 | 17.0 | 20.3 | 2.0 | 5.9 | 10.1 | 13.4 | 16.7 | −0.5 | 3.2 | 7.3 | 10.7 | 14.1 |
| | 0.15 | 3.1 | 7.1 | 11.2 | 14.5 | 17.7 | −1.4 | 1.8 | 5.8 | 9.1 | 12.4 | −5.1 | −2.3 | 1.6 | 5.0 | 8.4 |
| | 0.20 | 1.2 | 4.7 | 8.7 | 11.9 | 15.2 | −4.8 | −2.3 | 1.4 | 4.7 | 8.1 | −9.8 | −7.9 | −4.1 | −0.7 | 2.8 |
| | 0.25 | −0.6 | 2.4 | 6.2 | 9.4 | 12.7 | −8.2 | −6.5 | −2.9 | 0.3 | 3.7 | −14.4 | −13.4 | −9.8 | −6.5 | −2.9 |
| Pattern 9 | 0.05 | 3.2 | 6.8 | 10.2 | 13.5 | 16.3 | 1.8 | 5.0 | 8.4 | 11.7 | 14.4 | 0.6 | 3.7 | 7.0 | 10.3 | 13.1 |
| | 0.10 | 1.4 | 4.4 | 7.7 | 10.9 | 13.6 | −1.5 | 0.8 | 3.9 | 7.1 | 9.8 | −4.0 | −2.0 | 1.1 | 4.3 | 7.0 |
| | 0.15 | −0.4 | 2.0 | 5.1 | 8.2 | 10.9 | −4.9 | −3.4 | −0.6 | 2.5 | 5.2 | −8.6 | −7.6 | −4.9 | −1.8 | 1.0 |
| | 0.20 | −2.3 | −0.4 | 2.5 | 5.5 | 8.2 | −8.2 | −7.6 | −5.1 | −2.1 | 0.6 | −13.2 | −13.3 | −10.9 | −7.8 | −5.1 |
| | 0.25 | −4.1 | −2.8 | −0.1 | 2.9 | 5.5 | −11.5 | −11.8 | −9.6 | −6.7 | −4.0 | −17.8 | −18.9 | −16.9 | −13.9 | −11.2 |

## 5.7 Conclusion

This chapter deals with the problem of low compound yield in wafer-on-wafer stacking. We propose a manipulation method involving sector symmetry and cut (SSC$n$). In this manipulation method, each wafer is cut into $n$ identical sectors that are used to replenish the repository for matching. By wafer cut, the matching restrictions for dies on a wafer are reduced and correspondingly the compound yield is improved. Extensive experiments are conducted to compare the compound yield of the proposed hybrid and SSC$n$ procedures with existing works under various defect distributions. It is demonstrated that the SSC$n$ procedure improves the compound yield significantly irrespective of the type of defect distribution.

We derive mathematical formulas for DPS and DPW calculation for rotationally symmetric wafers. We find greater flexibility of wafer matching by sector symmetry and cut, which on the other hand induces larger die loss in turn reducing the total number of final good 3D ICs. Based on experiments, we conclude that SSC4 should be a rule-of-thumb in practice to maximize the benefit of the proposed technique. A cost model of 3D IC manufacturing is constructed and cost-effectiveness of SSC$n$ is analyzed. It is demonstrated that SSC4 largely reduces the 3D IC cost under various defect models, especially for situations where the number of stacked layers is large. As 3D technology reaches maturity, even larger cost benefits of SSC4 may be expected.

# Chapter 6

# Conclusions and Future Work

This dissertation mainly focuses on the topics of speeding up pre-bond TSV test and improving the compound yield of wafer-on-wafer stacked 3D ICs. The emerging IEEE P1838 standard supports all other test moments except for pre-bond TSV test, which makes pre-bond TSV test very challenging and important. Chapters 2, 3, and 4 form a complete piece of work on speeding up pre-bond TSV test. The fabrication of a 3D IC using wafer-on-wafer stacking has its irreplaceable advantages and is widely used in memory on memory stacking. Thus, chapter 5 focuses on how to improve wafer-on-wafer stacking yield and reduce the cost of 3D wafer-on-wafer stacked ICs.

Chapter 1 gives the reader a broad view of various existing 3D technologies, 3D IC fabrication process, 3D IC test moments, test solutions, and challenges. In chapter 1, we mainly focus on three topics. The first topic is about TSV, including its fabrication process, possible defects, and the electrical models of both normal and defective TSVs. The introduction of TSV characteristics makes the illustration of the pre-bond TSV probing technology clearer. The second topic is on introducing the state-of-the-art TSV probing technique, which serves as the basis for our work presented in chapters 2, 3 and 4. The third topic is on the developing IEEE P1838 standard. We introduce the standard for two reasons. First, it gives the reader the pre-knowledge of the up-to-date test solutions for 3D ICs. The other reason is the pre-bond TSV probing technique is actually compatible with the standard, utilizing boundary scan registers to drive TSVs during probing.

In chapter 2, we proposed an ILP model to generate near-optimal set of test sessions for pre-bond TSV probing. There are advantages of our ILP model over existing heuristic methods. First, the total test time of all the sessions is always less for the ILP model.

Second, the total number of generated test sessions is smaller in most of the cases. For cases where the number of sessions is the same as that of heuristic method, we demonstrated that the total test time is still much less. Third, for various TSV networks with different parameters, the test time reduction of the ILP model remain pretty consistent, and thus eliminates the need for separately designing and optimizing the test for each TSV network as required by previous work. There is still space for future work on test session generation, such as possibly finding a necessary and sufficient condition to generate globally optimal set of sessions.

The ILP model in chapter 2 constructs test sessions but it doesn't provide any information on how to identify faulty TSVs based on the sessions. In chapter 3, we proposed a fast TSV identification algorithm which actually identifies the faulty TSVs based on given test sessions. This algorithm speeds up pre-bond TSV probing from two aspects. First, any unnecessary session during the test is skipped. Second, the test terminates as soon as either all TSVs have been identified or a pre-specified maximum number of faulty TSVs have been identified. Extensive experiments are done, and the benefits of the algorithm are explained in detail.

In chapter 4, we first proposed a session sorting procedure to sequence test sessions in such a way that the pre-bond TSV test can terminate as soon as possible for small number of faulty TSVs within a network. The motivation of our proposal is based on the observation that TSV yield is relatively high in practice and the probability of small numbers of faulty TSVs (less than 2) within a network approaches 100%. After introducing the session-sorting procedure, we combine it with the work presented in chapter 2 and 3, and further propose a 3-Step test time Optimization Simulator (SOS3). In SOS3, the ILP model in chapter 2 is first used to generate a series of test sessions with certain fault identification capability. Then, these test sessions are sorted to reduce the expectation of test time. Lastly, the fast TSV identification algorithm is used for early test termination. SOS3 as a framework is

demonstrated to greatly reduce pre-bond faulty TSV identification time. SOS3 is expected to greatly reduce pre-bond TSV test cost in real silicon.

The work on pre-bond TSV testing provides necessary known good die information for wafer-on-wafer stacking, which is the topic of chapter 5. Chapter 5 proposed to design rotationally symmetric wafers and cut each wafer into several identical sub-wafers during wafer matching process. Our proposal, named Sector Symmetry and Cut $n$ (SSC$n$), is demonstrated to largely improve the wafer-on-wafer stacking yield and reduce the 3D IC cost for various defect distributions. Since wafer-on-wafer stacking has its unreplaceable advantages and is widely used in memory on memory stacking, the achieved yield improvement and cost reduction of our work could be pretty significant. Note some future work can still be done. The reported experiments assume that wafers used in the same stack all have the same kind of defect distribution. This may not be the case in practice since wafers from different vendors may be used for 3D stacking. Even for the same manufacturer, the fabricated wafers may have different defect distributions. More experiments are needed to study the compound yield of stacking wafers with different defect distributions. Another direction of future research is to develop a mechanism that can effectively force the unattractive wafers to leave repositories so as to reduce repository pollution. Once the problem of pollution is solved, the SSC$n$ procedure is likely to reveal larger advantages.

# Bibliography


[1] Avialable from http://www.fpgatips.com/xilinx-wins-2013-3d-incites-award/, accessed on May 18, 2014.

[2] Avialable from http://www.samsung.com/global/business/semiconductor/news-events/press-releases/detail?newsId=12990, accessed on May 18, 2014.

[3] CPLEX Optimizer. avialable from http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/, accessed on May 18, 2014.

[4] PTM 45nm Model. available from http://ptm.asu.edu/.

[5] M. Aoki, F. Furuta, K. Hozawa, Y. Hanaoka, H. Kikuchi, A. Yanagisawa, T. Mitsuhashi, and K. Takeda. Fabricating 3D Integrated CMOS Devices by using Wafer Stacking and Via-Last TSV Technologies. In *IEEE International Electron Devices Meeting*, pages 29.5.1–29.5.4, 2013.

[6] R. Beica, C. Sharbono, and T. Ritzdorf. Through Silicon Via Copper Electrodeposition for 3D Integration. In *Proc. 58th Electronic Components and Technology Conference*, pages 577–583, 2008.

[7] B. Black, D. Nelson, C. Webb, and N. Samra. 3D Processing Technology and Its Impact on IA32 Microprocessors. In *Proceeding of International Conference on Computer Design*, pages 316–318, 2004.

[8] S. Borkar. 3D Integration for Energy Efficiency System Design. In *Proceeding of Design Automation Conference*, pages 214–219, 2011.

[9] M. L. Bushnell and V. D. Agrawal. *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits.* Springer, 2000.

[10] H. Chen, J. Shih, S. W. Li, H. C. Lin, M. Wang, and C. Peng. Electrical Tests for Three-Dimensional ICs (3DICs) with TSVs. In *International Test Conference 3D-Test Workshop*, pages 1–6, 2010.

[11] P. Chen, C. Wu, and D. Kwai. On-Chip Testing of Blind and Open-Sleeve TSVs for 3D IC Before Bonding. In *IEEE 28th VLSI Test Symposium*, pages 263–268, 2010.

[12] M. Cho, C. Liu, D. H. Kim, S. K. Lim, and S. Mukhopadhyay. Design Method and Test Structure to Characterize and Repair TSV Defect Induced Signal Degradation in 3D System. In *IEEE/ACM International Conference on Computer-Aided Design*, pages 694–697, 2010.

[13] 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. *IEEE Design & Test of Computers*, 22(6):498–510, 2005.

[14] G. De Nicoao, E. Pasquinetti, G. Miraglia, and F. Piccinini. Unsupervised Spatial Pattern Classification of Electrical Failures in Semiconductor Manufacturing. In *Proc. Artificial Neural Networks Pattern Recognition Workshop*, pages 125–131, 2003.

[15] S. Deutsch and K. Chakrabarty. Non-Invasive Pre-Bond TSV Test using Ring Oscillators and Multiple Voltage Levels. In *Design, Automation & Test in Europe Conference & Exhibition*, pages 1065–1070, 2013.

[16] F. Di Palma, G. De Nicolao, G. Miraglia, E. Pasquinetti, and F. Piccinini. Unsupervised Spatial Pattern Classification of Electrical-wafer-sorting Maps in Semiconductor Manufacturing. *Pattern Recognition Letter*, 26(12):1857–1865, Sept. 2005.

[17] X. Dong and Y. Xie. System-Level Cost Analysis and Design Exploration for Three-dimensional Integrated Circuits (3D ICs). In *Proc. Asia and South Pacific Design Automation Conference*, pages 234–241, 2009.

[18] J. Dukovic et al. Through-Silicon-Via Technology for 3D Integration. In *Proc. IEEE International Memory Workshop*, pages 1–2, 2010.

[19] A. V. Ferris-Prabhu. An Algebraic Expression to Count the Number of Chips on a Wafer. *IEEE Circuits and Devices Magazine*, pages 37–39, 1989.

[20] D. Fick, R. Dreslinski, B. Giridhar, G. Kim, S. Seo, M. Fojtik, S. Satpathy, Y. Lee, D. Kim, N. Liu, M. Wieckowski, G. Chen, T. Mudge, D. Sylvester, and D. Blaauw. Centip3De: A 3930DMIPS/W Configurable Near-threshold 3D Stacked System with 64 ARM Cortex-M3 Cores. In *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pages 190–192, 2012.

[21] S. K. Goel and E. J. Marinissen. Effective and Efficient Test Architecture Design for SOCs. In *Proc. International Test Conference*, pages 529 – 538, 2002.

[22] A. Gupta, W. A. Porter, and J. W. Lathrop. Defect Analysis and Yield Degradation of Integrated Circuits. *IEEE Journal of Solid-State Circuits*, 9(3):96–102, Mar. 1974.

[23] S. Hamdioui and M. Taouil. Yield Improvement and Test Cost Optimization for 3D Stacked ICs. In *Proc. IEEE Asian Test Symposium*, pages 480–485, 2011.

[24] A. Hsieh, T. Hwang, M. Chang, and M. Tsai. TSV Redundancy: Architecture and Design Issues in 3D IC. In *Design, Automation & Test in Europe Conference & Exhibition*, pages 166–171, 2010.

[25] Y. Huang, J. Li, J. Chen, D. Kwai, Y. Chou, and C. Wu. A Built-In Self-Test Scheme for the Post-Bond Test of TSVs in 3D ICs. In *IEEE 29th VLSI Test Symposium*, pages 20–25, 2011.

[26] R. C. Jeager. *Introduction To Microelectronic Fabrication*. Prentice Hall.

[27] L. Jiang, Q. Xu, and B. Eklow. On Effective Through-Silicon Via Repair for 3-D-Stacked ICs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 32(2):559–571, 2013.

[28] M. Jung, J. Mitra, D. Z. Pan, and S. K. Lim. TSV Stress-Aware Full-Chip Mechanical Reliability Analysis and Optimization for 3D IC. In *Proc. 48th Design Automation Conference*, pages 188–193, 2011.

[29] D. H. Kim, K. Athikulwongse, M. Healy, M. Hossain, M. Jung, I. Khorosh, G. Kumar, Y. Lee, D. Lewis, T. Lin, C. Liu, S. Panth, M. Pathak, M. Ren, G. Shen, T. Song, D. H. Woo, X. Zhao, J. Kim, H. Choi, G. Loh, H. Lee, and S. K. Lim. 3D-MAPS: 3D Massively parallel processor with stacked memory. In *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pages 188 – 190, 2012.

[30] H.-H. S. Lee and K. Chak. Test Challenges for 3D Integrated Circuits. *IEEE Design & Test of Computers*, 26(5):26–35, 2009.

[31] H. Liao. Microfabrication of Through Silicon Vias (TSV) for 3D Sip. *Solid-State and Integrated-Circuit Technology*, (1):20–23, Nov. 2008.

[32] H. Liao, M. Miao, X. Wan, Y. Jin, L. Zhao, B. Li, Y. Zhu, and X. Sun. Microfabrication of Through Silicon Vias (TSV) for 3D SiP. In *Proc. 9th International Conference on Solid-State and Integrated-Circuit Technology (ICSICT)*, pages 1199–1202, 2008.

[33] E. J. Marinissen. Challenges and Emerging Solutions in Testing TSV-Based $2\frac{1}{2}$D- and 3D-Stacked ICs. In *Proc. Design, Automation & Test in Europe Conference & Exhibition*, pages 1277–1282, 2012.

[34] E. J. Marinissen, C. C. Chi, J. Verbree, and M. Konijnenburg. 3D DfT Architecture for Pre-Bond and Post-Bond Testing. In *IEEE International 3D Systems Integration Conference*, pages 1–8, 2010.

[35] E. J. Marinissen, S. K. Goel, and M. Lousberg. Wrapper Design for Embedded Core Test. In *Proc. International Test Conference*, pages 911–920, 2000.

[36] E. J. Marinissen, J. Verbree, and M. Konijnenburg. A Structured and Scalable Test Access Architecture for TSV-Based 3D Stacked ICs. In *Proc. 28th IEEE VLSI Test Symposium*, pages 269–274, 2010.

[37] E. J. Marinissen and Y. Zorian. Testing 3D Chips Containing Through-Silicon Vias. In *Proc. International Test Conference*, pages 1–11, 2009.

[38] M. Miao. Process Simulation of DRIE and Its Application in Tapered TSV Fabrication. In *International Conference on Electronic Packaging Technology & High Density Packaging*, pages 28–31, 2008.

[39] B. Noia and K. Chakrabarty. Identification of Defective TSVs in Pre-Bond Testing of 3D ICs. In *Proc. 20th IEEE Asian Test Symposium*, pages 187–194, 2011.

[40] B. Noia and K. Chakrabarty. Pre-Bond Probing of TSVs in 3D Stacked ICs. In *Proc. International Test Conference*, pages 1–10, 2011.

[41] B. Noia and K. Chakrabarty. *Design-for-Test and Test Optimization Techniques for TSV-based 3D Stacked ICs.* Springer, 2014.

[42] B. Noia, S. K. Goel, K. Chakrabarty, E. J. Marinissen, and J. Verbree. Test-Architecture Optimization for TSV-Based 3D Stacked ICs. In *Proc. 15th IEEE European Test Symposium*, pages 24–29, 2010.

[43] D. Z. Pan, S. K. Lim, K. Athikulwongse, M. Jung, J. Mitra, J. S. Pak, M. Pathak, and J. Yang. Design for Manufacturability and Reliability for TSV-Based 3D ICs. In *Proc. 17th Asia and South Pacific Design Automation Conference*, pages 750–755, 2012.

[44] M. Puech. A Novel Plasma Release Process and Super High Aspect Ratio Process Using ICP Etching for MEMS. In *MEMSINEMS seminar*, page Paper 16.3, 2003.

[45] M. Puech, J. M. Thevenoud, J. M. Gruffat, N. Launay, N. Arnal, and P. Godinat. Fabrication of 3D Packaging TSV Using DRIE. In *Proc. Symposium on Design, Test, Integration and Packaging of MEMS/MOEMS*, pages 109–114, 2008.

[46] J. Rajski and J. Tyszer. Fault Diagnosis of TSV-Based Interconnects in 3-D Stacked Designs. In *Proc. International Test Conference*, pages 1–9, 2013.

[47] S. Reda, G. Smith, and L. Smith. Maximizing the Functional Yield of Wafer-to-wafer 3-D Integration. *IEEE Transactions on Very Large Scale Integration Systems*, 17(9):1357–1362, Sept. 2009.

[48] A. Rogers. *Statistical Analysis of Spatial Dispersions.* Pion Limited, United Kingdom, 1974.

[49] S. K. Roy, S. Chatterjee, C. Giri, and H. Rahaman. Faulty TSVs Identification and Recovery in 3D Stacked ICs During Pre-bond Testing. In *Proc. International 3D Systems Integration Conference*, pages 1–6, 2013.

[50] S. Seguin. World's First Stacked 3D Processor Created. Tomshardware.com, accessed on August 29th.

[51] E. Singh. Exploiting Rotational Symmetries for Improved Stacked Yields in W2W 3D-SICs. In *Proc. IEEE 29th VLSI Test Symposium*, pages 32–37, 2011.

[52] E. Singh. Impact of Radial Defect Clustering on 3D Stacked IC Yield From Wafer to Wafer Stacking. In *Proc. International Test Conference*, pages 1–7, 2012.

[53] K. Smith, P. Hanaway, M. Jolley, R. Gleason, and E. Strid. Evaluation of TSV and Micro-Bump Probing for Wide I/O Testing. In *Proc. International test Conference*, pages 1–10, 2011.

[54] L. Smith, G. Smith, S. Hosali, and S. Arkalgud. Yield Considerations in the Choice of 3D Technology. In *Proc. International Symposium on Semiconductor Manufacturing*, pages 1–3, 2007.

[55] C. H. Stapper. On Yield, Fault Distributions, and Clustering of Particles. *IBM Journal of Research and Development*, 30(3):326–338, 1986.

[56] C. H. Stapper. Simulation of Spatial Fault Distributions for Integrated Circuit Yield Estimations. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 8(12):1314–1318, Dec. 1989.

[57] C. H. Stapper, F. M. Armstrong, and K. Saji. Integrated Circuit Yield Statistics. In *Proceedings of the IEEE*, pages 453–470, 1983.

[58] M. Taouil and S. Hamdioui. Yield Improvement for 3D Wafer-to-wafer Stacked Memories. *Journal of Electronic Testing: Theory and Applications*, 28(4):523–534, Aug. 2012.

[59] M. Taouil, S. Hamdioui, K. Beenakker, and E. J. Marinissen. Test Impact on the Overall Die-to-wafer 3D Stacked IC Cost. *Journal of Electronic Testing: Theory and Applications*, 28(1):15–25, Feb. 2012.

[60] M. Taouil, S. Hamdioui, J. Verbree, and E. J. Marinissen. On Maximizing the Compound Yield for 3D Wafer-to-wafer Stacked ICs. In *Proc. IEEE International Test Conference*, pages 1–10, 2010.

[61] D. Teets. A Model for Radial Yield Degradation as a Function of Chip Size. *IEEE Transactions on Semiconductor Manufacturing*, 9(3):467–471, 1996.

[62] I. V., K. Chakrabarty, and E. J. Marinissen. Test Wrapper and Test Access Mechanism Co-Optimization for System-on-Chip. In *Proc. International Test Conference*, pages 1023 – 1032, 2001.

[63] G. Van der Plas et al. Design Issues and Considerations for Low-Cost 3-D TSV IC Technology. *IEEE Journal of Solid-State Circuits*, 46(1):293–307, Jan. 2011.

[64] J. Van Olmen et al. 3D Stacked IC Demonstration Using a Through Silicon Via First Approach. In *Proc. IEEE International Electron Devices Meeting (IEDM)*, pages 303–306, 2008.

[65] J. Verbree, E. J. Marinissen, P. Roussel, and D. Velenis. On the Cost-effectiveness of Matching Repositories of Pre-tested Wafers for Wafer-to-wafer 3D Chip Stacking. In *Proc. 15th IEEE European Test Symposium*, pages 36–41, 2010.

[66] D. K. Vries. Investigation of Gross Die per Wafer Formulas. *IEEE Transactions on Semiconductor Manufacturing*, 18:136–139, 2005.

[67] D. H. Woo, N. H. Seong, D. L. Lewis, and H.-H. S. Lee. An Optimized 3D-Stacked Memory Architecture by Exploiting Excessive, High-Density TSV Bandwidth. In *Proc. 16th IEEE International Symposium on High Performance Computer Architecture*, pages 1–12, 2010.

[68] X. Wu, Y. Chen, K. Chakrabarty, and Y. Xie. Test-Access Mechanism Optimization for Core-Based Three-Dimensional SOCs. In *Proc. IEEE International Conference on Computer Design*, pages 212–218, 2008.

[69] O. Yaglioglu and B. Eldridge. Direct Connection and Testing of TSV and Microbump Devices using NanoPierce Contactor for 3D-IC Integration. In *Proc. 30th IEEE VLSI Test Symposium*, pages 96–101, 2012.

[70] T. Yanagawa. Influence of Epitaxial Mounds on the Yield of Integrated Circuits. *Proceedings of the IEEE*, 57(9):1621–1628, Sept. 1969.

[71] T. Yanagawa. Yield Degradation of Integrated Circuits Due to Spot Defects. *IEEE Transactions on Electron Devices*, 19(2):190–197, 1972.

[72] C. Yang, C. Chou, and J. Li. A TSV Repair Scheme Using Enhanced Test Access Architecture for 3-D ICs. In *Proc. 22nd IEEE Asian Test Symposium*, pages 7–12, 2013.

[73] J. You, H. S., D. Kwai, Y. Chou, and C. Wu. Performance Characterization of TSV in 3D IC via Sensitivity Analysis. In *Proc. 19th IEEE Asian Test Symposium*, pages 389–394, 2010.

[74] B. Zhang and V. D. Agrawal. Wafer Cut and Rotation for Compound Yield Improvement in 3D Wafer-on-wafer Stacking. In *Proc. IEEE North Atlantic Test Workshop*, 2013.

[75] B. Zhang and V. D. Agrawal. A Novel Wafer Manipulation Method for Yield Improvement and Cost Reduction of 3D Wafer-on-Wafer Stacked ICs. *Journal of Electronic Testing: Theory and Applications*, 30:57–75, 2014.

[76] B. Zhang and V. D. Agrawal. An Optimal Probing Method of Pre-Bond TSV Fault Identification for 3D Stacked ICs. In *IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference*, Oct. 2014.

[77] B. Zhang and V. D. Agrawal. An Optimized Diagnostic Procedure for Pre-bond TSV Defects. In *Proc. 32nd IEEE International Conference on Computer Design*, Oct. 2014.

[78] B. Zhang and V. D. Agrawal. Diagnostic Tests for Pre-Bond TSV Defects. In *28th International Conference on VLSI Design*, Jan. 2015.

[79] B. Zhang, B. Li, and V. D. Agrawal. Exploiting Sector-on-Sector Stacking for Yield Improvement of 3D ICs. In *International Test Conference 3D-Test Workshop*, pages 1–6, 2013.

[80] B. Zhang, B. Li, and V. D. Agrawal. Yield Analysis of a Novel Wafer Manipulation Method in 3D Stacking. In *Proc. IEEE International 3D Systems Integration Conference*, pages 1–8, 2013.

[81] Y. Zhao, S. Khursheed, and B. M. Al-Hashimi. Cost-Effective TSV Grouping for Yield Improvement of 3D-ICs. In *Proc. 20th IEEE Asian Test Symposium (ATS)*, pages 201–206, 2011.

# Appendix A

## Impact of Number of Cuts on Final Production Size of Good 3D ICs

Figure A.1 shows the final production size of good 3D ICs considering different number of cuts. Same setup as in Section 5.5.1 applies here. As we can see, in most cases four-cuts produces the largest number of good 3D ICs. Note that 2 cuts are not used in these experiments because DPW of 2 cuts is identical to that for 4 cuts. However, 2-cuts provide less flexibility in matching and will definitely yield fewer good 3D ICs than 4 cuts. Also note that 3 cuts are not used either because the DPW for 3 cuts is lower than that for 4 cuts. Besides, 3 cuts provide less flexibility in wafer matching. More experiments have been done considering different wafer sizes, die sizes, defect models, etc. Since results are similar, they are not duplicated here.

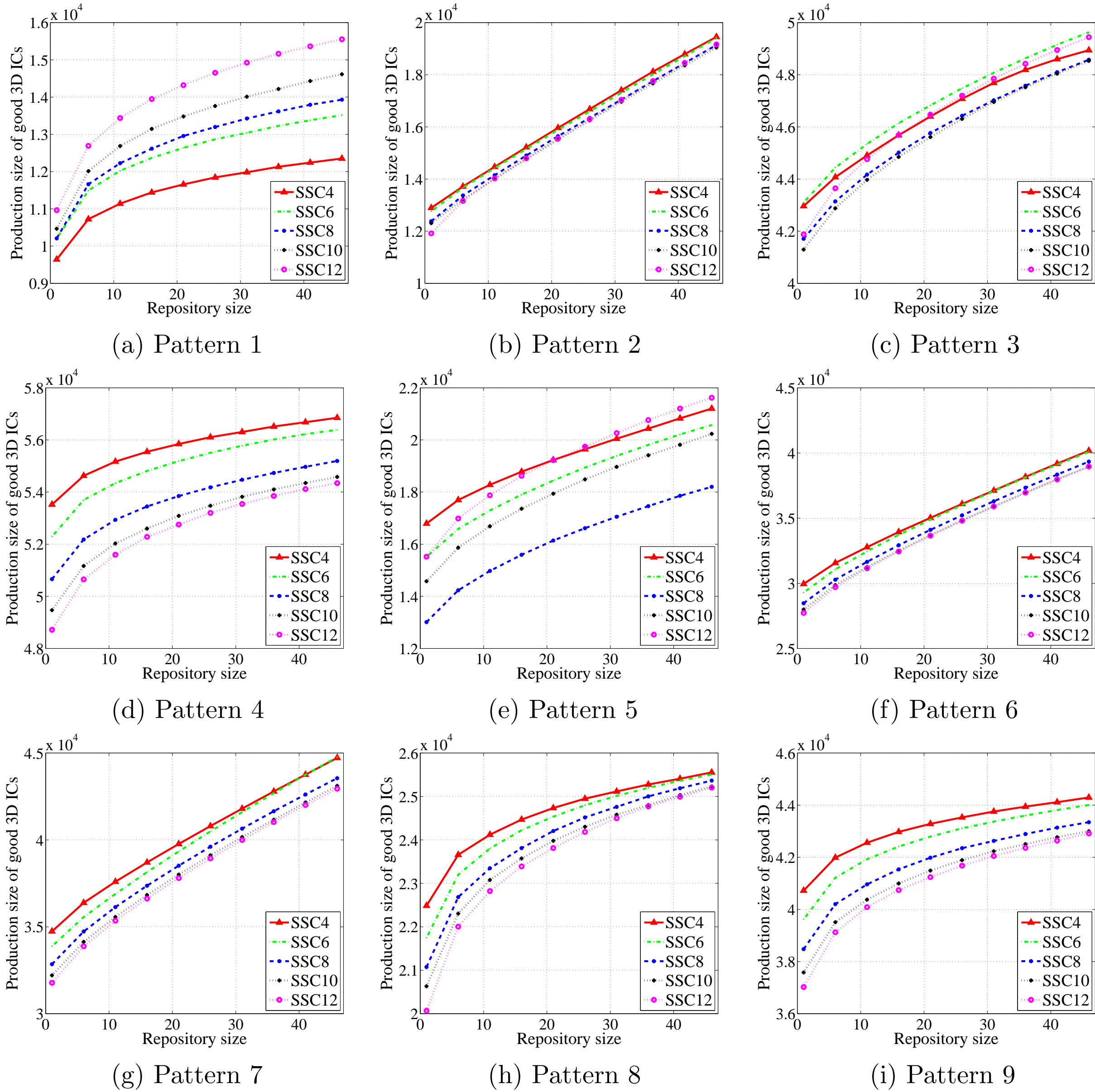



Figure A.1: Exploring the impact of number $n$ of cuts on final production size of good 3D ICs produced by the sector symmetry and cut (SSC$n$) procedure.