Test Cost Reduction of 3D Stacked ICs

Test Planning and Test Flow Selection

Breeta SenGupta

Breeta SenGupta
Department of Electrical and Information Technology
3D Stacked Integrated Chips
Lund University
P.O. Box 118, 221 00 Lund, Sweden

*To my parents
and
Guha sir*

Popular Science Summary

It would certainly be pretty difficult, if not impossible, to find a person who would not fancy owning gadgets they flaunt in Star-Trek, or for that matter, in any Sci-Fi film. And would that be too irrational of us to imagine it to be as affordable as any ordinary gadget we own at present? Following recent advances in science and electronics, it is not unlikely we see that day before we see our grandchildren!

As technology progressed, the heart of all gadgets, the Integrated Circuit or IC, became crowded by the day. Technology had hit the wall when no more could be accommodated within the IC without making our gadgets perform worse. But, technology bounced back with this magnificent solution that held exceptional promises.

Typically ICs had been produced where only one layer of components crowded interiors of the black box, like people in a single storeyed building. But with the new solution, components were now being shared over several layers within the IC. We had evolved into the age of multi-storeyed buildings. We called these multi-layered ICs the 3D Stacked IC. Amongst the first of its kind was the test chip, Pentium 4 processor produced by Intel corporation in the year 2004. The three dimensional version, where circuitry was shared among two layers, gave a performance boost and power saving upon single layered IC by fifteen percent each.

With new ideas came new challenges.

Just as one would essentially need to enter or exit a multi-storeyed structure from the ground floor, so was the case of signals within the 3D Stacked IC. And instead of stairs, electrical signals travels vertically

along the layers of the 3D Stacked IC through copper wires. These copper wires, which have a thickness of just a few nanometers are called the Through-Silicon-Vias, in short, TSVs. However, when even the human hair posed ten times the width of TSVs, constructing them did not come easy. Things got even worse when millions of perfectly made TSVs were required in each layer of the 3D Stacked IC. TSVs were constructed separately for each layer, and eventually lined up. Imagine yourself cutting one strand of hair, and gluing it back exactly as it was! It is also true, not every time do we construct each of the TSVs and line them up impeccably. Neither, in the first place, had we always been precise about creating single layered ICs.

To err might be human, but so is to aspire against!

We confronted the errors in our ICs by developing test mechanisms that ascertained only good ICs were incorporated within our gadgets. However, these test mechanisms did not come for free. All the more, as improved technology kept reducing the cost of manufacturing ICs, not much progress was made in the domain of tests. This led to a test cost accounting for more than three quarters of the total cost for present day ICs. This still pays off for the losses that the manufacturers would have otherwise encountered. For the past half of the century research has been conducted to improve the test of single layered ICs. But 3D Stacked IC brings about a whole new dimension to testing. That includes consideration of TSV technology, among many others. Over the past half of the decade rigorous research is being performed on improving test mechanisms for 3D Stacked IC. Each research has contributed to the betterment of test mechanism for the 3D Stacked IC, howsoever insignificant it may seem. A design and manufacturing group at IMEC, Belgium has suggested several mechanisms to adapt traditional testing strategies for ICs to the 3D Stacked IC. This includes planning of additional circuitry to be incorporated within the 3D Stacked IC that would be involved in testing the TSVs. Research groups at the Georgia Institute of Technology, and University of Michigan have helped develop prototypes for the same.

However, the price we pay for test may not always be tangible. A large share of the cost is the testing time. The remainder of which is accounted for the circuitry, machinery and equipment dedicated for testing. Saving, what might seem a negligible amount of the test time on each IC might prove profitable in the long run. The testing time of an IC in itself takes less than a minute. It would then enable us to test

many more ICs per hour, if even the reduction of test time is a mere second. This in turn would facilitate production of thousands more ICs per day, thereby adding to the profit.

Not much activity has been observed that address the issue of reducing test time for 3D Stacked ICs. A major share of this test time accounts for the many additional stages of production during which 3D Stacked ICs may be tested. Researchers at the Department of Electrical and Information Technology at Lund University in Sweden have shown that random decisions of performing tests at various stages of production might lead to a few hundred times of the minimum test time. They addressed this problem for the first time by building a mathematical model that helps minimize test time by making the right choices during the manufacturing steps, whether or not to perform a test.

In addition, they provide several cost effective measures of laying out the circuitry and organizing tests. This may in some cases help reduce the total cost of the 3D Stacked ICs by up to a tenth.

It may seem infinitesimal, but with each tiny step we strive to create gadgets that do not only provide a superior performance, which also are more robust and reliable. With further improvement in test and manufacturing technologies, slowly but certainly we look forward to the future fancied only in science fiction until now. While Intel engages itself in refining 3D Stacked IC technology with two layers of varying technology, reputed names like Samsung and IBM have been mooting the idea of an eight layer memory stack.

With each new step, past multi-storeyed structures, the age of skyscrapers now dawns upon us!

Abstract

Ever higher levels of integration within the Integrated Circuit (IC) to meet progressively widening scope of its application in respect of functionality, size, performance and manufacturing issues inspired development of the three-dimensional (3D) Stacked IC as a device having viable architecture. However, with increased complexity, manufacturing cost increased. The manufacturing cost includes the test cost component, essential to ensure fidelity to the desired design specifications. Of the several challenges faced by 3D Stacked ICs, cost efficient testing of the manufactured product is most critical. Reduction of test cost for 3D Stacked ICs through test planning along with test flow selection methods is addressed in this thesis.

Test planning for 3D Stacked ICs is performed by reducing the total cost accounting for the test time and Design-for-Test (DfT) hardware. Three test architecture standards are used: Built-In Self-Test (BIST), IEEE 1149.1 and IEEE 1500. The test cost corresponding to each test architecture is detailed and test planning algorithms are proposed. The algorithms are implemented and experiments are performed on several 3D Stacked IC designs formed with multiple 2D IC benchmarks. For experiment, a test flow is presented that comprises the wafer test of each chip followed by test of the entire packaged IC. Results indicate effectiveness of the proposed algorithms in terms of test cost.

Test flow selection, to decide stages at which tests are to be performed, for 3D Stacked ICs is addressed motivated towards the reduction of test time required to produce each single fault-free package. A model to calculate the total test time for any given test flow is detailed. An algorithm is proposed to find a test flow for reducing test time. The algorithm is implemented and executed on several 3D Stacked IC designs with up to ten chips in the stack. Results indicate considerable reductions in test time as compared to predetermined test flows.

Preface

This thesis summarizes my academic work carried out since May 2010. During the period between May 2010 to December 2013, the research work was conducted in the Embedded System Laboratory group at the Department of Computer and Information Science, Linköping University, Sweden. Henceforth the research work has been conducted in the Digital ASIC group at the Department of Electrical and Information Technology, Lund University, Sweden.

The material presented in this thesis is based on the following publications.

Journal articles

- B. SenGupta, U. Ingelsson, and E. Larsson, »Scheduling Tests for 3D Stacked Chips under Power Constraints,« *Journal of Electronics Testing: Theory and Applications (JETTA)*, Vol. 28, No. 1, pp. 121-135, Jan 2012.

  ☞ The first author was responsible for the evaluation of the problem, proposing test scheduling methods, implementation of the experiments and composition of the text. The research work is an extension of the article published in DELTA, 2011 and has been carried out by the first author under the guidance of the remaining authors.

- B. SenGupta, U. Ingelsson, D. Nikolov and E. Larsson, »Test Planning for Core-based Integrated Circuits under Power Constraints,« *Journal of Electronics Testing: Theory and Applications (JETTA)*, Vol. 33, No. 1, pp. 7-23, Feb 2017.

☞ The first author investigated and proposed the test planning algorithm, and implemented the test planning algorithm for experiments. The research work is an extension of the VTS, 2014 paper, written by the first author under the guidance and constant input from the remaining authors.

- B. SENGUPTA, D. NIKOLOV, A. DASH AND E. LARSSON, »Test Flow Selection for Stacked Integrated Circuits,« *Journal of Electronics Testing: Theory and Applications (JETTA)*, Vol. 35, No. 4, pp. 425-440, Aug 2019.

  ☞ The first author has developed and implemented the mathematical model used in TFSA for different test architectures, and written with the remaining supervising authors.

Peer-reviewed conference articles

- B. SENGUPTA, U. INGELSSON, AND E. LARSSON, »Scheduling Tests for 3D Stacked Chips under Power Constraints,« in *6th International Symposium on Electronic Design, Test and Applications (DELTA)*, Queenstown, New Zealand, pp. 72-77, IEEE, Jan 2011.

  ☞ The research work including the development and implementation of the Partial Overlap and ReScheduling algorithms has been carried out by the first author under the guidance of the remaining authors.

- B. SENGUPTA, U. INGELSSON, AND E. LARSSON, »Test Planning for Core-based 3D Stacked ICs with Through-Silicon Vias,« in *25th International Conference on VLSI Design*, Hyderabad, India, pp. 442-447, IEEE, Jan 2012.

  ☞ The first author proposed and implemented the test planning algorithms using the test architecture with continuous guidance from the remaining authors.

- B. SENGUPTA, AND E. LARSSON, »Test Planning and Test Access Mechanism Design for Stacked Chips using ILP,« in *VLSI Test Symposium (VTS)*, Napa, CA, USA, IEEE, Apr 2014.

  ☞ The ILP was formulated and simulated on IEEE 1500 test architecture by the first author under the guidance of the second author.

Acknowledgments

First of all, I would like to convey my heartfelt gratitude for my supervisor Prof. Erik Larsson, for giving me the opportunity and his invaluable guidance over all these years.

I would also like to thank my co-supervisors at Linköping University, Dr. Urban Ingelsson and Prof. Zebo Peng, and Prof. Viktor Owäll, as well as Dr. Dimitar Nikolov at Lund University, for the valuable comments and encouragement.

I would like to thank all the professors at the Digital ASIC Laboratory, here at Lund University for your constructive feedbacks and support. Also, thanks to my fellow doctoral students Hemanth, Rakesh, Isael, Babak and the others for letting us be a part of the journey and the most entertaining parties. Thank you Anne Andersson, Pia Bruhn, Erik Göthe and Linda Bienen for the smooth transition from Linkoping helping out with anything formal.

My life away from home started halfway across the world at Linköping, Sweden. I could not have asked for a more cordial welcome from my colleagues at Linköping University. I would like to thank Petru Eles for the most intriguing coffee-break conversations, Unmesh Bordoloi, Ke Jiang, Adrian Lifa, and the entire team for the table-football sessions between the long work hours and the afterworks. To Min Bao for being the wonderful intercontinental travel guide cum companion and Ivan Ukhov for being the patient listener to my endless ramblings. A big thanks to the administrative staff Anne Moe and Eva Pelayo Danils for making life at the department feel like a breeze.

My deepest gratitude to my wonderful colleagues cum mentors at work, Hans Israelsson, Rajeshwari, Hind, Milan, Jörgen and so many others, from whom I learn something everyday while making the work-

place as loud as it can get. I am also as thankful to my previous colleagues from EC Solutions, specially Oscar, Mathias and Angela for giving me the opportunity and believing in me.

The acknowledgments would not be close to complete without mentioning Dimitar Nikolov and Farrokh Ghani Zadegan, for simply putting up with me for the longest time, through Linköping University, Lund University, and now Ericsson! Thank you for the fruitful discussions, great travels and most of all for being the wonderful friends I can always count on.

I am deeply grateful to all the friends at my corridor in Linköping, who helped make the corridor a home away from home. I can not begin to count my friends here without appreciating Begoña Montanchez, you are the best, most dependable friend one can ask for! I am equally thankful to your wonderful family for the best Christmases I ever had. Surely, the corridor days would have been dull without the lot of you, especially Miguel, Isabella, Jeremy and Mounika.

All my professors during my undergraduate studies at Indian Institute of Technology, Kharagpur, India, for their continuing support and encouragement during and after my time there, thank you. Also, to my friends from the days, who have not let time zones get the better of us. A big shout out to Resham and Abhinendra whose encouragement and example kept me going. My childhood companions Amrita, Gangotri, Niharika, Ipsita and so many others who may be in different continents, yet just a phone call away. I would like to thank all my teachers ever, who have helped shape who I am today.

Today would not have been remotely feasible had it not been for the immeasurable and unconditional love and sacrifice of the two most important people in my life, my parents. Any amount of gratitude would fall short, yet as a token of my respect, here I dedicate this thesis to my two idols: my parents. I am equally indebted to all my grandparents, whose blessings, I am sure, have always been with me every step of the way. And lots of love to all my cousins, aunts and uncles who make all my vacations home feel like a royal endeavor.

<div align="right">

Breeta SenGupta
Lund, June 2020

</div>

Contents

List of Figures

List of Tables

List of Algorithms

Part I

Prologue

This part comprises of two chapters. In Chapter 1 we first introduce the design and test process of 3D Stacked ICs, as it evolved from traditional non-stacked ICs. It is followed by a discussion of the scope of the thesis and the contributions. Chapter 2 presents previous work related to test planning and test flow selection.

1

Introduction

Electronic devices are omnipresent today. They support almost all aspects of our everyday life: phones, digital watches, heaters, lamps, washing machines, televisions, cameras and computers, aviation, automobiles and health-care. At the heart of every electronic device lies Integrated Circuits (ICs). An IC is an electronic circuit, comprising of transistors and passive electrical components - such as resistors and capacitors. Starting with tens of transistors in the late fifties, contemporary ICs may contain several billion transistors.

With an increasing number of transistors being crammed within a single layer of silicon, *aka* die [1–3], even the smallest manufacturing defect resulting in the malfunction of a single transistor can hamper proper functioning of the IC which in turn might lead to a breakdown of the gadget built around it. Therefore, each IC must be carefully tested, to prevent defective gadgets.

Various solutions have been proposed over the years for testing ICs with a single die in the package [20–23]. The cost of testing demands a large share of the total cost of manufacturing ICs [20] [21] [23]. Among the major factors contributing to the cost of testing are test time and hardware [20] [21]. Several methods have been addressed in prior research to reduce the cost of testing ICs comprising a single die, by optimizing the contributing factors to test cost. However, most methods may prove sub-optimal for ICs with multiple layers of silicon bonded together to produce 3D Stacked ICs. Thus, in this thesis, we address the problem of test cost optimization for 3D Stacked ICs by optimizing both test time and hardware.

This chapter is organized as follows. First we discuss design concepts and test methodologies for non-stacked ICs and 3D Stacked ICs

in Section 1.1 and Section 1.2, respectively. The thesis scope is presented in Section 1.3, followed by listing the contributions of the thesis in Section 1.4. Finally, the thesis outline is given in Section 1.5.

1.1. Design of Integrated Circuits

In this section, first we discuss the design of ICs and the cost related to the design and manufacturing process of non-stacked ICs, followed by that for 3D Stacked ICs.

1.1.1. Non-Stacked ICs

In 1958, Jack Kilby presented for the first time an operating integration of semiconductor devices forming a functional circuit on a single block, a monolith. The invention led to the production of the first ICs, earning Jack Kilby the Nobel Prize in physics in the year 2000 [1].

The journey of ICs began with the Small Scale Integration (SSI) technology, that hosted up to ten transistors [2]. These ICs starkly exhibited the improvements – size, cost, speed and accuracy. Reaping the benefits, within less than a decade, IC production technology ascended to Medium Scale Integration (MSI) with a few hundred transistors, followed by Large Scale Integration (LSI) containing tens of thousands transistors [2]. This led Gordon Moore to suggest, in 1965, that the number of transistors in an IC will double every two years [3], illustrated by Figure 1.1. Keeping pace with Moore's law, Very Large Scale Integration (VLSI) accommodated up to a million transistors by early eighties. Since the late eighties, ICs with over millions of transistors are being constructed with Ultra Large Scale Integration (ULSI) technology.

The state-of-art production process of ICs is complex, rigorous and time consuming. Furthermore, it is a costly process that not only includes the cost of the materials, but more importantly that of the process equipment.

The complex manufacturing procedure of ICs begin at the Front-End-of-Line (FEOL) processes, that comprises fabricating the transistors on the silicon, the design is planned and verified. Once the design is frozen, masks are made to pattern out the circuits. Silicon crystals with a very high purity are used to produce a grid of multiple dies, called wafers, illustrated by the figure on the left in Figure 1.2. The wafers are then coated with silicon dioxide followed by an insulating silicon oxide layer. The mask is used for photolithography

Figure 1.1.: Representation of Moore's Law, that states that the number of transistors, per square inch, on a IC doubles approximately every two years

with ultra violet light and then the wafer is etched and finally doped. The Back-End-Of-Line (BEOL) process involves adding the connecting copper wires. In this process a metal layer is deposited on the silicon surface, which undergoes photolithograpphy followed by etching to bring about the wires. The side of the wafer where the circuit is materialized by etching and metal deposition is termed the active (face) side, while the substrate is the back side. The final wafer is obtained after all the layers of wiring have been developed. The final wafer is then cut into individual dies, and each die is packaged in an insulated casing with pins for transfer of signals, as shown by the right figure in Figure 1.2. Figure 1.3 depicts the schematic of a packaged IC. The packaged die comprises of the active metal layer and a semiconductor substrate. Connections to the package pins are established via the flip-chip bonds, for example.

As the electronic circuits become progressively complex and diverse (heterogeneous), the IC design is partitioned into smaller blocks, known as cores, which are connected by on-chip wires for the transport of sig-

(a)   (b)

Figure 1.2.: Manufacturing stages that provide test op-
portunities for traditional ICs: (a) a wafer
containing a grid of dies (chips), and (b) a
packaged IC
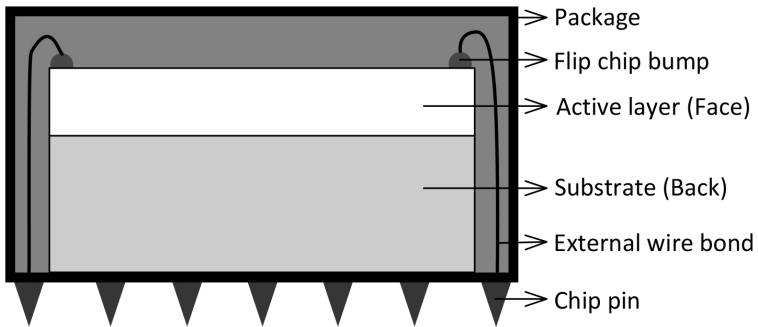
Figure 1.3.: Lateral-cross section schematic of a wafer
within a package and the hardware interface
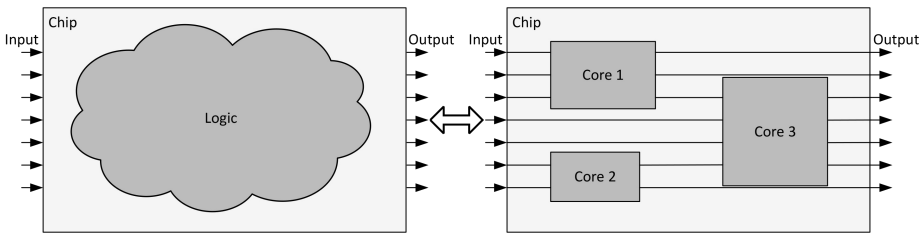
Figure 1.4.: Partitioning of IC logic into cores

nal and data. In Figure 1.4, an example of an IC is shown, that contains three cores: Core 1, Core 2 and Core 3. The cores within the chip are accessed during operation or test by a number of wires. While core based design of ICs successfully match with the increasing complexity of IC designs, it also supports the import of external design expertise by integrating third-party design blocks.

The emergence of portable electronic devices required smaller form factors. This was initially achieved over the two-dimensional space by scaling transistors or integrating multiple chips inside a package. This led to the development of System-on-a-Chip (SoC), where all components were developed on a single chip (die) [4]. In other words, the SoC comprised all: digital, analog, timer, interface, logic, memory and other blocks within the same package. The high manufacturing cost of SoCs, due to low production yield, was addressed by Multi-Chip Modules (MCMs) [20]. MCMs are constructed by interconnecting multiple chips placed beside one another on a Printed Circuit Board (PCB) to achieve a smaller form factor. However, MCMs turned out to be capital-incentive and technology-complex as compared to Multi-Chip Packages (MCPs). As the name suggests, MCPs comprise of bare wafers placed side by side within a single pakage. MCPs were succeeded by the System-in-Package (SiP) that contained all components of a system integrated inside a package. A major drawback with SoCs, MCMs, MCPs and SiPs was low transistor density owing to the large portion of chip area dedicated to interconnects and other passive components [5]. System-on-Package (SoP) addressed the issue by using nano-scale embedded thin film components, thereby reducing the system size by up to a thousand times. In addition, SoPs also reduced the latency of previous integrations. Eventually the Package-on-Package (PoP) was introduced to account for the low yield and inflexibility of SoPs that provided even higher PCB-space savings and improved performance. Finally, 3D Stacked ICs with Through-Silicon Vias (TSVs) were introduced to achieve even better performance within a smaller form factor, where multiple bare chips are stacked and bonded with vertical copper wires through the substrate within a single package.

1.1.2. 3D Stacked ICs

The quest for achieving performance improvement on a smaller footprint led to the development of 3D Stacked ICs. 3D Stacked ICs may be manufactured by a monolithic approach. Similar to ICs with a single chip, the circuit is fabricated on a wafer, followed by coating the circuitry with a layer of dielectric material, for electrical insulation.

TSVs are etched through the dielectric layer, and additional circuits can be fabricated on top of the layer. However, the monolithic approach proves to be economically challenging, therefore typically each layer of a 3D Stacked IC is fabricated on separate wafers. The wafers are then stacked and bonded with vertical copper wires, called TSVs, through the substrate. The advent of 3D integration technology promises to revolutionize the IC industry. Among the benefits are [6–11]:

- Smaller footprint due to the vertical stacking of chips, as compared to laying them on the same plane.

- Heterogeneous integration is convenient. Chips (dies) manufactured with different technologies from different manufacturers as well as different functionality such as logic, memory or sensors can be stacked together.

- Partitioning the functionality of a system over individual chips reduces the complexity of each chip. In addition, the chips can be tested individually that enables reduction of cost by providing higher yield.

- The introduction of TSVs has a number of advantages [12]:
    - The total wire length on the chip is reduced due to direct vertical interconnects. A shorter wire length in turn reduces propagation delay, thereby increasing signal speed within the system, leading to better performance.
    - The smaller dimensions of TSVs as compared to external wires reduces the capacitance and hence boosts performance by up to 30% and 40% less power.
    - In addition, it is possible to provide a large number of TSVs within the chips and thus increasing the bandwidth.

The 3D Stacked IC illustrated in Figure 1.5 consists of three chips in the package, bonded by TSVs. The manufacturing process of 3D Stacked ICs is a complex and rigorous one. After the chips constituting the stack are manufactured, they are thinned, aligned and bonded prior to packaging.

In contrast to non-stacked ICs that do not contain TSVs, for 3D Stacked ICs TSVs are fabricated at the end of the manufacturing process of the dies. TSVs are essentially cylindrical copper wires that run through the substrate layers of chips, orthogonal to the active layer, hence making an electrical connection between the chips, enabling
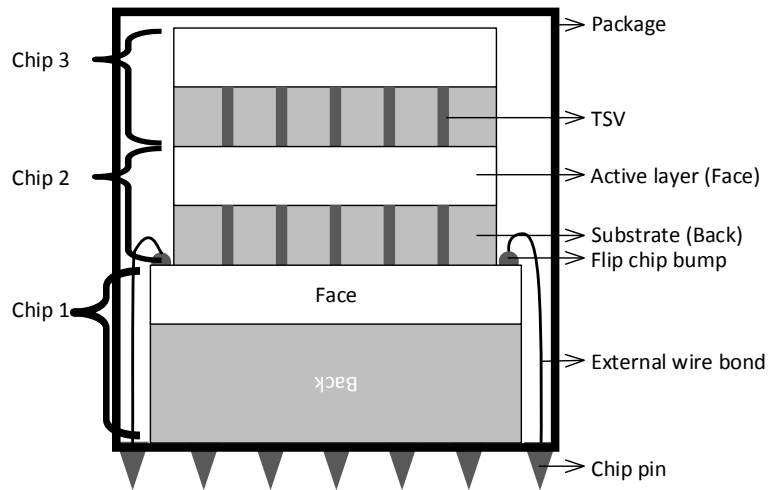
Figure 1.5.: Block diagram of a packaged 3D Stacked IC

stacks. However, compared to external wires, TSVs have considerably smaller dimensions: height about a few tens μm and diameter less than ten μm. TSVs are fabricated by first etching the TSV holes in the silicon substrate. The etching process is followed by oxide and copper deposition.

The chips then undergoes copper plating, and finally chemical - mechanical polishing. TSVs can be fabricated on the chip at various stages, depending on whether it undergoes the via-first, via-middle or via-last process. During the via-first process, TSVs are fabricated before both the FEOL or BEOL processes. In via-middle process, the TSVs are fabricated post FEOL but prior to BEOL. Finally, for via-last, both FEOL and BEOL processes are concluded before fabricating TSVs. The dimensions of via-first and via-middle are much smaller as compared to via-last. Consequently, via-first and via-middle are analogous to semiconductor interconnect technology whereas via-last resembles interconnect technology.

Once the chips are fabricated on the wafers, the substrate layers undergo thinning to expose the ends of TSVs. During the process, the substrate layer, which is close to one millimeter thick, is thinned down to a few tens of μm, to match the height of TSVs. Conversely, TSVs require a much larger diameter in order to match the thickness of the substrate to ensure proper copper deposition.

Following the thinning process, the chips are stacked. There are sev-

eral alternatives to consider during the stacking process, one of which is the orientation of the chips. The chips may be stacked face-to-face, back-to-back, or face-to-back. For face-to-face stacking, the active layers, *aka* faces, of two chips are interconnected; the top chip is placed face down on the bottom chip. TSVs may be unnecessary in face-to-face stacking, and external wire bonds used for external communication, that are connected to the lower chip having a larger surface area.

Back-to-back stacking involves the interconnecting the substrate layers, *aka* backs, of two chips using TSVs. External communications can be carried out using flip-chip bumps or wires from one of the faces. Thus, for two chips in the stack, back-to-back stacking would prove more cost efficient, due to the need of TSVs. However, both face-to-face and back-to-back bonding is limited to scaling over two chips in the stack. Face-to-back bonding is scalable to any number of chips, where the face of one chip is connected to the back of another chip using TSVs. The 3D Stacked IC illustrated in Figure 1.5 consists of three chips with face-to-back stacking.

Another variation in the stacking process occurs with die-to-die (D2D), die-to-wafer (D2W) and wafer-to-wafer (W2W) stacking [13]. In case of D2D stacking, the wafers are separated into individual chips, while only certain chips in the stack are separated from the wafers in D2W stacking. In W2W stacking where entire wafers are stacked, the time associated with the stacking and alignment of individual dies is avoided. However, on the other hand, W2W stacking provides less flexibility while stacking known good dies (KGDs). As a result, faulty dies may be stacked on good dies, adversely affecting the yield of production. Furthermore, the yield decreases exponentially with scaling [6][8][14].

The chips are aligned and eventually bonded, after stacking. Thermo-compression bonding is commonly used among bonding technologies. The pressure and temperature required for the bonding process varies with the materials used for manufacturing TSVs and the respective landing-pads.

Despite providing several ground-breaking advantages over its predecessors, a lot of factors thwart the course of large scale production of 3D Stacked ICs. For instance, the cost benefit of 3D Stacked ICs over non-stacked ICs is maximized by scaling the number of chips forming the stack [6][8][9][11][14–17]. However, the complex production process limits the scaling of 3D Stacked ICs. In addition, 3D Stacked ICs require a different set of design tools as compared to non-stacked ICs. The additional process steps provide a scope for additional defects like cracking while thinning or bonding, or misalignment of TSVs while stacking, thereby reducing the production yield. New

challenges arise with the introduction of TSVs, such as the additional steps in the manufacturing process and area overhead. Finally, traditional test methods used for non-stacked ICs, may not apply directly to 3D Stacked ICs. Despite the challenges, several industrial 3D Stacked IC design prototypes have been produced. Intel corporation simulated a 3D Stacked prototype of the Pentium 4 CPU with two chips bonded face-to-face [18]. This led to a 15% reduction in power consumption with respect to its non-stacked counterpart, and a 35% improvement in efficiency, as measured by performance per watt. In addition, Tezzaron Semiconductor built multiple 3D Stacked IC designs [19] with two chips in the stack, also bonded face-to-face. All designs exhibited considerably higher speed and lower power consumption. Due to a flexible design, 3D Stacked memory ICs with up to eight chips in the stack have been manufactured. Hence, 3D Stacked IC technology promises great outcomes regardless of a few design challenges.

1.2. **Test of Integrated Circuits**

Testing is an integral part of IC production. As feature sizes keep decreasing, the manufacturing process of ICs demands atomic precision. This makes ICs more susceptible to defects introduced during the manufacturing process. Even the smallest impurity may cause an electrical component to malfunction, and in turn, causes the whole system to fail. Therefore, each IC must be carefully tested to determine whether the chip has been manufactured correctly. Testing not only sorts out the faulty parts but can also help in improving production yield of the manufacturing process, if the cause of defects is analysed. This makes testing an inevitable process during the manufacturing of ICs.

Besides being a complex process there is a considerable cost related to testing. Different factors contribute to the total test cost, which can be broadly categorized into two domains: fixed cost and variable cost. The fixed cost comprises of the cost of equipment external to the ICs, which do not change with the production process. An example of fixed cost is the cost of Automatic Test Equipment (ATEs). Variable cost, on the other hand increases with each additional unit produced. It comprises of factors internal to each individual IC, such as the time required for testing and the hardware dedicated to test [20]. Although reductions in variable test cost of each individual IC may be small, in large scale production the cumulative reduction in the cost of testing has a big impact. Hence, a lot of research has addressed the minimization of the variable test cost of ICs [2] [20–23] as is the goal of this thesis.

New technologies enhance the yield of devices reducing the fabrication cost. However, strategies developed for testing do not reduce test cost in synchrony, thus increasing the share of test cost for the manufacturing process [2] [21–23]. Therefore, it is not sufficient only to thoroughly test each IC, but it is also important that the tests consume minimal resources, and in turn the cost of testing is reduced.

2D ICs are tested by applying test stimuli to the Device Under Test (DUT) and compare the response to an expected output, as illustrated in Figure 1.6. Generally, ICs are tested using an ATE, which is capable of performing a number of tests based on the test program. The ATE performs the test by supplying a set of binary patterns, called test vectors to the inputs of the DUT. The test vectors are generated by analyzing the DUT. The number of test vectors impact the test application time. The DUT is deemed fault-free or good if the output matches the expected response; otherwise the DUT is faulty and is extracted.

The order in which the test vectors are applied is determined in
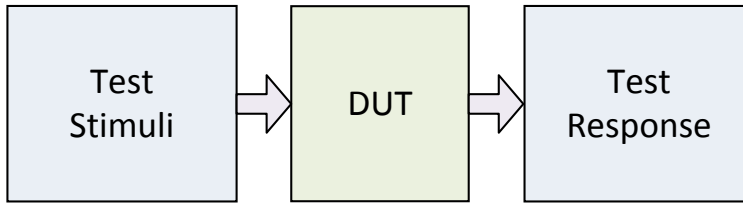
Figure 1.6.: A generic approach to testing ICs
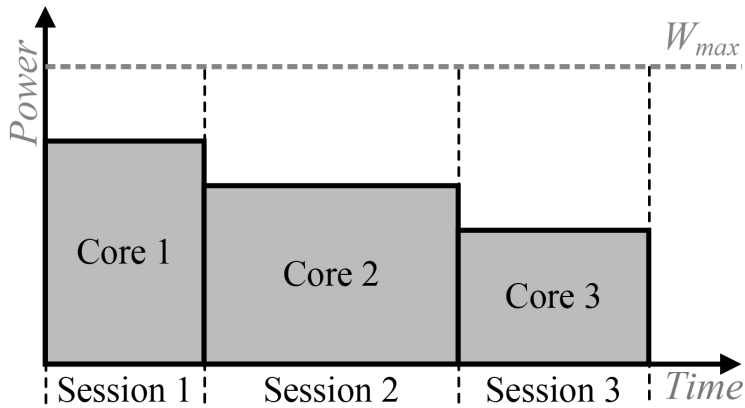
Figure 1.7.: Test schedule showing the power constraint
and time taken

advance. For core-based designs, there are tests for each core. Each
core has a time assigned for the application of tests. The time for
initiating the tests of each core is determined such that all constraints,
such as power dissipation, resource management and test conflicts, are
met. This order of tests is known as the test schedule. An example of
a possible test schedule for each core of the chip given in Figure 1.4 is
shown in Figure 1.7. The time taken to run the entire test schedule is
the test time for the DUT, shown by the horizontal axis in Figure 1.7.
The total time required by the test schedule is the sum of the time
taken by the individual cores, as all cores are tested serially in this test
schedule.

The test schedule can be optimized with respect to test time. Reduc-
tion in test time not only reduces the time to market, but also increases
the profit margin by enabling more chips to be produced within the
same time frame.

The vertical axis in Figure 1.7 illustrates the power consumed. The

horizontal line marked by W_{max} shows the maximum power dissipation allowed. The test power consumption must be kept under control, to avoid false test positives due to voltage drop or damage due to overheating.

It can also be noticed in Figure 1.7 that the test of a core is initiated only when the test of previous core(s) are completed. Multiple cores may be tested in the same sessions, and the test of each core is initiated simultaneously. In other words, a session can be defined as a group of tests that start simultaneously and no other tests are initiated until all tests of the session are finished. The concept of sessions simplify test scheduling under power constraints, because once the tests have been allocated to sessions, that are within the power limit, the schedule is found by processing the sessions in a sequence.

Prior to determining a cost efficient test schedule for ICs, provision should be made for controllability and observability of signals at various points inside the IC. Thus, tests can be applied and responses obtained for each core of the IC individually. Therefore, considering test early during the design process, additional hardware dedicated to testing is introduced into the IC. The process is known as design for testability (DfT). DfT was first defined during the seventies [2], when ad hoc methods were developed, aimed at testing parts of the IC that were difficult to access. Later, as DfT methods began to be standardized, several test architecture standards were proposed. With the standardized test architectures, all parts of the IC could be tested, enabling testing of cores developed by different manufacturers.

Therefore, in the following section, we discuss the test process of traditional non-stacked ICs, including the test scheduling and corresponding test architectures.

1.2.1. Non-Stacked ICs

In this section we discuss two factors contributing to the test cost of ICs: DfT hardware and test time. The DfT hardware based on three test architectures is discussed, *namely*, Built-In Self-Test (BIST), IEEE 1149.1 and IEEE 1500. We discuss the impact of test scheduling and test flow selection as contributors to test time.

Scan based designs, introduced by Williams *et. al.* [24], use the flipflops in the IC to form shift registers. A multiplexer and clock is added to every circuit flip-flop for testing, shown in Figure 1.8, so that in scan mode, the flip-flops are converted into scan chains. The advantage with scan based testing is that the DUT behaves as a combinational circuit. The inputs and outputs of the shift registers behave as the
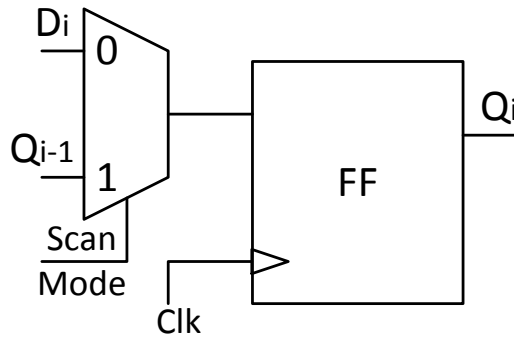
Figure 1.8.: Scan design

primary inputs and outputs, respectively. By using the scan design, the test vectors, also called test patterns are shifted in to the shift register. After a capture cycle, the test response is shifted out, while the next test pattern is loaded. All scan paths in the DUT can be observed within the number of clock cycles equal to the number of flip-flops in the longest scan chain.

ICs may be tested at several occasions, both during manufacturing and operation. Hence, test can be integrated as one of the functions of the IC, making the test process both easy and efficient. Frohwerk [25], thus conceptualized the BIST, where additional logic was inserted into the hardware of the IC to generate test vectors, and analyze the output. BIST could be used at any level of granularity, from core-level to wafer-level tests.

BIST provides several advantages. As external devices such as the ATE make use of the DUT's I/O ports for transfer of signal, test becomes a comparatively slow and lengthy process. BIST is elaborated in Appendix A.1.

Test architecture standards utilizing scan based designs have been used to simplify the test of non-stacked ICs. Two such standard designs are the IEEE 1149.1 and the IEEE 1500. Both test architecture standards are briefly discussed below.

The IEEE 1149.1, commonly known as boundary scan, or Joint Test Action Group (JTAG), was initially developed for testing interconnects on the PCB. The concept of scan design was implemented on the terminals of an IC. The I/O terminals were used to form a shift register, to improve controllability and observability on the IC. The IEEE 1149.1 standard was later implemented within ICs, that enabled testing of cores. The standard requires four obligatory and an optional port for

testing. For performing tests on the logic internal to the IC, flip-flops are concatenated to form scan chains. The standard is detailed in Appendix A.2.

The IEEE 1500 test architecture standard was developed primarily to standardize the core test architecture. The architecture was standardized by defining a test access mechanism (TAM) for the cores and the interconnect between the cores. The most important feature of the standard is the provision of a test wrapper, formed by inserting a scan cell at the terminals of each core. The core test wrapper serves as the interface between the TAM and the core. The wrapper allows three modes of operation: the normal mode during operation of the core, the internal test mode to test logic internal to the circuit, and the external test mode for testing interconnects. The TAM transports data from the test source to the core, and the core to the test sink, during different modes of operation of the test wrapper. In addition to a mandatory serial port (WSP), IEEE 1500 provides for an optional parallel port (WPP), that allows to considerably reduce the test time as compared to IEEE 1149.1 by allowing multiple smaller scan chains, as opposed to a single longer scan chain. The concatenated scan cells between the terminals of a core, are referred to as wrapper chains. An IEEE 1149.1 test access port (TAP) controller can be used for accessing an IEEE 1500 wrapper. The IEEE 1500 test standard is elaborated in Appendix A.3.

Several methods are addressed in [26] for the reduction of the variable component of test cost. Important among them are the test time and DfT hardware optimization. Other approaches include: test data compression, yield learning, adaptive or structural testing, built in fault tolerance, or introducing new technologies of testing. Minimizing one factor of test cost might adversely affect other contributors to the cost due to co-dependency, thereby increasing the overall test cost. As an example, we discuss the co-dependency between test time and the DfT hardware. Simultaneous testing of multiple cores in a test schedule reduces test time. However, it may not be possible to test all cores concurrently, as that may employ resources beyond the acceptable values. Other notable factors that restrict the concurrent testing of cores is the power dissipation. To avoid false test positives due to voltage drop, or damage due to overheating, the power dissipation during testing must be checked [20]. In addition, being able to control each core individually during test would require more DfT hardware dedicated to each core, as compared to limited controllability by sharing hardware resources. Therefore it is important to find the best trade-off among the several factors contributing to test cost while devising a test cost reduction strategy, *i.e.,* test plan. In this thesis, test plan refers to

scheduling core tests for 3D Stacked ICs with a test architecture, such that the total cost related to test time and DfT hardware is reduced.

Performing tests at each step ensures lower wastage of additional products during following stages, providing fault-free inputs to the next stage. Therefore, test can performed after every manufacturing step. These possible opportunities when a test can be inserted are called test instances. For non-stacked ICs, there are two different manufacturing steps, wafer fabrication followed by packaging. This provides an opportunity to perform tests at two instances. The test flow, *i.e.*, sequence of test instances when test is performed, for non-stacked ICs typically includes both wafer sort and package test [20] [27] as shown in Figure 1.9(a):

1. *Wafer sort*: The bare chip is tested to avoid packaging of faulty chips. The chips which appear to be fault free during wafer sort are termed known good dies (KGDs), shown by the upper block in Figure 1.9(a).

2. *Package test*: KGDs are packaged, and the test is applied to the complete packaged IC, illustrated by the lower block in Figure 1.9(a).

A chip is typically tested both during wafer sort and package test. Hence, test planning strategies developed to optimize either test instance for cost would result in an optimized test plan for the non-stacked IC.

1.2.2. 3D Stacked ICs

In this section we discuss test of 3D Stacked ICs. We first discuss test flow, followed by the challenges faced in testing 3D Stacked ICs when traditional test methods for non-stacked ICs are applied.

During the test of non-stacked ICs, the same tests are applied at both wafer sort and package test instances. This is because the same DUT is tested both at wafer sort and package test. Hence, it is sufficient to optimize any of the test instances to determine a test plan. However, unlike non-stacked ICs, the manufacturing process of 3D Stacked ICs involves multiple manufacturing steps. A test can be inserted after each manufacturing stage of the 3D Stacked IC. Contrary to non-stacked ICs, the DUT in case of 3D Stacked ICs changes for every test instance. Hence, optimizing the test cost for individual test instances might yield sub-optimal results overall.

The possible number of stages when a test can be performed are proportional to the number of chips in the 3D Stacked IC. The test
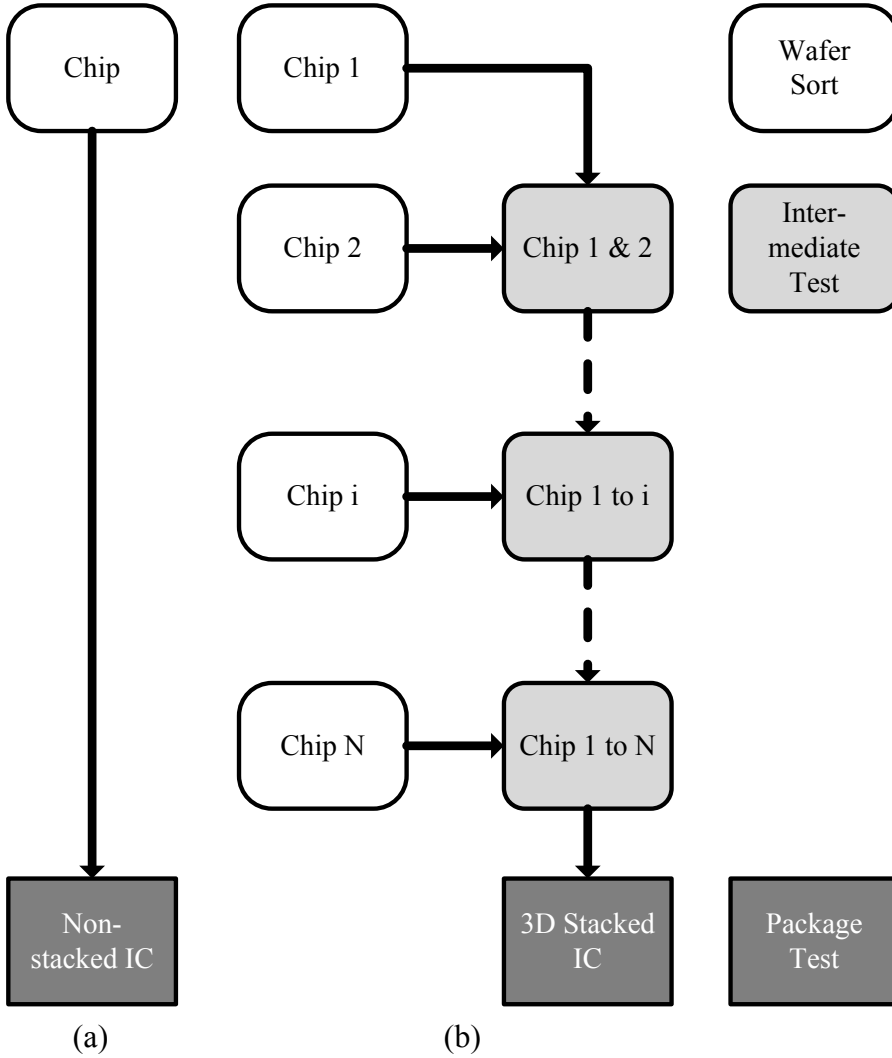
Figure 1.9.: Test flow comparison between
(a) Non-stacked IC, and
(b) 3D Stacked IC

instances for a 3D Stacked IC constituting N chips in the stack is shown in Figure 1.9(b). Each box/rectangle in the figure represents an unit obtained after each manufacturing stage. An arrow from one box to another shows that the components of the former box contributes to the later. The test instances can be broadly classified as follows:

1. *Wafer Sort (WS):* During this stage, each individual chip can be tested prior to the stacking process. This prevents additional wastage incurred by stacking good dies along with faulty ones. However, it may be often possible that testing an individual die requires so much time and resources that it is more efficient to waste a few good dies which have been stacked on them.

   In Figure 1.9(b), the left column in white represents all wafer sort instances. For a 3D Stacked IC with N chips in the stack, there are N possible wafer sort instances, one for each chip. It can be seen that each wafer sort instance contributes to a intermediate test instance.

2. *Intermediate Tests (IT):* At each stacking event, starting from the second chip stacked over the first one, to the final chip being stacked, a test can be performed. This avoids wastage of good dies stacked over faulty partial stacks. Similar to wafer sort, it may also be more cost efficient at certain instances to incur the wastage of good dies than to perform the test.

   In Figure 1.9(b), the first to the last-but-one box in the right column represent the intermediate test instances. For a 3D Stacked IC with N chips in the stack, there are $N-1$ possible intermediate test instances, starting with the stacking of the second chip on the first, up to the Nth chip. Each intermediate test instance, as can be seen, receives components from two prior manufacturing stages: a wafer sort and the preceding intermediate stage. Therefore it can be said, provided that the tests preceding a certain intermediate test instance have not been performed, the yield of the intermediate test instance effectively reduces. In other words, if test has not been performed during the test instances preceding a certain intermediate test instance, more number of units need to be tested to obtain the same number of good partial stacks, than otherwise.

3. *Package Test (PT):* To ensure product reliability a test is performed by the manufacturer after packaging the 3D Stacked IC before delivery to the customer. In this thesis, it is assumed that the

package test is performed for all 3D Stacked ICs to ensure all outgoing chips are fault-free.

In Figure 1.9(b), the last box in the bottom row represents the package test instance. It can be seen that all previous test instances eventually contribute to the package test.

More test instances in case of 3D Stacked ICs, as compared non-stacked ICs, demand different approach for the reduction of test cost by test planning. Thus, test strategies developed for non-stacked ICs might be rendered sub-optimal if applied to 3D Stacked ICs.

Akin to non-stacked ICs, research is being devoted for the development of test standard for 3D Stacked ICs. The IEEE P1838 test architecture standard is being developed for test access to and between chips in 3D Stacked ICs [28]. The standard proposes test of TSVs using a die wrapper register (DWR) at the interconnects on each chip compliant to the test standard. In addition, each chip also comprises of a TAP controller based on the IEEE 1149.1 standard, along with an user defined flexible parallel port (FPP). However, the IEEE P1838 standard does not include intra-chip DfT, *i.e.*, the test architecture for testing the cores in each chip.

Beside the need for a test plan optimized with respect to the test flow, several new test challenges arise with 3D Stacked ICs:

- Additional steps during the manufacturing of 3D Stacked ICs increase the chances of introducing a defect to the chip.

- Unlike non-stacked ICs, where the same test schedule is applied on the same circuit at both instances, in case of 3D Stacked ICs, different circuits are tested at different instances. Thus optimizing all test instances individually may lead to sub-optimal test plans for 3D Stacked ICs. To arrive at a test plan for 3D Stacked ICs, all tested instances must be considered simultaneously.

- The introduction of TSVs brings the possibility of new defects. The copper filling process during the fabrication of TSVs is often imperfect, leading to short or open faults. Hence, test specialized for ensuring the proper functioning of TSVs must be developed.

- Since test cost takes a large share of the entire manufacturing process, the scaling of test cost against the benefits provided by the 3D integration process must be considered.

- Due to additional manufacturing steps, 3D Stacked ICs provide more opportunities for testing at different stages. The choice of different test flows increases exponentially with each additional chip in the stack. The optimal test flow that chooses to test at certain instances to reduce the overall cost must be determined.

- Approaches to reduce the test cost of non-stacked ICs have been studied over the years. However, the methods used for non-stacked ICs may not be directly applicable for 3D Stacked ICs. New studies are needed to obtain optimal test strategies for 3D Stacked ICs.

- Thermal constraints must be considered while planning test strategies. As compared to non-stacked ICs, heat dissipation in 3D Stacked ICs is entirely different, rendering the studies on power and heat on non-stacked ICs redundant for 3D Stacked ICs.

- The tests applied to the chips at wafer sort is limited by the fact that circuitry might be spread among a number of chips on the stack. To ensure best performance, structural constrains and nano-scale speed and layout defects need focus.

1.3. Thesis Scope

3D Stacked ICs may be manufactured with different approaches, such as manufactured as a monolith or by stacking individual layers. Stacking multiple layers of chips however gives greater flexibility and fewer defects. Hence we consider 3D Stacked ICs produced by stacking multiple chips in this thesis. However, methods previously employed for the reduction of test cost for non-stacked ICs may not hold good for 3D Stacked ICs. ICs are typically partitioned into a number of cores that not only ease integration of components from different providers, but also facilitate the development of standardized DfT methods for the ICs.

This thesis therefore, aims at factors contributing to the test cost of ICs, and their optimization.

Two factors that contribute significantly to the cost of testing are test time and DfT hardware. Small reductions in the testing time of each IC add up to a notably shorter time to market for large scale productions. Reduction of DfT hardware, on the other hand, not only cuts down on the corresponding cost, but also reduces/spares routing area on the

silicon. However, optimizing test time may often lead to increased use of DfT hardware, and vice versa.

Hence, we address test planning, to reduce test cost, by co-optimizing both test time and DfT hardware.

The DfT hardware based on three test architectures is discussed: BIST, IEEE 1149.1 and IEEE 1500.

BIST is a widely employed DfT method that allows large reductions in test time by reducing the amount of data correspondence with the ATE. The test time is reduced by scheduling the the test of each core. In this thesis, the core tests are scheduled in sessions. By so scheduling the core tests, it is possible to share hardware resources among the cores within a session. For example, we assume the cores belonging to the same session share the same TDR. Thus, by compromising on the flexibility of the scheduling of core tests, the DfT hardware cost can be reduced. In addition, the test schedule is further constrained by the maximum power dissipation allowed at any point in time. This prevents, at all instances during testing, damage due to overheating of the chip.

Test planning for 3D Stacked ICs under a power constraint, where each chip is provided with an IEEE 1149.1 TAP, is addressed in this thesis.

A common approach to scan-based designs is the IEEE 1149.1 standard. The test plan co-optimizes the cost corresponding to the number of TDRs and the time taken by the test schedule. However, test planning for scan-based chips is different from BISTed chips. Unlike chips with BISTed cores, the test time for each core is not constant for scan-based test. The time required to test any core depends on the maximum number of test patterns required by any core in the same session.

The benefits of using the IEEE 1149.1 standard gets limited for chips with cores that comprise of a large number of scan-chains. Hence, we adopt the IEEE 1500 standard that allows parallelization of the scan-chains to form wrapper-chains within each core. By dividing the total time over multiple TAM lines, this allows reduction in test time, as compared to an IEEE 1149.1 based test architecture. However, each additional TAM line corresponds to an added pin to the 3D Stacked IC, accumulating to a higher cost of DfT hardware.

Consequently, we address test planning to co-optimize the cost related to the width of the TAM required by the 3D Stacked IC, and that of the time required by the test schedule. For performing experiments on the proposed test planning approaches, we assume a test flow similar to that employed for non-stacked ICs.

The test flow comprises of the wafer sort of each chip, followed by the package test. However, unlike non-stacked ICs, 3D Stacked ICs can assume a number of different test flows. The impact of a chosen test flow on the cost of a 3D Stacked IC has been sparsely studied.

Therefore, in this thesis, we focus on obtaining a suitable test flow for 3D Stacked ICs, such that the total time required to produce each good package is reduced. The impact of varying yield and test time of each component is studied to propose a method to lower the time required to obtain each good package of 3D Stacked IC.

Thus, this thesis addresses test cost reduction for core-based 3D Stacked ICs. The problem is approached from two directions: test planning and test flow selection. Test planning methods are employed to co-optimize the cost of test time, corresponding to the test schedule and the DfT hardware required. Three DfT methods are assumed to establish and solve the problem: BIST, IEEE 1149.1 and IEEE 1500. The time taken by any test flow, based on given test time and yield values, is studied to propose a method for determining a test flow to lower the time required to obtain each good packaged 3D Stacked IC.

1.4. Contributions

The contribution in this thesis broadly includes test planning methods for 3D Stacked ICs that lead to a lower cost of test. The problem of test cost reduction for 3D Stacked ICs is addressed in two parts. The first part involves test planning of core-based 3D Stacked ICs, whereas in the next part, the problem of determining a test flow for 3D Stacked ICs is addressed.

1. Test planning of core-based 3D Stacked ICs is addressed for three different test architectures, discussed below in order.

    a) *Test planning with a BISTed test architecture.* A BIST scheme is considered for test scheduling of 3D Stacked ICs within a power constraint. We assume each core in the 3D Stacked IC is provided with a BIST engine. The cores are accessed through an on-chip JTAG TAP via TDRs. Only one TDR of a particular JTAG TAP can be chosen at a time. This obliges the concept of sessions within a chip. The power dissipated while testing any session may not exceed the given power limit. It is further assumed, TDRs connected to different JTAG TAPs can be chosen simultaneously. Therefore, a test

session may include tests of cores from different chips simultaneously.

The test schedule obtained by applying methods proposed for non-stacked ICs is applied on several 3D Stacked IC designs by Serial Processing. In Serial Processing, the optimized wafer sort schedule of each chip forming the 3D Stacked IC is executed in full without any changes. Two approaches for reducing the test time are proposed, *viz.*, Partial Overlap and ReScheduling. For Partial Overlap, entire test sessions from different chips in the 3D Stacked IC may be performed in parallel, during package test, provided that the power constraint is met. In case of ReScheduling, core tests contributing to the test sessions during wafer sort may be separated and scheduled along with cores of other chips in the stack.

Experiments were performed on several 3D Stacked IC designs formed by benchmarks defined for non-stacked ICs.

It is established that Partial Overlap reduces the test time as compared to Serial Processing. However, the lowest test time is obtained by ReScheduling.

b) *Test planning with an IEEE 1149.1 based test architecture* Test planning of 3D Stacked ICs is done by an on-chip IEEE 1149.1 test architecture provided in each chip of the stack. The test cost is given as the weighted sum of the test time and the DfT hardware. The test time is calculated as the sum of the time taken for wafer sort of individual chips and the package test time. The cost corresponding to the DfT hardware is given by the number of TDRs. A mathematical model is presented to arrive at the overall test cost. A heuristic is proposed to reduce the test cost. The proposed heuristic is compared against the test plan obtained by Simulated Annealing on several 3D Stacked IC designs obtained by stacking ITC'02 benchmark designs.

It is substantiated that the heuristic performs efficiently as compared to Simulated Annealing, within a shorter CPU time.

c) *Test planning with an IEEE 1500 based test architecture* The test cost comprises the weighted sum of the test time and the DfT hardware. The test time is calculated as the sum of the

time taken for wafer sort of individual chips and the package test time. The cost corresponding to the DfT hardware is given by the width of the TAM required. A mathematical model is presented to arrive at the overall test cost. An algorithm, using Integer Linear Programming (ILP), is proposed to lower the test cost by co-optimizing the total test time and the TAM. The algorithm is executed on several 3D Stacked IC designs formed by stacking ITC'02 benchmark designs and compared against two test planning schemes developed for non-stacked ICs, that do not take into account several distinctive features of 3D Stacked ICs.

It is found that the ILP based algorithm can achieve lower test cost as compared to the other test planning schemes.

2. *Test flow selection* Finally, the impact of the selected test flow on a 3D Stacked IC is explored. We present a mathematical model to calculate the test time spent to produce every good package, given the test time and yield at every test instance. To reduce the test time we propose the Test Flow Selection Algorithm (TFSA) to arrive at the test flow with the lowest test time using a greedy approach.

It is confirms that the TFSA arrives at the same test flow when compared against the test flow obtained by exhaustive search for minimum test time.

1.5. Thesis Organization

The succeeding chapters of this thesis is organized as follows. Chapter 2, presents the previous research related to test planning and test flow of 3D Stacked ICs.

In Part II, test planning using different test architecture schemes has been addressed. In Chapter 3 test scheduling for 3D Stacked ICs with BISTed cores has been discussed. Chapter 4 presents test planning for 3D Stacked ICs using IEEE 1149.1 test architecture, while in Chapter 5 an IEEE 1500 based test architecture has been examined.

The third part if the thesis, Part III, addresses the problem of test flow selection for 3D Stacked ICs to minimize the test time. The test flow problem is defined and addressed in Chapter 6.

Finally, the thesis is summarized and concluded in Part IV. Possible research for the future is also discussed in this chapter.

2

Related Work

Extensive research has been conducted with the goal of minimizing the cost of production of ICs through the components representing cost of testing. Several publications addressing reduction of test cost by test planning, that involves test scheduling and test architecture design, or by selection of certain test flows, have been discussed here.

2.1. Test Architecture

In this section we discuss related work on test architecture optimization using several standards.

Test architecture design includes planning the tangible resources required for DfT. The resources include test pins, TAM, routing, silicon area, etc. to test the cores or modules within the IC. Several articles focus on optimizing the test architecture design in order to minimize the DfT hardware cost [29–36]. In addition, research is visible on test scheduling to minimize the cost associated with test time of ICs [20] [27] [37–43]. However, there is usually a trade-off between the DfT hardware and the test time. In other words, reduction of the DfT hardware cost usually results in increased test time and vice versa. Therefore, test planning takes into account the overall test cost resulting from the DfT hardware and the cost corresponding to the test time to find the most favorable trade-off.

Thorough research has been performed to reduce the test cost for non-stacked ICs [20] [27] [38] [39] [41–46] by co-optimizing the test architecture and the test time.

3D Stacked ICs, have lately attracted a fair amount of research [6] [7] [47–

49]. Recent research have addressed test architecture design for 3D Stacked ICs [13], testing the TSVs [6] [7] [13] [47–49] and 3D Stacked IC specific defects [6] [7].

Test planning to reduce the overall test cost for 3D Stacked ICs, considering both test time and test hardware has been addressed in [50] [51]. However, the proposed approach is rendered inefficient due to the inability to re-use wafer sort test hardware while performing package test, and is not scalable in case intermediate tests are necessary. In addition, for 3D Stacked ICs with TSVs, all test data commute via the lowermost chip [6] [7] [52] unlike illustrated in [50] [51], where the wide test data buses called TAMs start and end on any chip. Thus, it is important to provide a standardized and scalable test architecture for each chip forming the 3D Stacked IC, which reuses the test infrastructure during all test instances.

In the following subsections, research on test architecture optimization is discussed, addressing BIST, IEEE 1149.1 and IEEE 1500 based test architeures.

2.1.1. BIST

As more logic is packed into smaller form factors and the pin and probing area available for testing is rapidly decreasing. Traditional test approaches, like the automated test equipments, have limited resources for communication, thus increasing the test time required against the expected fault coverage. Therefore a DfT technique was proposed, the BIST, addressing the problem by incorporating additional circuitry during the design stage that enables the entire DUT to test itself. BIST can be implemented at the core, chip, board or system levels.

BIST helped to overcome several shortcomings of the ATE listed below:

- Test time is reduced as compared to ATE, as more tests can be performed concurrently.

- Can be implemented at different levels of hierarchy using same technique.

- Enables test during both manufacture or operation.

- The cost of an ATE can be reduced by using BIST.

- Unaccessible circuitry or IP cores can be easily tested using BIST.

- Specifically, in case of 3D Stacked ICs, BIST solves a lot of issues, including:

- Wafer probing specially in case of thinned wafers prior to bonding is a major concern for the manufacture of 3D Stacked ICs. With the use of BIST, the number of probe points can be minimized.

- TSVs in 3D Stacked ICs carry signals from the bottom to the topmost chip, and then back. Hence, testing intermediate stacks is a complicated process. BIST addresses the problem by localized testing.

Several papers address test cost reduction through test architecture design using a BISTed system [29] [30] [53–55]. In [53], Rajski *et. al.* propose a test architecture and algorithm for design flow to reduce test cost by by reducing scan test data volume and scan test time. Stroele in [54] present several algorithms for test scheduling. The objective is to minimize the hardware overhead for test control and test evaluation under several constraints. A framework for a common base for the algorithms was also proposed. The framework includes a cost function that allows to rate test schedules with respect to BIST hardware costs. The algorithm considers the trade-off between test time and area overhead required for BIST. Huang *et. al.* in [30] propose a cost-effective BIST scheme to test TSVs of 3D Stacked ICs. The scheme requires low test/diagnosis time and low silicon area cost, and arranges TSVs into arrays similar to memory. They discuss the area overhead and test time with respect to different array configurations, CMOS technologies, and TSV diameters. The area overhead and the required test/diagnosis time of the proposed BIST is found to be lower than using IEEE 1500 test architecture. However, due to unavailable data on test access for 3D Stacked ICs, the addressed researches could not be implemented on 3D Stacked ICs. Therefore, in this thesis, we present a test architecture for core-based 3D Stacked ICs, where all cores are BISTed, as part of the test planning approach.

Despite the advantages, BIST also possesses certain drawbacks that eventually require the use of several other test architecture standards.

- Area overhead caused by the additional DfT hardware. As a result the performance factor over the chip area is lower.

- Unlike the fixed cost of an ATE, BIST requires added cost for each unit produced. Therefore, for large scale productions, BIST may prove costlier than ATE.

- Owing to delays due to a longer critical path, the system is slower.

For core-based systems where each core is to be tested, the most effective way of reducing test application time is to perform core tests concurrently. However, it might not be possible to test all cores simultaneously due to constraints such as resource conflict or exceeding the safe amount of heat dissipation. Resource conflicts may occur when different cores share the limited hardware available for testing. On the other hand, performing tests concurrently leads to higher power consumption than performing them sequentially. The test power consumption must be kept under control [20], to avoid false test positives due to voltage drop or damage due to overheating. For core-based systems, Chou *et al.* [20] proposed a method to schedule tests in sessions while taking resource conflicts and power consumption into account. A session is defined as a group of tests that start simultaneously and no other tests are initiated until all tests of the session are finished. The concept of sessions simplify test scheduling under power constraints. That is because once the tests have been grouped in sessions, that are within the power limit, the schedule can be found by processing the sessions in any sequence.

In [29], Zorian discussed power constrained scheduling of tests for BISTed cores in a chip using sessions. In [20], Chou *et al.* proposes a solution for the same problem but also considers constraints, by the formation of sessions. The concept of sessions simplifies test scheduling. Muresan *et al.*, in [27], has developed an algorithm to schedule tests in sessions, while reducing the test application time for non-stacked chips under power constraints. The algorithm is described as follows:

- All core tests are sorted in descending order of their test times.

- The longest test is considered first, which forms the first session.

- While descending through the list of sorted core tests, each test is compared for power compatibility with the existing sessions.

- The test is included in the first (longest) session which is compatible in terms of power. If no prior power compatible sessions exist, the test forms a new session.

- Each test is considered in descending order of their length until all tests of the list are exhausted.

2.1.2. IEEE 1149.1

The IEEE 1149.1 test architecture standard was motivated by the increasing device density, ball-grid-array packaging, surface mount and

multiple layers, that have made the test of modern PCBs more complex and expensive by manifold. Finer pin spacing and use of double sided PCBs have limited traditional in circuit testing using bed of nails. Merging components from different suppliers required built in test delivery system that used a standard mechanism for all systems. Thus, the IEEE 1149.1 standard came into place that converted the outside nails of a bed of nails adapter into inside, electronic nails, providing access to pins without direct physical contact. It has also enabled serial read/write of test data to the core, and similar to BIST, can operate at different levels of hierarchy. The standard not only enabled components and interconnects to be tested independently, but the responses can be propagated to different components of the system from the DUT.

The standard provided several advantages:

- Serial input of test data to DUT

- Allows serial reading out of test results

- Can operate at chip, PCB and system levels

- Control of tri-state signals during testing

- Other chips can collect test response from DUT

- Components and interconnects can be tested separately

The IEEE 1149.1 standard is used for testing digital chips and interconnects between chips [2] [56–58]. Test architecture based on IEEE 1149.1 has been proposed for various embedded core-based systems [32] [33] [59]. For non-stacked ICs, much work is available which describes and optimizes test architecture using IEEE 1149.1 [34] [60]. Zhang *et. al.* in [34] utilizes the additional flexibility of a soft IP core to design an optimized TAP for an IEEE 1149.1 based system, whereas in [60], Cadwell *et. al.* perform a case study to conclude that using IEEE 1149.1 based test architectures for manufacturing tests is cost effective. Based on the IEEE 1149.1 test architecture, test scheduling for embedded core-based non-stacked ICs has been addressed in [31] [61]. In [31], a test sequence with reduced time is generated for detecting design faults for conformance testing or for detecting manufacturing or run- time defects and faults. The sequence is based on a functional-level FSM description of the circuit. It is generated using Rural Chinese Postman tours and Unique Input/Output sequences (UIO) approaches. A UIO sequence of a given state in a FSM is an I/O sequence of minimum

length starting from the given state which could not be produced by starting at any other state. Thus, UIO can be used to verify the initial state of an I/O sequence. The set of tests is assembled in an optimal manner, using the Chinese Postman problem, such that the test sequence (i) contains a test for each transition of the FSM, (ii) begins and terminates at a certain start state of the FSM, and (iii) requires a short time. Whetsel, in [61], describes an approach to reduce test time for IEEE 1149.1 based systems. This is achieved by grouping the data registers and adding an I/O serializer between the TAP and data registers. The methods are however not widely applicable on all non-stacked ICs, thus 3D Stacked ICs formed from different manufacturers would not perform well.

Despite the simplicity and convenience of use, several other standards have added up on the IEEE 1149.1 to enable parallel testing, thus reducing the test time. One such commonly used standard in the IEEE 1500.

2.1.3. IEEE 1500

The IEEE 1500 standard provides a platform for testing core-based non-stacked ICs with low test time and low test hardware overhead [2] [36] [43] [62–67]. Design and optimization of test architecture for non-stacked ICs with IEEE 1500 is described in [36] [42] [67]. Mullane *et al.* in [36] propose a hybrid scan for non-stacked ICs provided with IEEE 1500 core wrappers, by combining the serial and parallel ports of the wrapper, resulting efficient test vector access and reduced test time.

Various works proposing test architecture design and optimization for non-stacked ICs are seen in [13] [41] [52] [68]. Iyengar *et al.* [40] [41] [44] proposes methods of optimizing test wrapper and TAM for multiple core based non-stacked ICs, using the rectangular packing problem addressed in [69]. In [44] the objective of reducing both test time and the TAM width, the problem is approached in the following steps. First, a method is proposed the test wrapper design of each core. The scan-chains in each core are arranged in order of the time taken. Test wrappers are then formed by using the best fit decreasing (BFD) method of bin packing to lower the time taken to test the core. Next, the optimal assignment of cores to respective groups of TAMs is achieved using ILP over a given range of TAM widths. ILP is further used for the optimal partitioning of the TAM width among the cores. In [41], the methods are used to reduce tester data volume. Although the applicability of those approaches are limited to multiple cores on a single

non-stacked chip, they form the basis of the test architectures defined
for 3D Stacked ICs.

For 3D Stacked ICs, test architecture optimized for each chip in the
stack during wafer sort may not lead to an optimized test architecture
when all the chips are tested jointly during package test. Furthermore,
unlike 3D Stacked ICs with TSVs, where each chip in the stack sends
and receives data only via the lowermost chip, for 3D Stacked ICs not
bonded with TSVs, such as SiPs and PoPs, data can be communicated
to each chip via dedicated wire bonds. Thus, 3D Stacked ICs with TSVs
require a test architecture standardized over all the chips in the stack
that allows test access for all test instances. Jiang *et. al.* in [50] [51] have
proposed test architecture optimization for core-based 3D Stacked ICs
with TSVs. They propose a reduction in test cost while considering
both test time and DfT hardware as weighed factors of the test cost. In
case of 3D Stacked ICs with TSVs, all test data commute via the low-
ermost chip [6] [7] [52] unlike illustrated in [50] [51], where the TAMs
start and end on any chip. Furthermore, during wafer sort and pack-
age tests, in several instances, separate TAMs are used. This would
result in increase in test cost due to the unavailability of re-usable test
architecture, and the proposed approach is not scalable in case inter-
mediate tests are necessary. It is described by using an IEEE 1149.1
in the bottom chip and is scalable for any number of chips forming
the stack. The test access architecture proposed in [52] uses few TSVs
and supports both pre-bond and post-bond testing, but scheduling of
tests has not been considered. Therefore, in this thesis we address test
scheduling and test architecture design for 3D Stacked ICs based on
an IEEE 1500 test architecture, to minimize the test cost.

2.2. Test Scheduling

In this section we discuss related work on test scheduling.

Overall in this thesis, we approach test planning using all BIST, IEEE
1149.1 and IEEE 1500 standards.

The problem of test scheduling for non-stacked ICs to minimize the
test time has been addressed in several publications [20] [27] [29] [38].
Huang *et. al.* in [38] addressed optimization of test time while consid-
ering resource constraints for core-based SoC designs. The proposed
method is achieved in the following three steps:

1. Rectangular transformation is applied to cores having more I/O
   pins than the number of I/O pins of the SoC.

2. A modified best fit algorithm is applied, where if a rectangle can not be packed on an existing level, a transformation will be applied to see if it can be packed on some existing levels before a new level is created.

3. The minimum resource requirement given an upper bound on the test time can be solved using 2D bin-packing algorithm, by optimizing the width of the SoC for a given test time.

Although the studies in [13] [20] [27] [29] [38] [40] [41] [44] [52] [68] perform well for test planning of non-stacked chips under power constraints, they do not generate good results for 3D Stacked ICs. This is because, unlike non-stacked ICs, where the same test schedule applies for both wafer sort and package test instances, different test schedules are applied on the additional test instances for 3D Stacked ICs. To minimize the overall test time, a test scheduling approach much consider all test instances simultaneously. While there are a few different approaches to 3D Stacked IC manufacturing and packaging, typically a chip in the middle of a stack has no wire-bonds to package pins, but are accessed through the top or bottom chip in the stack and the TSVs of chip layers in-between.

2.3. Test Flow Selection

In this section we address previous work on test flow selection for 3D Stacked ICs.

It is essential to determine a test flow for a 3D Stacked IC that minimizes the test time to produce each good packaged 3D Stacked IC. The time must include the time required to test all faulty components in the process. However, the search space to determine an optimized test flow increases exponentially with each additional chip comprising the 3D Stacked IC [6]. For a 3D Stacked IC with N chips, the number of possible test instances are $2N$, shared among N wafer sorts, $N - 1$ intermediate tests and a package test. This gives rise to 2^{2N} possible test flows.

The impact of test flow is explored by several researchers [79–86]. Besides the individual test time of the constituent chips, the yield plays an important role in determining the test time required by a given test flow to produce each good packaged chip. A defect model for non-stacked ICs, based on yield and fault coverage has been proposed in [24]. Yield remains as a major limitation to the mass production of

3D Stacked ICs [6] [7] [11] [12] [87–89]. Methods to improve the overall yield for 3D Stacked ICs has been discussed in [87] [89–102]. Majority of the works discuss yield improvement in specific wafer-to-wafer or die-to-wafer scenarios. Yield improvement of individual chips and other manufacturing events, such as bonding is visible in [97]. Several papers address the impact of yield on the choice of test flow for 3D Stacked ICs [6] [88].

Several papers study the impact of determining a test flow for 3D Stacked ICs [84] [88] [91] [98]. Yield improvement has been discussed in [84] [98] for test cost reduction of 3D Stacked ICs for wafer-to-wafer and die-to wafer stacking processes. The works in [88] [91] have agreed on a generic test flow stating that performing wafer sort results in lower test times. However, an approach to obtain a test flow to minimize test time has not been discussed. Agrawal *et al.* have proposed a heuristic for arriving at a test flow that reduces the test cost where each component in the stack undergoes a predetermined number of tests using matrix partitioning method [79]. The paper explains an extensive model for test cost calculation of a given test flow. However, the paper assumes given cost and yield values for all components during test. For example, during intermediate tests the cost and yield of individual chips in the stack are considered. These values would be extremely difficult to extract during the actual production process as the cost and yield of individual chips are correlated after bonding.

In [103], a test cost analysis has been performed for 3D Stacked ICs, with up to six chips in a packaged stack. The yield of each die is assumed to be within a range of 60% to 90%, while the stack yield and TSV yield are assumed to be constant, 93% and 99%, throughout the paper. [103] compares the test flow of non-stacked ICs with 3D Stacked ICs. Case studies show that including wafer sort in the test flow results in reduction of the overall chip cost. In addition, it is concluded that fewer number of tests may not reduce the overall 3D Stacked IC cost and the test cost and cost loss also depends on the test yields of the intermediate partial stacks and the final stack before and after packaging.

As established by numerous researches, it can be seen that the test cost of 3D Stacked ICs depends greatly on the test architecture, as well as the test time. The test time in return varies not only with the test architecture, but also with the chosen test flow. Hence, in this thesis, we explore test cost reduction of 3D Stacked ICs by determining a suitable test flow.

Part II

Test Planning

In this part of the thesis, our work on test planning for 3D Stacked ICs is presented assuming the three test architectures: 1.BIST, 2. IEEE 1149.1 and, 3. IEEE 1500. For each test architecture a cost model is presented to calculate the cost related to a given test plan. A test planning scheme is proposed for each test architecture. It is seen that all test planning schemes reduce the test cost as compared to existing schemes (or naive schemes).

3

BISTed Architecture

In this chapter, we address test planning under power constraints for 3D Stacked ICs where each core is provided with a BIST engine. Unlike non-stacked ICs, where the same test schedule is applied both during wafer sort and package test instances, the package test of 3D Stacked ICs requires a test schedule to test all cores from all chips in the stack. Thus optimizing the wafer sort schedules may lead to sub-optimal test time. Therefore, we propose test scheduling approaches, with the objective of reducing the test time.

This chapter proceeds with presenting an overview of the system in Section 3.1. A cost model for 3D Stacked ICs using a BIST architecture is discussed in Section 3.2, followed by a motivating example in Section 3.3. The ReScheduling approach is proposed in Section 3.4 and experiments are performed on several benchmarked circuits, and the results are presented and discussed in Section 3.5 and finally the chapter is concluded in Section 3.6.

3.1. System Overview

Here we discuss the test architecture of the 3D Stacked IC assumed in
this chapter.

We assume a core-based 3D Stacked IC, where each core is provided
with a BIST engine. The BIST engines of the cores are connected to a
on-chip JTAG TAP through TDRs.

The test process is conducted as follows. The BIST engine is started
by using the JTAG TAP to shift in configuration data (possibly includ-
ing a seed for an LFSR) into a register within the BIST engine. After
the completion time of the core test, which is assumed to be known
for each core, JTAG is used to shift out the test response in the form
of a signature from a register within the BIST engine. Typically the
time required for shifting in configuration data and for shifting out
signatures is negligibly small compared to the time the BIST engine is
running to conduct the test. Therefore, only the BIST engine test time
is considered in this chapter.

A 3D Stacked IC with two chips, where each core is provided with a
BIST engine is depicted in Figure 3.1. Each chip in the stack comprises
of a JTAG TAP to access the cores vis TDRs. Only one TDR can be
accessed at a time. Figure 3.1 illustrates the TDRs as loops that start
from a JTAG TAP, proceeds through one or more BIST engines and
returns back to the JTAG TAP. Thus, if tests for more than one core
of a chip run concurrently in a session, the corresponding cores must
be connected in series on the JTAG interface, on a single TDR. The
number of TDRs is directly related to the DfT hardware cost. Therefore
we keep the number of TDRs low. It should be noted that this enforces
the session concept. For a single chip, only cores that are in the same
TDR can be tested concurrently. Furthermore, if two cores are tested in
sequence, in different sessions, they cannot be connected in the same
TDR. On the other hand, a session of tests (corresponding to a TDR)
from a chip can be performed concurrently with a session of tests from
another chip by selecting the TDRs in the TAPs of to the two chips.
While testing concurrently can lead to power dissipation above the
safe power limit W_{max}, scheduling must take power dissipation into
account.

The TSV interconnect between two chips may be tested by using the
JTAG TDR called the boundary scan register. The scan cells are trans-
parent when the 3D Stacked IC is in functional mode, but it test mode,
the scan cells are control points and observation points. Boundary
scan registers are implemented on both the chips and both are used in
TSV interconnect test. Test stimuli are applied on out-going TSVs and

test responses are captured on in-coming TSVs. Since the boundary scan register is a separate TDR, testing of TSVs cannot be performed concurrently with any other test.

It should be noted that the TSV interconnect tests will contribute with a constant term to the total test time and could not be scheduled with any other core tests, due to limitations of the JTAG. Therefore, TSV interconnect tests will not be regarded when addressing test scheduling in the remainder of the chapter.

3.2. Problem Formulation

Here we formulate the problem of test scheduling for core-based 3D Stacked ICs, assuming BIST for all cores. We start by introducing the notations used through a system architecture.

The notations are listed in Table 3.1, and are grouped among the given variables or the ones to be calculated. The system architecture is illustrated using the corresponding notations in Figure 3.1. A 3D Stacked IC with $N = 2$ chips in the stack is shown Figure 3.1. Chip 2 is staked over chip 1. Each chip contains a number of cores with their corresponding BIST engines. Chip 1 has three cores $C_1 = 3$, illustrated
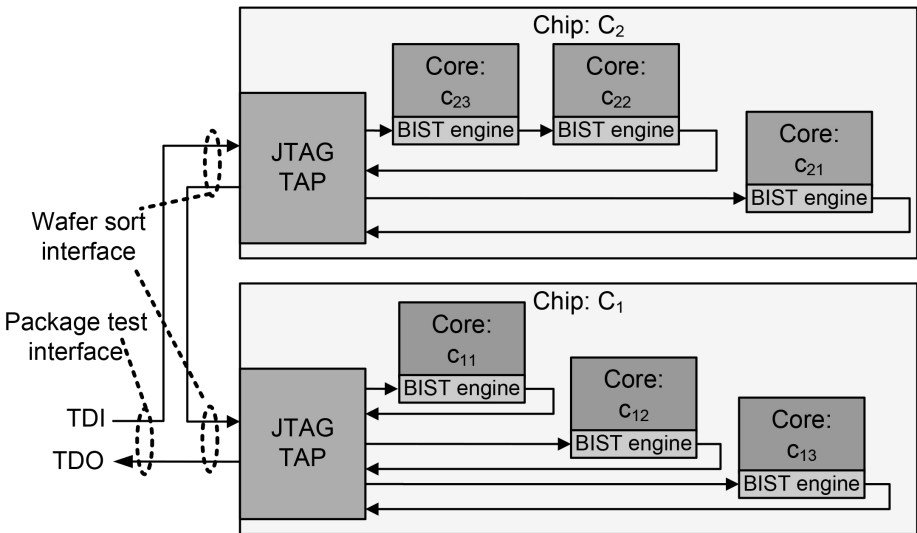
Figure 3.1.: DfT of a core-based 3D Stacked IC with BISTed cores

as c_{11}, c_{12} and c_{13}. Chip 2 has three cores as well $C_2 = 3$, illustrated as c_{21}, c_{22} and c_{23}. The cores are connected to a JTAG TAP, provided in each chip, by TDRs. It can be seen in Figure 3.1, that cores c_{11}, c_{12}, c_{13} and c_{21} have dedicated TDRs, whereas c_{22} and c_{23} share a common TDR.

Two parameters are associated with each core test: test time, t_{ij}, and power consumption, w_{ij}. The test controller, which in this case accesses the cores through JTAG , determines when the test for each core is initiated. Figure 3.2(a) shows a test schedule for the tests of the three cores of Chip 1, the power consumption at any moment is less than the maximum power consumption allowed, W_{max}, indicated by the (blue) horizontal line. The test schedules are illustrated by a rectangle corresponding to each core test, where the height represents power consumption, while the time taken by the test is represented by the width. The horizontal axis shows the time taken to perform the tests, and the vertical axis marks the power consumption. Two types

Table 3.1.: List of notations

Given	
N	Number of chips constituting the stack, $N \in \mathbb{N}_2$
i	i^{th} chip in the stack, $1 \leq i \leq N$
C_i	Number of cores on chip i
c_{ij}	Core j at chip i where $j \in 1..C_i$
t_{ij}	Time taken to test core c_{ij}
w_{ij}	Power dissipated by core c_{ij} during testing
W_{max}	Maximum power constraint
Calculated	
S_i	TDRs in chip i
s_{ik}	k^{th} session in chip i, where $s_{ik} \in S_i$
M_k	k^{th} package test session
T_{ik}	Test time of a session s_{ik} or M_k
W_{ik}	Power dissipated during testing session s_{ik} or M_k
Tws_i	Time taken to test the ith chip in the stack
Tpt	Time taken for package test of the 3D Stacked IC
TAT_{SP}	Time taken to test the 3D Stacked IC using Serial Processing
TAT_{PO}	Time taken to test the 3D Stacked IC using Partial Overlap
TAT_{RS}	Time taken to test the 3D Stacked IC using ReScheduling
W_t	Power dissipated during testing session s_{ik}
TAT	Total time required to test the 3D Stacked IC

of constraints are considered for the test schedule. The first constraint type is a resource constraint, as has been discussed in [27], which expresses that tests of cores which share some common resource cannot run at the same time. The second constraint type is a constraint regarding the maximum power consumption, W_{max}, which cannot be exceeded. The test schedule contains three sessions: s_{11}, s_{12} and s_{13}, as marked in Figure 3.2(a). The wafer sort test time is given as the sum of the time taken by each test session of the corresponding chip. Chip 1 takes 19 *time units* as illustrated in the horizontal time. The test time for Chip 1 is calculated as follows:

$$
\begin{aligned}
Tws_1 &= \sum_{k=1}^{3} s_{1k} \\
&= s_{11} + s_{12} + s_{13} \\
&= t_{11} + t_{12} + t_{13} = 5 + 8 + 6 \\
&= 19 \ \textit{time units}
\end{aligned}
\tag{3.1}
$$

Similarly, Chip 2 requires $Tws_2 = 9$ *time units* for wafer sort. Testing the 3D Stacked IC requires wafer sort of Chip 1 and Chip 2 and a package test of the stacked chip including tests of all cores in both Chip 1 and Chip 2. Consecutively, the package test time is given by the sum of the time taken to test each session at package test:

$$
Tpt = \sum_{\forall i} T_{ik}
\tag{3.2}
$$

The total test time is given as the sum of wafer sort of each chip and the package test times.

$$
\begin{aligned}
TAT &= \sum_{\forall i} Tws_i + Tpt \\
&= \sum_{\forall i,k} T_{ik} + \sum_{\forall k} T_{ik}
\end{aligned}
\tag{3.3}
$$

The objective is to minimize TAT while meeting the power constraint W_{max}.

3.3. Motivating Example

In this section we present an example motivating the need of test scheduling for core-based 3D Stacked ICs under power constraints.
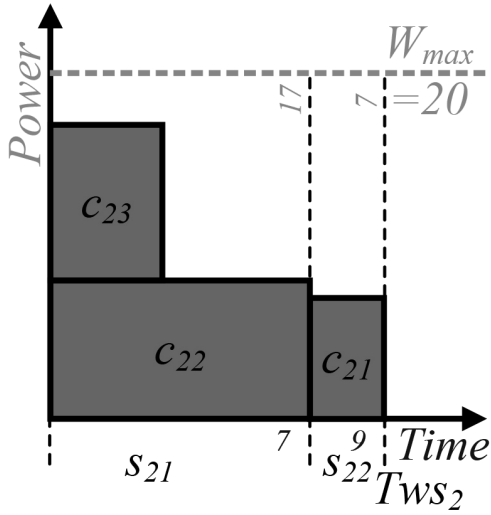
(a)
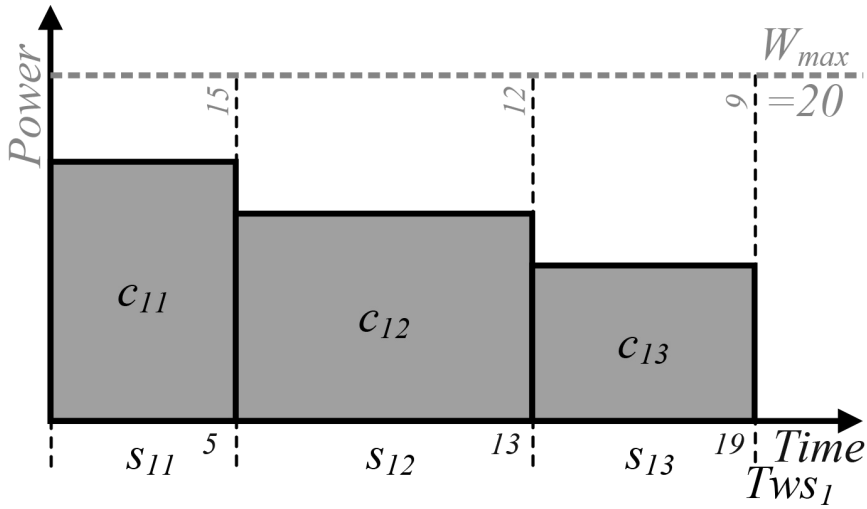
(b)

Figure 3.2.: Scheduling wafer sort of chips comprising
the 3D Stacked IC: (a) Chip 1 (b) Chip 2

Table 3.2.: Test time and power consumption
for core tests in Chip 1 and Chip 2

Chip	Core	Time	Power
i	c_{ij}	t_{ij}	w_{ij}
	c_{11}	5	15
1	c_{12}	8	12
	c_{13}	6	9
	c_{21}	2	7
2	c_{22}	7	8
	c_{23}	3	9
W_{max}		20	

A system is assumed similar to the illustration in Figure 3.1. The TDRs illustrated in the figure are disregarded, as they will be finalized along with the test schedule. The test durations and power consumption values for each core tests are provided in Table 3.2. The power constraint value is $W_{max} = 20$ units.

Prior to stacking chips into a 3D Stacked IC, each chip can be considered as individual non-stacked chips and the methods in [20] [27] apply for generating the wafer sort schedules. Figure 3.2 shows examples of the wafer sort schedules for the two chips, Chip 1 and Chip 2, from Table 3.2. The test schedule for Chip 1 contains a set of three sessions $S_1 = (s_{11}, s_{12}, s_{13})$ and the test schedule for Chip 2 contains a set of two sessions $S_2 = (s_{21}, s_{22})$ as shown in the figure. The test time for the schedules as obtained by [27] are $Tws_1 = 19$ and $Tws_2 = 9$ time units for Chip 1 and Chip 2, respectively.

In case the only knowledge of the wafer sort schedules consist of the test time for the schedules and the fact that the wafer sort schedules are within the power constraint, the limited knowledge available restricts the test schedules that are possible. In this case the package test is performed by executing the wafer sort schedules of chip serially, as illustrated in Figure 3.3. It should be noted that no tests from different chips are performed concurrently, because otherwise we would risk exceeding the power limit from lack of information of the actual power consumption. The time taken to run the package test schedule is equal to the sum of the time taken to test the individual chips. For the schedule in Figure 3.3, the overall test time can be calculated as:

$$TAT_{SP} = Tws_1 + Tws_1 + Tws_2 + Tws_2$$
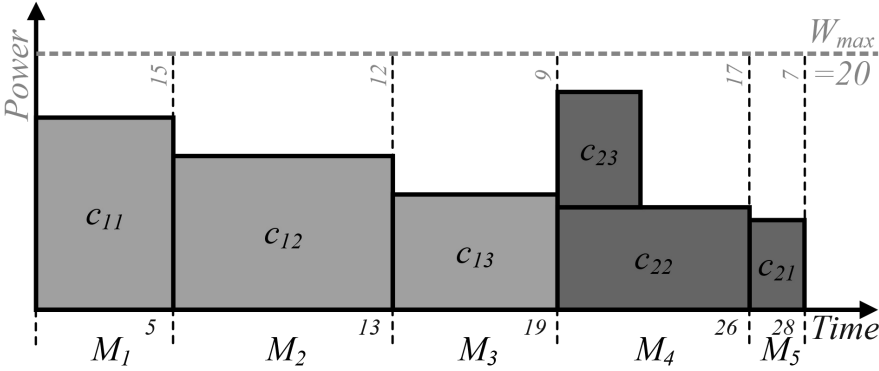
Figure 3.3.: Package test schedule obtained using by se-
rially performing wafer sort tests

$$= 19 + 19 + 9 + 9 = 56 \ \textit{time units}$$

If the maximum power reached by individual sessions and the test
time for the sessions are known, we can determine the power com-
patible test sessions of different chips that can be performed concur-
rently without exceeding the power constraint. Figure 3.4 illustrates a
package test schedule obtained by performing power compatible test
sessions from different chips concurrently. In the schedule for package
test, core c_{13} of Chip 1 (session s_{13}) and core c_{21} of Chip 2 (session s_{21})
are tested concurrently because they are power compatible. The wafer
sort schedule of the chips remain unchanged, but there is a reduction
in the total test time equal to the length of test $t_{21} = 2$ time units (cor-
responding to session s_{21}). For the schedule in Figure 3.4, the overall
test time is calculated as:

$$
\begin{aligned}
TAT_{PO} &= Tws_1 + Tws_2 + Tpt \\
&= 19 + 9 + 26 = 54 \ \textit{time units}
\end{aligned}
$$

When full knowledge is available concerning individual tests and
sessions of the wafer sort schedules, the knowledge can be utilized to
create a package test schedule to reduce test time. The correspond-
ing package test schedule may cause changes to the wafer sort sched-
ules. In this context, changing the wafer sort schedule means to split
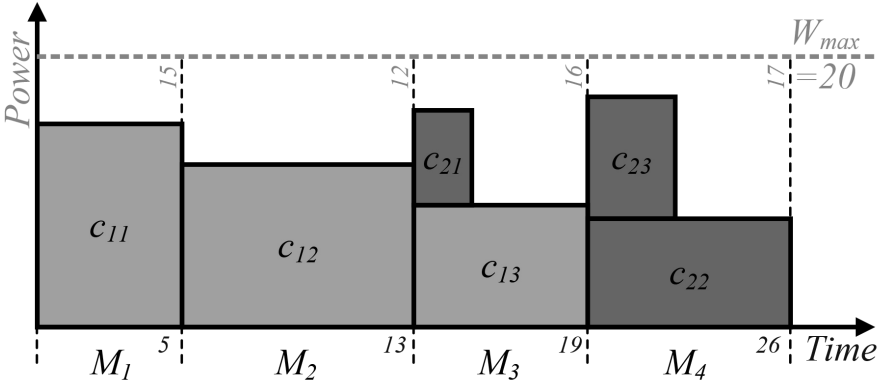
Figure 3.4.: Package test schedule obtained by merging power compatible sessions from different chips

a session and replace it with two new sessions. This means that the corresponding chips are redesigned so that a TDR is replaced by two TDRs in the test architecture. The benefit of splitting a session is that the two new sessions can be scheduled concurrently with sessions of the other chip during package test, if that reduces the overall test time. Figure 3.5 depicts the result of the approach. In the original wafer sort schedule (Figure 3.3), session s_{21} consisted of tests of cores c_{22} and c_{23}. In the wafer sort schedule, after rescheduling (Figure 3.5(b)), test of core c_{22} is performed serially with the test of core c_{12}, while core c_{23} is tested together with core c_{13}. This results in a reduction of test time for the package test equal to the duration of $t_{22} = 7$ time units as compared to that of TAT_{SP}. This approach results in splitting session s_{21} and renumbering the sessions, as shown in Figure 3.5(a), session s_{21} includes the test of core c_{22}, session s_{22} comprises core c_{23} and session s_{23} tests core c_{21}. But because of the splitting of the original session s_{21}, there is an increase in test time for Chip 2 wafer sort from $Tws_2 = 9$ time units to $Tws_2 + t_{23} = 9 + 3 = 12$ time units. The increase is equal to the duration of $t_{23} = 3$ time units, which is now tested serially with core c_{22}. Compared to TAT_{SP}, the reduction in the overall test time is equal to the sum of the durations of tests $t_{21} = 2$ and $t_{22} = 7$, minus the duration of test $t_{23} = 3$. For the schedule in Figure 3.5, the overall test time can be calculated as:

$$TAT_{RS} = Tws_1 + Tws_2 + Tpt$$

(a)

(b)
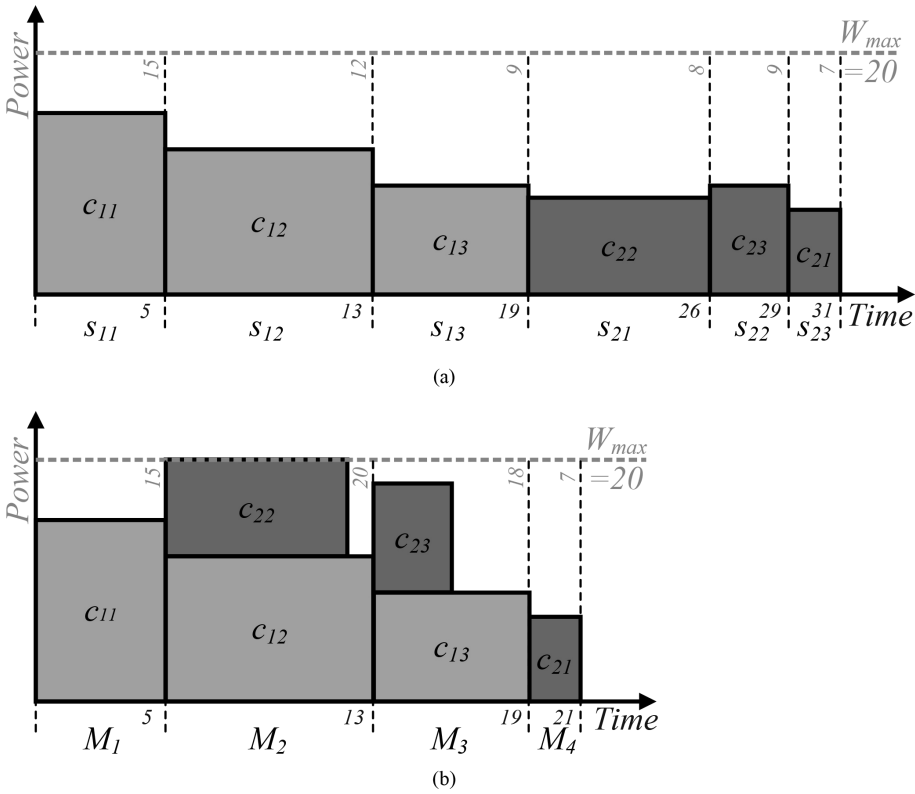
Figure 3.5.: ReScheduling: (a) wafer sort schedule (b) package test schedule

$$=19 + 12 + 21 = 31 + 21 = 52 \; time \; units$$

From the above illustration, it can be seen that TAT_{RS} required a lower overall test time as compared to TAT_{SP} and TAT_{PO}, as is shown in Figure 3.5. However, in contrast to TAT_{SP} and TAT_{PO}, TAT_{RS} can lead to an increase in the routing of JTAG interconnect lines, as a result of splitting sessions, which means additional TDRs. Therefore, in the following section we describe an approach for TAT_{RS}, while taking into account the additional routing that results from splitting sessions.

Table 3.3.: Maximum possible time reduction of sessions

		Chip 1		
		s_{11}	s_{12}	s_{13}
Chip 2	s_{21}	0	(3)	2
	s_{22}	0	0	(5)

Table 3.4.: *TAT* reduction vs. increase in number of additional TDRs

Schedule no.	1	2	3	4	5
TAT **reduction**	8	2	3	5	0
Increase in TDRs	1	1	1	0	0

3.4. Proposed Approach: ReScheduling

We define three different approaches for test scheduling depending on the available knowledge from wafer sort. In this chapter, three approaches called Serial Processing (SP), Partial Overlapping (PO) and ReScheduling (RS), are presented. In addition, we also discuss the complexity of the approaches.

3.4.1. 3D Stacked IC with two chips in the stack

The following describes the RS approach which is achieved in three broad steps as mentioned below:

1. Scheduling wafer sort of each chip comprising the 3D Stacked IC.

2. Tabulate the reductions (if any) in test time by rearranging the tests of cores from all possible pairs of sessions, where each session belongs to to a different chip.

3. Find the maximum reduction in test time, using the table, by choosing multiple ReScheduling instances.

I. *Wafer sort schedule*

The cores of each chip are scheduled for wafer sort as per the heuristic defined in [27]. The heuristic is illustrated in Algorithm 1. We explain the heuristic by scheduling the tests for Chip 2.

First, we sort all cores in order of their respective test times t_{ij}. For Chip 2, the sorted cores would be c_{22}, c_{23} and c_{21} respectively. Next,

---

Algorithm 1 Wafer sort schedule of Chip i

1: Given: c_{ij}, t_{ij}, w_{ij} for $1 \le j \le |C_i|$
2: **Reenumerate**: c_{ij} in descending order of t_{ij}
3: **for** $j = 1$ to $|C_i|$ **do**
4:         Number of sessions: $K = 1$
5:         **while** W_{max} is met **do**
6:                 **for** $k = 1$ to K **do**
7:                         **Include** Core c_{ij} in session s_{ik}
8:                         **Break**
9:                 **end for**
10:         **end while**
11:         **if** Core $c_{ij} \notin$ session s_{ik}, $\forall k$ **then**
12:                 **New session** Core $c_{ij} \in$ session $s_{i\ K+1}$
13:                 $K = K + 1$
14:         **end if**
15: **end for**
16: **Save**: s_{ij} $\forall j$

---

each core is considered, in turn, for inclusion in the existing sessions, or forming a new session. In case of Chip 2, core c_{22} is considered first. Since there are no existing sessions for comparison, c_{22} forms the first session s_{21}. Core c_{23} is considered next, and included in session s_{21}. As the power constraint $W_{max} = 20$ is not violated, core c_{23} is finalized in session s_{21}. Lastly, core c_{21} is taken into account and tried in session s_{21}. However, due to a violation in the power constraint, core c_{21} is tried in the next available session. As there are no other existing sessions, core c_{21} forms session s_{22}.

The complexity of the scheduling approach is given to be $O(C^2)$, as per [27].

II. *Constructing the pair-wise session ReScheduling (PSR) table*

The objective is to tabulate the results for the pair-wise ReScheduling of sessions (PRS) table, illustrated by Table 3.3. The columns represent the sessions of Chip 1 and the rows represent the sessions of Chip 2. Each entry in the table indicates the reduction in test time that is possible by rescheduling the corresponding session from Chip 1 (a specific column) and Chip 2 (a specific row). The idea of creating the table is that a schedule can be defined by selecting a unique entry in each column and each row. These selected entries correspond to

---

Algorithm 2 ReScheduling a pair of sessions from different chips

---

 1: **Given from Algorithm 1:** $c_{1i} \in s_{1a}$ and $c_{2j} \in s_{2b}$
 2: **Define** New empty package test sessions X, Y
 3: **Define** New empty wafer sort sessions X_1, X_2, Y_1, Y_2
 4: **for all** $c_{ij} \in [s_{1a} \cup s_{2b}]$ **do**
 5:     **Sort** by descending order of t_{ij}
 6: **end for**
 7: **for all** $c_{ij} \in [s_{1a} \cup s_{2b}]$ **do**
 8:     **while** W_{max} is met **do**
 9:         **Move** Core c_{ij} to session X
10:     **end while**
11:     **Move** remaining cores c_{ij} to session Y
12: **end for**
13: **if** $Time(X, Y, X_1, X_2, Y_1, Y_2) < 2 \cdot Time(s_{1a}, s_{2b})$ **then**
14:     **Store** $Time(s_{1a} + s_{2b}) - Time(X + Y)$ in PRS Table 3.3
15: **elseStore** 0 in PRS Table 3.3
16: **end if**

---

sessions that are rescheduled in the defined schedule. Sessions for which no entry was selected will be added to the defined schedule.

Algorithm 2 depicts the steps leading to the ReScheduling of a pair of sessions from different chips. The key idea is to group the tests of two wafer sort sessions from different chips in two sessions for package test such that the long tests are grouped together and the short tests are grouped together. This way, there will be one long test session and one short test session, instead of the previous two long sessions. We assume any two sessions, s_{1a} and s_{2b}, from the original wafer sort schedules of two different chips, Chip 1 and Chip 2, respectively. The tests of s_{1a} and s_{2b} are then arranged in descending order of their test times. A session for package test, X, is produced along with its respective wafer sort sessions in either chip, sessions X_1 and X_2, which will eventually replace the existing wafer sort sessions s_{1a} and s_{2b}. Starting with the first test in the sorted list, *i.e.*, the core with the longest test time, tests are included in session X, as long as the power constraint W_{max} is met. As soon as W_{max} is exceeded, a new final-test session Y, is created with the remaining tests. It should be noted that if all tests can fit in session X without exceeding the power limit, it would lead to PO of the wafer sort sessions s_{1a} and s_{2b}. Final-test sessions X and Y are disjoint sets of tests that together contain all the tests from sessions s_{1a} and s_{2b}. The wafer sort sessions s_{1a} and s_{2b} are replaced with test

sessions X_1, X_2, Y_1 and Y_2, according to how the tests were allocated in X and Y. It should be noted that some of the sessions Y, X_2, Y_1 and Y_2 may become empty of tests and can be disregarded. The modified overall test time is calculated. If the time taken by the new test schedule, including the wafer sort tests X_1, X_2, Y_1 and Y_2, and the package tests X and Y, is shorter than the test schedule for SP, *i.e.,* twice the sum of the time taken by sessions s_{1a} and s_{2b}, the value is included in the PRS table, Table 3.3 column a and row b. For no reduction in test time the value is set to zero. While adding values to Table 3.3, extra considerations are required if both chips are of the same design. Then, if a session is split, it should affect both chips, because sessions correspond directly to JTAG TDRs which are specific to the chip design. If the chips are of the same design and the session pair for rescheduling indicates a schedule that requires two different designs, rescheduling that session pair is infeasible and the entry in Table 3.3 is set to zero.

The process described above is repeated for all possible combinations of two sessions from the wafer sort schedules of the two chips. As each session in Chip 1 is ReScheduled with each session in Chip 2, there are $C_1 \cdot C_2$ entries in the table.

III. *Deciding on session pairs to ReSchedule for final test schedule*

In the last step, a schedule is defined with the maximum reduction in the total test time compared to SP by considering the PRS Table. Table 3.3 shows the possible reduction in the total test time as a result of rescheduling a session of Chip 1, as denoted by the column number, with a session of Chip 2 of the corresponding row number. Given a table such as Table 3.3, a schedule is generated by rescheduling each session of one chip with different sessions of the other chip, such that every session is considered only once. The sessions that are not rescheduled are added to the final schedule without any modification. For example, in Table 3.3, two session pairs are selected, namely s_{12} with s_{21} and s_{13} with s_{22}. s_{11} is not included in any of the selected pairs and is added to the final schedule unmodified.

A key observation regarding the PRS Table is that pairs of sessions can be handled independently. If combining a pair of sessions leads to a reduction in the overall test time compared to the test schedule in SP, a new test schedule can be constructed by combining several independent session pairs. The total reduction in the overall test time can be summed up from the reductions in test time when all session pairs have been considered, while each session has been taken into account only once.

The objective is to find the combination of rescheduled session pairs, which would give the lowest overall test time. For example, with respect to Figure 3.3, considering s_{12} from Chip 1 and s_{21} from Chip 2 results in a reduction of 3 time units with ReScheduling, compared to the time required to perform the original s_{12} of Chip 1 and s_{21} of Chip 2 sequentially, as in SP. In case of PO, where no sessions are split, the values in the table would either be zero (when the sessions are not power compatible), or equal to the length of the smaller session. For example, it was not possible to reduce the overall test time by combining s_{11} with s_{21} as marked by 0 in Table 3.3. The test schedule and the total reduction in the overall test time are obtained by rescheduling each session of Chip 1, with sessions of Chip 2. As discussed before, tests from s_{12} of Chip 1 and tests from s_{21} of Chip 2 upon rescheduling, result in a reduction of 3 time units (marked with (3) in Table 3.3). Similarly, rescheduling s_{22} of Chip 2 with s_{13} of Chip 1 give a reduction of 5 time units. The sessions that result from the marked session pairs are included in the final-test schedule with the summed total of test time reduction adding up to $3 + 5 = 8$ time units. The test time of the rescheduled session pairs are added to the remaining sessions to give the overall test time. Thus, the final-test schedule has s_{11} in series with the combination of s_{12} with s_{21} and s_{13} with s_{22}. From this it can be seen that RS results in a reduction in the overall test time by 8 time units, as was predicted by selecting entries worth 5 and 3 time units in Table 3.3.

Arriving at the test schedule using the PRS Tablewith the lowest the overall test time is complex. The combination of session pairs that give the lowest overall test time on rescheduling could be found by comparing of all possible combinations of session pairs in the PRS Table. However, the number of possible combinations can be prohibitively large. To arrive at the complexity of exploring all possible schedules from the PRS Table, Table 3.3, there are S_1 columns and S_2 rows, assuming $S_1 < S_2$. The first value can be chosen from among $S_1 \cdot S_2$ values. For the second value, the corresponding row and column of the first value is not considered. Therefore, there are $(S_1 - 1) \cdot (S_2 - 1)$ values, and so on. For the ith value, there are $(S_1 - 1) \cdot (S_2 - 1)$ choices. Hence, the search space can be calculated to be $\sum_{i=0}^{S_1} \{(S_1 - i) \cdot (S_2 - i)\}$. Thus, it can be seen that the problem of selecting session pairs from the PRS Table 3.3 to explore all possible test schedules is difficult.

Existing heuristics can be applied to obtain a schedule with low overall test time (but not necessarily lowest) from Table 3.3. The heuristic that has been used selects the table element with the highest value

and continues to select the table element with the next highest value, while restricting the selection to columns and rows that are not corresponding to a previous selection. The heuristic ensures that only independent session pairs are selected. The process continues until all rows are exhausted. The sum of all the values corresponding to selected session pairs give the net reduction in test time that is achieved by the rescheduling the selected session pairs. Sessions that were not joined with other sessions are added to the list of session pairs to form the schedule. The particular combination of session pairs that lead to the schedule correspond directly to the wafer sort and package test schedules for the 3D Stacked IC. The combination of session pairs that gives the largest reduction in terms of the overall test time corresponds to a candidate for the schedule.

To arrive to the schedule, the heuristic is iterated K times, where K is the sum of the number of rows and columns, with different session pairs (not necessarily the element with the highest value) as starting point to produce a number of solutions that can be evaluated by the designer of the 3D Stacked IC with regard to the acceptable amount of JTAG interconnect line routing. This results in Table 3.4 for the considered example. Schedule 1 is the result of combining s_{12} with s_{21} as well as s_{13} with s_{22}. Schedule 2 is the result of combining s_{12} with s_{22}.

RS of sessions resulting in a reduction of the overall test time can lead to a corresponding increase in the number of TDRs due to splitting of sessions, and consequently more routing of JTAG interconnect lines. Table 3.4 shows an example providing the reduction in the overall test time and the number of additional TDRs for five of the test schedules produced by the proposed RS approach.

3.4.2. 3D Stacked IC with N chips in the stack

To perform power-constrained test scheduling for 3D Stacked ICs with more than two chips in the stack using the approach described in Section 3.4.1, the following generalizing step is applied.

By using the approach described in Section 3.4.1 for the first two chips (say Chip 1 and Chip 2), a final-test schedule is defined. By abstracting from the fact that it is the final-test schedule is for two chips, it can be considered as a wafer sort schedule for a single chip, Chip 1∗, that contains the cores of Chip 1 and Chip 2. The procedure is illustrated in Figure 3.6. The same approach (described in Section 3.4.1) can again be applied to add another chip, Chip 3, to the test scheduling process. By applying the abstraction, the table created in the second
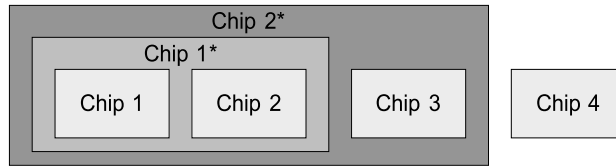
Figure 3.6.: Generalization process by abstracting from already processed chips

step of the approach from Section 3.4.1 will remain two-dimensional. The process can continue by adding chip after chip until all the chips of the stack are included, as shown in Figure 3.6 for four chips. The final-test schedule for the 3D Stacked IC consists of the sessions that are defined when the last chip is processed. The wafer sort schedules for the individual chips (now without the abstraction) are found by removing all tests but the ones belonging to the considered chip from the final-test schedule (sessions that become empty while removing tests are disregarded).

3.5. Experiments

To demonstrate the benefits of the proposed test scheduling approach, this section describes an experiment to compare *TAT* achieved by PO and RS with *TAT* achieved by the straight forward SP approach, which is used as baseline. In the Table 3.5, the following notation is used:

- Z : ASIC Z

- L : SYSTEM L

- M : Muresan's design

- SP : Serial Processing

- PO : Partial Overlapping

- RS : ReScheduling

- $R = \frac{T_{SP} - T_{RS}}{T_{SP}}$: Reduction

The experiments are performed with the circuits ASIC Z [29], System L [46] and Muresan [27] (marked by Z, L and M respectively in Table 3.5 and Table 3.5) and these circuits were used to create 3D Stacked

ICs. These designs are seen as single-die chips and have 9 [29], 14 [46] and 10 [27] cores, respectively. To enable experiments, the Muresan design and System L were scaled to have the same power limit W_{max} as ASIC Z. The test duration and power consumption were scaled with the same factor.

To make a 3D Stacked IC, a number of the three single-die chips are combined to form a stack. The column marked Design shows the single-die chips that form the stack. The group of four columns marked Total time for wafer sort, shows the test times for SP, PO and RS for the wafer sort schedules of the stack. The third column in the group shows the relative reduction in wafer sort test time of RS compared to SP. It should be noted that a negative reduction is an increase. The next group of four columns marked Time taken for package test, shows the test times for the final-test schedules generated by SP, PO and RS, and gives the relative amount of final-test time reduction achieved comparing the results for SP with the result for RS. The group of columns marked TAT includes the sum of the wafer sort times and package test times. The first three columns in the group show TAT for the SP, PO and RS approaches, respectively. The relative reduction in TAT is shown in the last column where RS is compared against SP. The right-most column of Table 3.5, shows the relative increase in the number of TDRs that result from splitting sessions in the RS approach. The number of TDRs for the SP approach is shown in parenthesis.

As the RS approach yields a table such as Table 3.4 with several different test schedule solutions where the acceptable number of TDRs determines the test schedule selection, the experiment is performed with the test schedule that results in the largest TAT reduction (8 time units in the case of Table 3.4). The initial wafer sort schedules were generated by the approach in [27] and our approaches were applied for generating the package test schedule. The approach proposed in Section 3.4 was used to find the maximum reductions in TAT while considering the number of TDRs as the number of sessions in the example designs were in a reasonable range.

From Table 3.5, it can be seen that RS can achieve up to 41.5% reduction in the final-test schedule time in comparison to SP, when two chips of System L are stacked to form the 3D Stacked IC. This result can be explained by a high power constraint, which enables a beneficial final-test schedule where many core tests are performed concurrently. In particular for the LL design, one session was split, resulting in an additional TDR and an increase in the wafer sort schedule duration. The reduction in TAT was 18.5%, while the amount of TDRs are increased by 3.8%. For three System L chips, LLL, the final-test schedule

Table 3.5.: Reduction in test time against increase in the number of TDRs for 3D Stacked ICs with up to four chips

Design	Time taken by wafer sort				Time taken for package test				TAT Wafer sort + package test				TDR increase	
	T_{SP}	T_{PO}	T_{RS}	$R(\%)$	T_{SP}	T_{PO}	T_{RS}	$R(\%)$	T_{SP}	T_{PO}	T_{RS}	$R(\%)$	%	init
2-chip 3D Stacked ICs														
ZZ	600	600	600	0	600	562	562	6.3	1200	1162	1162	3.2	0	6
LL	2050	2050	2224	-8.5	2050	1412	1199	41.5	4100	3462	3343	18.5	3.8	26
MM	3900	3900	3900	0	3900	3900	3300	15.4	7800	7800	7200	7.7	0	10
ZL	1325	1325	1325	0	1325	958	958	27.7	2650	2283	2283	13.9	0	16
LM	2975	2975	2975	0	2975	2530	2530	15.0	5950	5505	5505	7.5	0	18
MZ	2250	2250	2250	0	2250	2212	2212	1.7	4500	4462	4462	0.8	0	8
											Average:	8.6	0.6	
3-chip 3D Stacked ICs														
ZZZ	900	900	900	0	900	862	862	4.2	1200	1160	1160	3.0	0	9
LLL	3075	3075	3597	-17.1	3075	1799	1199	61.0	6150	4874	4796	22.1	5.1	39
MMM	5850	5850	5850	0	5850	5850	5250	10.3	11700	11700	11100	5.2	0	0
ZML	3275	3275	3275	0	3275	2724	2724	16.8	6550	5999	5999	8.4	0	21
											Average:	9.7	1.3	
4-chip 3D Stacked ICs														
ZMLZ	3575	3575	3575	0	3575	2896	2896	19.0	7150	6471	6471	9.5	0	24
ZMLM	5225	5225	2966	0	5225	4634	4634	11.3	10450	9859	9859	5.7	0	26
ZMLL	4300	4300	4474	-4.0	4300	3624	3411	20.7	8600	7924	7885	8.3	2.9	34
											Average:	7.8	0.9	
					Average over all designs considered							8.8	0.9	

time is 61.0% and the TAT reduction was 22.1%. It should be noted that other 3D Stacked ICs consisting of two identical chips (such as the pair of ASIC Z chips, denoted by ZZ) does not lead to the same result. For the 3D Stacked IC design made up by a pair of ASIC Z chips, TAT was reduced by 3.2% and RS and PO achieved the same result. This corresponds to a case when it is not possible to reduce TAT by splitting sessions. Four experiments led to splitting of sessions, which increased the number of TDRs, as can be seen in the right-most column of Table 3.5. For the other experiments, the reduction in TAT was achieved without splitting sessions and the best result achieved without splitting sessions was 13.9% reduction in TAT for design ZL.

It can be calculated from Table 3.5, for the sub-column $R(\%)$ under

TAT, that the average reduction for the overall *TAT* is 8.8% for the twelve stacks considered, while the average increase in the amount of TDRs is 0.9%.

3.6. Discussion

In this chapter, the problem of power-constrained test scheduling for 3D Stacked ICs with TSVs has been addressed for the first time. It is shown that the test planning for 3D Stacked ICs with TSVs is different, compared to the test planning for non-stacked ICs, and requires specific test scheduling solutions. Based on a proposed test cost model, the chapter proposes two test scheduling approaches, PO and RS that reduce test application time while taking power-constraints and the need to route JTAG and TDRs into account. Experiments done with the two scheduling approaches and a straight forward approach (SP) with several benchmarks show up to 22% reduction in test application time and an average reduction of 9% in test application time with less than 1% average increase in the amount of TDRs over the SP scheme.

4

IEEE 1149.1 Based Architecture

In this chapter, we address test planning for scan-based 3D Stacked ICs with an IEEE 1149.1 test architecture. The test plan, which reduces the overall test cost is achieved by co-optimizing test time and the DfT architecture. We define a cost model to arrive at the test cost for a given test plan for non-stacked ICs and 3D Stacked ICs. An algorithm is proposed that arrives at a test plan with reduced test cost by considering both wafer sort and package test instances simultaneously. The algorithm was implemented and experiments were performed on several designs formed by ITC02 benchmark designs. The results were compared against that obtained by Simulated Annealing method. It was observed that the proposed test planning algorithm arrived at a low cost, marginally higher than with Simulated Annealing, but with considerably lower computational time. The algorithm is first implemented on non-stacked ICs and then extended to 3D Stacked ICs.

This chapter proceeds with presenting the assumed mechanism of the system in Section 4.1. A cost model for 3D Stacked ICs using a IEEE 1149.1 test architecture scheme is discussed in Section 4.2, followed by a motivating example in Section 4.3. A test planning method proposed in Section 4.4 and experiments are performed on several benchmarked circuits, and the results are presented and discussed in Section 4.5.

4.1. Test Process

In this section we first illustrate the test process for core-based non-stacked ICs followed by 3D Stacked ICs provided with the IEEE 1149.1 test architecture.

4.1.1. Non-stacked IC

The test architecture of a non-stacked IC, which is assumed in this chapter, is shown in Figure 4.1. A chip is considered to consist of a number of cores that are accessed by an on-chip IEEE 1149.1 infrastructure [104]. In Figure 4.1, the chip consists of three cores: Core1, Core2 and Core3. The IEEE 1149.1 TAP may have up to five terminals, namely Test Data Input (TDI), Test Data Output (TDO), Test Mode Select (TMS), Test Clock (TCK) and an optional Test Reset (TRST) as seen at the bottom of Figure 4.1. The scan chain of each core is accessed by the TAP controller via TDRs. Core1 and Core2 are on TDR1, whereas Core3 has a dedicated TDR2 in Figure 4.1. If tests for more than one core of a chip are to run concurrently in a session, these cores are connected in series on the IEEE 1149.1 interface, via a single TDR. Only one TDR can be accessed at a time. This enforces the session concept that was introduced in Section 2.2. If two cores are to be tested in sequence, in different sessions, they are not connected in the same TDR. Thus, if tests for more than one core of a chip are to be executed concurrently in a session, as shown in Figure 4.2, these cores are to be connected in series on the IEEE 1149.1 interface in one TDR. Since Core1 and Core2 are tested in the same session, denoted by $(1, 2)$, as in Figure 4.2, the two cores are connected to the TAP controller by the same TDR, as seen in Figure 4.1. Correspondingly, in Session2, denoted by (3), only Core3 is tested, which is connected to the TAP controller by a dedicated TDR.

4.1.2. 3D Stacked IC

Figure 4.3 illustrates a core-based 3D Stacked IC with two chips, where Chip2 has been stacked on top of Chip1. Chip1 contains Core1, Core2 and Core3, while Chip2 hosts Core4 and Core5.

The test application process during package test is as follows. During package test of the 3D Stacked IC in Figure 4.3, the TDOup of Chip1, *i.e.*, the lower chip in the stack, serves as the TDIdown of Chip2, *i.e.*, the chip on top . The TDOup of the topmost chip is directed out via the TSVs by the TDOdown of Chip1. The TDIup of the lowermost chip, the TDOdown of the lowermost chip, TMS, TCK and an optional TRST serve as the package test interface for the 3D Stacked IC. A session of

tests from one chip can be performed concurrently with a session of tests from another chip by selecting the corresponding TDRs by the respective on-chip TAPs of to the two chips.

The test process for 3D Stacked IC is conducted as follows.

1. A test stimulus is provided to configure the TAP of each chip of the 3D Stacked IC. The chips containing the cores to be tested in the respective session are set to INTEST, while the remaining chips are configured in BYPASS mode.

2. A boundary-scan test instruction is shifted into the IR of the lowermost chip (Chip1, in Figure 4.3) through the TDIdown.

3. The instruction is decoded by the decoder associated with the IR to generate the required control signals in order to properly configure the test logic.

4. A test pattern is shifted into the selected TDRs through the respective TDIdown of the chips and then applied to the core(s) to be tested.

5. The test response is captured in a TDR.
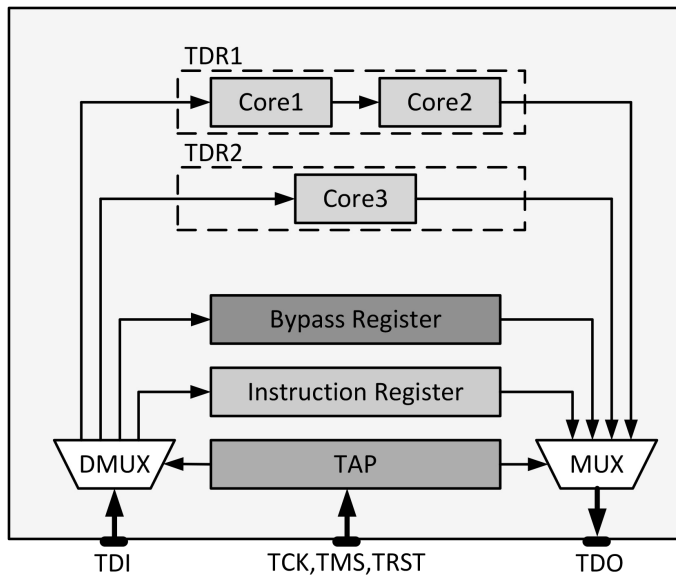
Figure 4.1.: Test architecture of a non-stacked chip with IEEE 1149.1
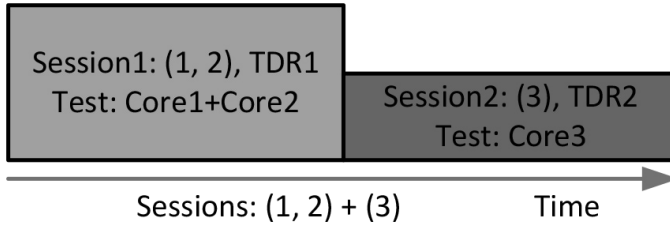
Figure 4.2.: Sessions formed by core tests

6. The captured response is shifted out through the TDOup of the chip, which acts as the TDIdown of the chip above (Chip2, in Figure 4.3). The test response, is shifted out through the TDOup of the topmost chip (Chip2, in Figure 4.3), which takes a U-turn and exits each chip through the TDOdown.

7. Each test response exits the 3D Stacked IC via the TDOdown of the lowermost chip. At the same time, a new test pattern can be scanned in through the TDIdown of the lower chip (Chip1, in Figure 4.3).

8. Steps 4 to 7 are repeated until all test patterns are shifted in and applied, and all test responses are shifted out.

9. The interconnects are tested by configuring the TAP of each chip in the 3D Stacked IC in EXTEST mode. Hence, no TDRs are selected while the interconnects are tested.

The TSV interconnect between two chips may be tested by using a TDR called the boundary scan register, which connects all input/output pins and TSVs with special scan cells forming a shift register. The boundary scan cells are transparent when the 3D Stacked IC is in functional mode, but in test mode, the boundary scan cells are control points and observation points. Boundary scan registers are implemented on both the chips and both are used in TSV interconnect test. Test stimuli are applied on out-going TSVs and test responses are captured on in-coming TSVs. For example in Figure 4.3, test patterns are applied through the TDIdown of Chip1, and the test response is captured on the TDOup of Chip2. Since the boundary scan register is a separate TDR, testing of TSVs cannot be performed concurrently with any other test.

It should be noted that the TSV interconnect tests will contribute with a constant term to the test time and could not be scheduled with
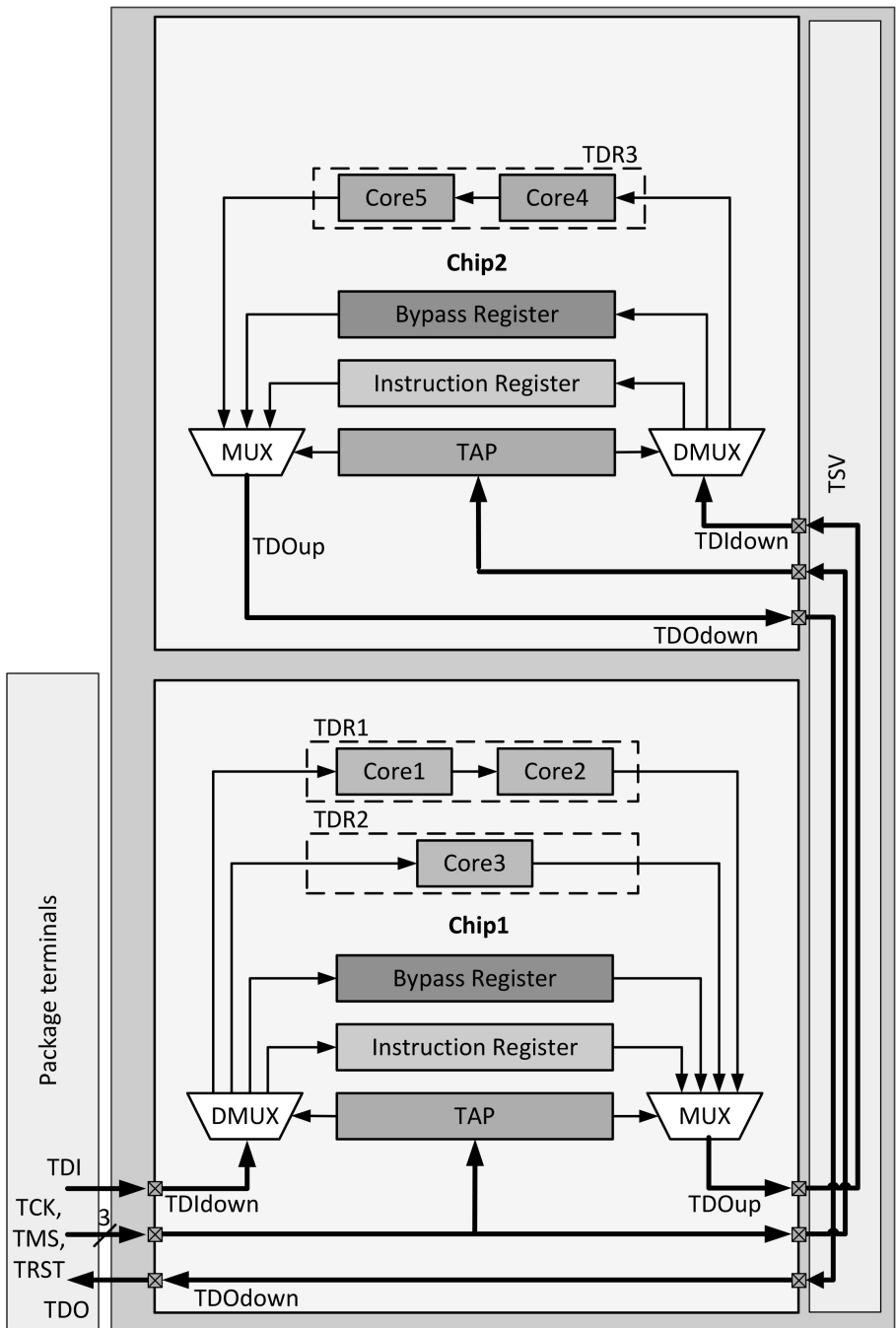
Figure 4.3.: Test architecture of 3D Stacked IC with a IEEE 1149.1

any other core tests, due to the IEEE 1149.1. Therefore, TSV interconnect tests will not be regarded when addressing test scheduling in the remainder of the chapter.

4.2. Problem Formulation

In this section we formulate the test planning problem for core-based non-stacked ICs, followed by that of 3D Stacked ICs.

4.2.1. Non-stacked IC

Here we formulate a mathematical framework for test planning of core-based non-stacked ICs assuming a test architecture using the IEEE 1149.1 standard. We start by introducing the notations used through a system architecture, followed by the test mechanism, and finally a cost model.

For a non-stacked IC supported by the IEEE 1149.1 test architecture,

Table 4.1.: List of notations for non-stacked IC

Given	
C	Set of cores in the IC
c	Each core $c \in C$
$l(c)$	Length of scan chain of core c
$p(c)$	Patterns required by core c
$w(c)$	Power dissipated by core c
w_{max}	Power constraint
α	Designer specified constant to estimate the cost of time
β	Designer specified constant to estimate the cost of DfT hardware
Calculated	
H	Set of TDRs
h	Each TDR $h \in H$
S	Set of sessions
s	Each session $s \in S$
$t(c)$	Time required to test core c
$t(s)$	Time required to test session s
$w(s)$	Power dissipated by session s
T	Total time required to test IC
$Cost(T, H)$	Test cost of IC defined in terms of T and H

the used notations are collected in Table 4.1. The values in Table 4.3 are used for the explanation of the terms, and the architecture in Figure 4.1 is used for illustration. The IC in Figure 4.1 comprises of a set of $C = \{Core1, Core2, Core3\}$ cores, each denoted by c, where $c \in C$. Each core c has a scan chain of length $l(c)$, requires $p(c)$ test patterns, and dissipates $w(c)$ power. Each column in Table 4.3, represents a core of the 3D Stacked IC with the corresponding scan chain length $l(c)$, patterns required $p(c)$ and power dissipated $w(c)$. For example, Core1 has a scan chain of length $l(Core1) = 30$, requires $p(Core1) = 30$ patterns, and dissipates $w(Core1) = 50$ units of power during testing.

Here we formulate test cost as a function of test time and DfT hardware for core-based non-stacked ICs with a IEEE 1149.1 test architecture.

The test time for a core c is given as $t(c)$:

$$t(c) = (\delta + l(c)) \cdot p(c) + l(c) \tag{4.1}$$

where, δ accounts for the number of clock cycles required for apply and capture, which is typically equal to 5 in the case of JTAG.

The time taken to test Core1 is calculated as:

$$t(Core1) = (\delta + l(Core1)) \cdot p(Core1) + l(Core1)$$
$$= (5 + 30) \cdot 30 + 30 = 1080 \; time \; units$$

For the IEEE 1149.1 test architecture, there are TDRs to enable test. In Figure 4.1, there are two TDRs which we denote with a set H. Hence, $H = \{TDR1, TDR2\}$, each denoted as h, where $h \in H$. Several cores may share a single TDR. For instance, Core1 and Core2 share TDR1. The number of TDRs is directly related to the number of sessions. All cores in each TDR are tested in the same session.

The test schedule for the set of C cores consists of a set of S sessions. A set of cores are tested in each session s, where $s \in S$. As illustrated in Figure 4.2, the set of sessions is denoted as $S = ((Core1, Core2), (Core3))$. Every core belongs to a unique session, $c \in s$. For example, Core1 and Core2 are tested in the first session, which is denoted as $s = (Core1, Core2)$. The test time $t(s)$ for any session is calculated as:

$$t(s) = \left(\delta + \sum_{c \in s} l(c)\right) \cdot \max_{c \in s}\{p(c)\} + \sum_{c \in s} l(c) \tag{4.2}$$

The time taken by session $s = (Core1, Core2)$ is calculated as:

$$t(Core1,Core2)$$
$$= (5 + l(Core1) + l(Core2)) \cdot$$
$$\max\{p(Core1), p(Core2)\}$$
$$+ \{l(Core1) + l(Core2)\}$$
$$= (5 + 30 + 30) \cdot max(30, 30)$$
$$+ (30 + 30) = 2010 \; time \; units$$

The power dissipated while testing a session s is given by $w(s)$, which is the sum of the power dissipated by each individual core tested in the session:

$$w(s) = \sum_{c \in s} w(c) \qquad (4.3)$$

The power dissipated by session $s = (Core1, Core2)$ is calculated as:

$$w(Core1,Core2)$$
$$= w(Core1) + w(Core2)$$
$$= 50 + 40 = 90 \; units$$

The overall test time for a test schedule T is calculated as follows:

$$T = \sum_{\forall s \in S} t(s) \qquad (4.4)$$

Which, in case of the given IC is:

$$T = t(Core1, Core2) + t(Core3)$$
$$= 2010 + 5320 = 7330 \; time \; units$$

The DfT hardware cost is directly related to the number of sessions, since each session corresponds to one TDR; hence, $|H| = |S|$, which is $|H| = 2$ as seen in Figure 4.1.

The test cost for any given configuration of the non-stacked IC is calculated as follows:

$$Cost(T, H) = \alpha \cdot T + \beta \cdot |H|$$
$$s.t. \ w(s) \leq w_{max}, \ \forall s \qquad (4.5)$$

where,

the power dissipated by each session, $w(s)$, is within the power constraint w_{max}, and α and β are weighting constants set by the designer depending on the co-relation between test time and TDR of the particular system.

The cost in the given example is calculated as:

$$Cost(T, H) = 1 \cdot 7330 + 2000 \cdot 2 = 11330 \ units$$

where, $\alpha = 1$ and $\beta = 2000$.

The problem is to find a test schedule such that the total test time and the number of TDRs required result in a reduced cost while meeting the power constraint.

4.2.2. 3D Stacked IC

Here we formulate a mathematical framework for test planning of core-based 3D Stacked ICs assuming a test architecture using the IEEE 1149.1 standard. We start by introducing the notations used through a system architecture, followed by the test mechanism, and finally a cost model.

For a 3D Stacked IC design having a stack of multiple chips, where each chip is supported by the IEEE 1149.1 test architecture, the used notations are collected in Table 4.2. Figure 4.3, along with the values provided in Table 4.3, is used for illustration. The 3D Stacked IC, which comprises of a set of $N = \{Chip1, Chip2\}$ chips in the stack. Each chip is denoted as n, $n \in N$, and has a set of $C(n)$ cores, each denoted by c, where $c \in C(n)$. For example, Chip1 comprises of three cores $C(Chip1) = \{Core1, Core2, Core3\}$. A core c has a scan chain of length $l(c)$, requires $p(c)$ test patterns, and the power dissipation during test is $w(c)$.

For the IEEE 1149.1 test architecture, there is a set of TDRs $H(Chip1) = \{TDR1, TDR2\}$, each denoted by h, where $h \in H(n)$.

The test schedule in chip n comprises $S(n)$ sessions, each denoted by s, where $s \in S(n)$. For example, Chip1 in Figure 4.3 comprises of a set of two sessions $S(Chip1) = \{(Core1, Core2), (Core3)\}$.

The test time $t(s)$ of a session s is given as in Eq. 4.2:

$$t(s) = \left(\delta + \sum_{c \in s} l(c) \right) \cdot \max_{c \in s}\{p(c)\} + \sum_{c \in s} l(c) \qquad (4.6)$$

The power dissipated while testing a session s, is given by $w(s)$, is

Table 4.2.: List of notations for 3D Stacked IC

Given	
N	Set of chips in the 3D Stacked IC
n	Each chip $n \in N$
$C(n)$	Set of cores in the chip n
c	Each core $c \in C(n)$
$l(c)$	Length of scan chain of core c
$p(c)$	Patterns required by core c
$w(c)$	Power dissipated to test core c
w_{max}	Power constraint
α	Designer specified constant to estimate the cost of time
β	Designer specified constant to estimate the cost of DfT hardware
Calculated	
$H(n)$	Set of TDRs in the chip n
h	Each TDR $h \in H(n)$
$S(n)$	Set of sessions in the chip n
s	Each session $s \in S(n)$
$t(c)$	Time required to test core c
$t(s)$	Time required to test session s
$w(s)$	Power dissipated by session s
$t(n)$	Wafer sort time of chip n
T_{ws}	Total wafer sort time required
S_{pt}	Set of sessions during package test
s_{pt}	Each session during package test $s_{pt} \in S_{pt}$
$t(s_{pt})$	Time required to test session s_{pt}
$w(s_{pt})$	Power dissipated by session s_{pt}
T_{pt}	Package test time
H	TDRs required to test 3D Stacked IC
T	Total time required to test 3D Stacked IC
$Cost_{SIC}(T, H)$	Test cost of 3D Stacked IC defined in terms of T and H

given as in Eq. 4.3:

$$w(s) = \sum_{c \in s} w(c) \qquad (4.7)$$

The time taken by each chip n during wafer sort is $t(n)$, which is calculated similar to the total time in case of non-stacked ICs in Eq. 4.4:

$$t(n) = \sum_{\forall s \in S(n)} t(s) \qquad (4.8)$$

Thus, the total time taken for wafer sort of the 3D Stacked IC, T_{ws}, is given as:

$$T_{ws} = \sum_{\forall n \in N} t(n) \qquad (4.9)$$

In the given example, the total wafer sort time is calculated as the sum of the wafer sort time for Chip1 $t(Chip1) = 7330$ and that of Chip2 $t(Chip1) = 7450$:

$$T_{ws} = t(Chip1) + t(Chip2)$$
$$= 7330 + 7450 = 14780 \; time \; units$$

For package test of the 3D Stacked IC, a test schedule is formed with S_{pt} sessions. Each core c belongs to a unique session s_{pt}, where $s_{pt} \in S_{pt}$. The test time $t(s_{pt})$ is represented in a similar manner as in case of wafer sort Eq. 4.6:

$$t(s_{pt}) = \left(\delta + \sum_{c \in s_{pt}} l(c) \right) \cdot \max_{c \in s_{pt}} \{p(c)\} + \sum_{c \in s_{pt}} l(c) \qquad (4.10)$$

The overall test time for package test of the 3D Stacked IC, T_{pt}, is given as the sum of the time taken by all sessions during package test, similar to Eq. 4.8:

$$T_{pt} = \sum_{\forall s_{pt} \in S_{pt}} t(s_{pt}) \qquad (4.11)$$

The power dissipated by each session s_{pt} of the package test, $w(s_{pt})$, is the sum of the power dissipated by each core belonging to all chips which are tested during the session, similar to Eq. 4.7:

$$w(s_{pt}) = \sum_{c \in s_{pt}} w(c) \tag{4.12}$$

The total time taken to test the 3D Stacked IC, T, is calculated as:

$$T = T_{ws} + T_{pt} \tag{4.13}$$

Assuming the package test schedule as
$S_{pt} = ((Core1, Core2), (Core3, Core4, Core5))$,
we get $T_{pt} = 14430$. Therefore, we can calculate T for the given 3D Stacked IC as:

$$T = T_{ws} + T_{pt} = 14780 + 14430 = 29210 \; time \; units$$

The DfT hardware, H, is given by the total number of TDRs, which is equal to the sum of the number of sessions during wafer sort of each chip.

$$H = \sum_{\forall n \in N} |H(n)| = \sum_{\forall n \in N} |S(n)| \tag{4.14}$$

For the given 3D Stacked IC, Chip1 and Chip2 require one TDR each during wafer sort. Thus, H is calculated as:

$$H = |H(Chip1)| + |H(Chip2)| = 2 + 1 = 3$$

The overall test cost for any given configuration of the 3D Stacked IC is similar to Eq. 4.5:

$$Cost_{SIC}(T, H) = \alpha \cdot T + \beta \cdot H$$
$$s.t. \; w(s_{pt}) \leq w_{max}, \; \forall s_{pt} \tag{4.15}$$

The power dissipated by any session during package test, $w(s_{pt})$, is within the power constraint w_{max}, and

α and β are weighting constants set by the designer depending on the co-relation between test time and TDR of the particular system.

The problem is to find the wafer sort schedule for each chip in the stack, and the package test schedule; such that the overall test time and the total number of TDRs required by all the N chips during wafer sort result in a reduced cost while meeting the power constraint.

4.3. Motivating Example

In this section we motivate the need of test planning of 3D Stacked ICs, while meeting a power constraint, by demonstrating the trade-off between test time and DfT hardware.

Figure 4.3 illustrates the 3D Stacked IC, and the corresponding values are provided in Table 4.3. The 3D Stacked IC comprises of two chips, $N = \{Chip1, Chip2\}$. The five cores are distributed among the two chips as: $C(Chip1) = \{Core1, Core2, Core3\}$ and $C(Chip2) = \{Core4, Core5\}$. Each column in Table 4.3, represents a core of the 3D Stacked IC with the corresponding scan chain length $l(c)$, patterns required $p(c)$ and power dissipated $w(c)$ respectively. It is assumed that the maximum power constraint $w_{max} = 100\ units$. The constants α and β are 1 and 2000 respectively.

A number of test plans are generated. The test plans and cost are presented in Table 4.4 and Table 4.5, respectively. Each test plan is analyzed individually in the remainder of this section. Table 4.4 is arranged as follows. It lists five test plans generated with the given 3D Stacked IC, while Table 4.5 shows the corresponding test costs. Five possible test plans have been listed, which generate the maximum and minimum costs with the given cores. In the first group of four columns, entitled wafer sort, we evaluate the test time and the number

Table 4.3.: Given values for the 3D Stacked IC

	Chip 1			Chip 2	
	Core1	Core2	Core3	Core4	Core5
Scan chain length $l(c)$	30	30	70	70	30
Patterns required $p(c)$	30	30	70	70	10
Power dissipated $w(c)$	50	40	40	20	10
Maximum power constraint w_{max}				100	
Time coefficient α				1	
Hardware coefficient β				2000	

Table 4.4.: Test schedule alternatives

Test	Wafer Sort		Package Test
Plan	Chip 1	Chip 2	
	$S(Chip1)$	$S(Chip2)$	S_{pt}
1	(Core1, Core2, Core3)	(Core4, Core5)	(Core1, Core2, Core3), (Core4, Core5)
2	(Core1, Core2, Core3)	(Core4), (Core5)	(Core1, Core2, Core3), (Core4), (Core5)
3	(Core1, Core2), (Core3)	(Core4, Core5)	(Core1, Core2), (Core3, Core4, Core5)
4	(Core1, Core2), (Core3)	(Core4), (Core5)	(Core1, Core2, Core5), (Core3), (Core4)
5	(Core1, Core2), (Core3)	(Core4), (Core5)	(Core1, Core2, Core5), (Core3, Core4)

Table 4.5.: Test costs achieved

Test	Wafer Sort				Package	Cost			Power
Plan	Chip 1		Chip 2		Test				Constraint
	Time	TDR	Time	TDR	Time	Time	Hardware	Overall	
	$t(Chip1)$	$\|H(Chip1)\|$	$t(Chip2)$	$\|H(Chip2)\|$	T_{pt}	T	H	$Cost_{SIC}(T,H)$	$w_{pt} \le w_{max}$
1	9580	1	7450	1	17030	34060	4000	38060	$130 \nleq 100$
2	9580	1	5700	2	15280	30560	6000	36560	$130 \nleq 100$
3	7330	2	7450	1	14430	29210	6000	35210	$90 \le 100$
4	7330	2	5700	2	13580	26610	8000	34610	$100 \le 100$
5	7330	2	5700	2	13230	26260	8000	34260	$100 \le 100$

of TDRs required by each chip forming the 3D Stacked IC. The corresponding session configuration is listed in Table 4.4. As an example, in the first row, corresponding to the columns under Chip1, for both Table 4.4, Core1, Core2 and Core3 are included within one session in Chip1, depicted by (Core1, Core2, Core3) during wafer sort, forming session $S(Chip1)$. The cost for each test plan in Table 4.4 is detailed in Table 4.5. Table 4.5 is arranged as follows. The first set of columns lists the test time and the number of TDRs required for wafer sort. The following column lists the time taken for package test. It should be noted here that the TDRs listed during wafer sort are reused for package test. The total cost is calculated in the following column, and finally the power constraint is in the last column. For example, the test time for wafer sort of Chip1 for the first test plan is calculated to be $t(Chip1) = 9580$ *time units*.

For the first test plan, all three cores in Chip1 share a common TDR, $|H(Chip1)| = 1$. Hence, the wafer sort schedule of Chip1 comprises of a single session, $S(Chip1) = (Core1, Core2, Core3)$, and the corresponding test time is calculated to be $t(Chip1) = 9580$ *time units* as in Eq. 4.8.

$$t(Chip1) = (l_{Core1} + l_{Core2} + l_{Core3} + 5) \cdot max(p_{Core1}, p_{Core2}, p_{Core3})$$
$$+ (l_{Core1} + l_{Core2} + l_{Core3})$$
$$= (30 + 30 + 70 + 5) \cdot max(30, 30, 70)$$
$$+ (30 + 30 + 70) = 9580 \ time \ units \tag{4.16}$$

Similarly, for Chip2, Core4 and Core5 share the same TDR, $|H(Chip2)| = 1$ for the third test plan. Hence, the wafer sort schedule of Chip2 is also comprised of single session, $S(Chip2) = (Core4, Core5)$, and the corresponding test time is $t(Chip2) = 7450 \ time \ units$. The package test schedule consists of two sessions

$S_{pt} = ((Core1, Core2, Core3), (Core4, Core5))$,

which requires a test time of $T_{pt} = 17030 \ time \ units$. Thus, the total time taken to test the 3D Stacked IC with the first test plan in Table 4.4 is calculated from Eq. 4.13 as the sum of the time taken for the wafer sort of each chip and the package test time to be $T = T_{ws} + T_{pt} = 34060 \ time \ units$.

The cores in either of the chips share a single TDR, hence from Eq. 4.14, we get $H = |H(Chip1)| + |H(Chip2)| = 2$. Therefore, the cost incurred $Cost_{SIC}(T, H) = 38060 \ units$ by the test plan proposed in case 1, is calculated as in Eq. 4.15.

However, the test plan is not valid. According to Eq. 4.12 the power dissipated for session $s_{pt} = (Core1, Core2, Core3)$ is

$w(Core1, Core2, Core3) = 130 \nleq 100 \ units$.

Thereby violating the power constraint. Hence, the first test plan is not valid.

For the second test plan, it can be seen that the overall test cost is reduced, despite an increase in the cost related to the DfT hardware. This is because the cost related to the test time more than compensates for it. Although the overall test cost for the second test plan is lower than that of the first, the test plan is infeasible due to the violation of the power constraint, similar to the first test plan.

For the third test plan in Table 4.4, during wafer sort of Chip1, (Core1, Core2) + (Core3), *i.e.*, Core1 and Core2 share a common TDR, while Core3 forms a separate session with a dedicated TDR. The test time of $t(Chip1) = 7330 \ time \ units$. For Chip2 session $S(Chip2) = (Core4, Core5)$ which means Core4 and Core5 share a TDR. The wafer sort time of $t(Chip2) = 7450 \ time \ units$. The hardware cost is one TDR. All power constraints are met during the wafer sorts. The cost incurred during package test is $Cost_{pt} = 14430 \ units$. Core1 and Core2 form a

package test session, while Core3, Core4 and Core5 form a different session, thus giving $S_{pt} = ((Core1, Core2), (Core3, Core4, Core5))$, and takes a test time of $T_{pt} = 35210$ *time units*. Each session meets the power constraint of $W_{max} = 100$ *units*. This results in a total test cost of $Cost_{case3} = 35210$ *units*. The third test plan has a lower test cost than the first test plan while meeting the power constraint.

For the fourth test plan, the wafer sort test time for Chip1 is $t(Chip1) = 7330$ *time units*, when two TDRs are used. Core1 and Core2 form a single session and Core3 forms another session, *i.e.*, $S(Chip1) = ((Core1, Core2), (Core3))$. For Chip2, where Core4 and Core5 are tested in separate sessions, represented as $S(Chip1) = ((Core4), (Core5))$, the time taken is $t(Chip2) = 5700$ *time units*. During package test, where Core1, Core2 and Core5 form a session, while Core3 and Core4 forms individual sessions $S_{pt} = ((Core1, Core2, Core5), (Core3), (Core4))$, while each session lies within the power constraint. The total test cost sums up to $Cost_{case4} = 34610$ *units*, which is less than $Cost_{case3}$. (Thus, it can be seen that although the cost incurred at wafer sort for the fourth test plan is more than that of the third test plan, the overall test cost for the fourth test plan is less than that of the third test plan.)

In the fifth test plan, where the wafer sort configurations for both Chip1 and Chip2 are same as in case 4, but during package test Core1, Core2 and Core5 are tested simultaneously during a package test session, while Core3 and Core4 are tested together in another session, giving $S_{pt} = ((Core1, Core2, Core5), (Core3, Core4))$, each meeting the power constraint, when the cost incurred is $Cost_{pt} = 13230$ *units*. The total test cost is $Cost_{case4} = 34260$ *units*, which is even less than $Cost_{case4}$. Hence, by properly forming sessions during package test, the test cost can be further reduced.

From the above studies, by comparing the wafer sort schedules of either Chip1 or that of Chip2 in the first test plan against the fifth test plan, it can be seen that the test time can be reduced by increasing the number of sessions and thereby increasing the number of TDRs. Although, an increased number of sessions implies increased DfT hardware cost. Hence, it is important to find a proper trade-off between the DfT hardware cost and the test time, meeting a power constraint, to obtain a test plan with the minimum test cost.

4.4. Proposed Approach

In this section we propose heuristics to reduce the test cost for non-stacked ICs and 3D Stacked ICs. The test cost is given as a weighted

sum of the test time and DfT hardware. The test time and the DfT hardware are co-optimized, while meeting the power constraint.

4.4.1. Non-stacked IC

For non-stacked ICs, the same test schedule is applied both during wafer sort and package test. Therefore, minimizing the test cost of either the wafer sort or package test, minimizes the overall test cost.

The basic principle of the algorithm to reduce the test cost is as follows:

The first core c is considered at a time from the given set of cores C. The core is assigned to a new TDR and the corresponding cost is calculated as $Cost(T, H)$. Thereafter, the core is included in each existing session s at a time. The test cost $\acute{C}ost(T, H)$, as a result of increased test time is calculated, provided that the power constraint is met. Each time a lower test cost is achieved, the test schedule is updated.

For example, considering Chip1 Table 4.3, illustrated by Figure 4.1, we have a set of three cores $C = \{Core1, Core2, Core3\}$. On sorting the cores in descending order of the number of patterns required $p(c)$, the elements of C can be rearranged as $C = \{Core3, Core1, Core2\}$. During the first iteration, we consider Core3, which forms the first session $S = (Core3)$. The corresponding test cost is calculated as $Cost(T, H) = 12640$. For the second iteration, Core1 is considered, which gives a test cost of $Cost(T, H) = 16800$ in a new session, while $\acute{C}ost = 16900$ along with Core3 in the first session, while meeting the power constraint. Since, the cost in the new session is lower, the test schedule is updated to $S = ((Core3), (Core1))$. Similarly, Core2 is considered during the third and last iteration, which gives a test cost $Cost(T, H) = 20960$ in a new session. On the same session as Core3, the test cost obtained is $\acute{C}ost(T, H) = 21060$, while meeting the power constraint, which is higher than the test cost obtained by including Core2 in the second session with Core1, calculated as $\acute{C}ost(T, H) = 18660$, while meeting the power constraint. In addition since $\acute{C}ost(T, H) < Cost(T, H)$, the cost of including Core2 in the same session as Core1 gives the lowest test cost. Thus, the test schedule for Chip1 can be given as $S = ((Core3), (Core1, Core2))$.

4.4.2. 3D Stacked IC

A test planning heuristic to reduce the test cost of 3D Stacked ICs is presented in Algorithm 3. Since, wafer sort of individual chips is

---

Algorithm 3 Test plan for 3D Stacked ICs with N chips in the stack

---

1: **Given:** $w_{max}, N, C(n), l(c), p(c), w(c), \alpha, \beta$
2: **Initialize:** $Cost_{SIC}(T, H) = 0$
3: **Sort** $C(n)$ in descending order of $p(c)$
4: **for all** $c \in C$ **do**
5:     **Calculate:** Cost in new wafer sort session
6:     **Calculate:** Cost in new package test session
7:     Cost in existing package test session:
8:     **for** Each existing session s_{pt} **do**
9:       **if** Power constraint is met, *i.e.,* $w(s_{pt}) + w(c) \leq w_{max}$ **then**
10:         **Calculate:** $\acute{C}ost(T, H)_{SIC}$
11:         **if** $\acute{C}ost(T, H)_{SIC} \leq Cost(T, H)_{SIC}$ **then**
12:           $Cost(T, H)_{SIC} = \acute{C}ost(T, H)_{SIC}$
13:           Update: $S_{pt} \cup c$
14:         **end if**
15:       **end if**
16:     **end for**
17:     Cost in existing wafer sort session:
18:     **for** Each existing session s **do**
19:       **if** Power constraint is met, *i.e.,* $w(s) + w(c) \leq w_{max}$ **then**
20:         **Calculate:** $\acute{C}ost(T, H)_{SIC}$
21:         Cost in existing package test session:
22:         **for** Each existing session s_{pt} **do**
23:           **if** Power constraint is met, *i.e.,* $w(s_{pt}) + w(s) \leq w_{max}$ **then**
24:             **Calculate:** $\acute{C}ost(T, H)_{SIC}$
25:             **if** $\acute{C}ost(T, H)_{SIC} \leq Cost(T, H)_{SIC}$ **then**
26:               $Cost(T, H)_{SIC} = \acute{C}ost(T, H)_{SIC}$
27:               Update: $S_{pt} \cup c$
28:             **end if**
29:           **end if**
30:         **end for**
31:       **end if**
32:     **end for**
33:     $\Delta\acute{C}ost = \Delta Cost_{ws} + \Delta Cost_{pt}$
34:     **if** $\Delta\acute{C}ost \leq \Delta Cost$ **then**
35:       $\Delta Cost = \Delta\acute{C}ost$
36:       Update: $S \cup c$
      $S_{pt} \cup c$
      $C(n) \cap c$
37:     **end if**
38: **end for**
39: **Output** Test plan: $Cost_{SIC}(T, H), S(n), S_{pt}$

---

followed by the package test of all chips together, in case of 3D Stacked ICs, optimizing the test plan of each chip individually may lead to a suboptimal test plan for the 3D Stacked IC. Hence, this heuristic takes into account both wafer sort and package test simultaneously, to arrive at a test plan.

The principle behind the proposed test planning heuristic is:

One core c is considered at a time from the given set of cores $C(n)$. The core is first assigned to a new TDR, and the corresponding wafer sort cost is calculated as $\Delta Cost_{ws}$. The additional package test cost is the minimum $\Delta Cost_{pt}$ obtained, by assigning the core to all possible sessions, one at a time, including a session where no other cores are tested. The total cost incurred by including core c in the test plan $\Delta Cost$ is the sum of the increased cost in wafer sort and package test. Thereafter, the core is included in each existing wafer sort session s at a time. The increase in the wafer sort cost $\Delta Cost_{ws}$ is calculated, provided that the power constraint is met. The corresponding increase in the package test cost is the minimum $\Delta Cost_{pt}$ obtained, by including the corresponding wafer sort session to all possible package test sessions. The overall increase in the test cost is calculated as $\Delta \acute{C}ost$. The test schedule is updated with the test schedule that provides the lesser test cost among $\Delta Cost$ and $\Delta \acute{C}ost$.

For example, considering Table 4.3, illustrated by Figure 4.3, we have a set of five cores

$C(Chip1, Chip2) = \{Core1, Core2, Core3, Core4, Core5\}$,

in the 3D Stacked IC. On sorting the cores in descending order of the number of patterns required $p(c)$, the elements of $C(n)$ are rearranged as $C = \{Core3, Core4, Core1, Core2, Core5\}$. During the first iteration, we consider Core3, which forms the first session $S_{pt} = (Core3)$. The corresponding test cost is calculated as $Cost(T, H) = 12640$. For the second iteration, Core4 is considered, which gives a test cost of $Cost(T, H) = 25280$ in a new wafer sort as well as package test session, while $\acute{C}ost = 24930$ along with Core3 in the first session, while meeting the power constraint. Since, the cost in the is lower in the existing session, the test schedule is updated to $S_{pt} = (Core3, Core4)$. Next, Core1 is considered for the third iteration, which violates the power constraint when $S_{pt} = (Core3, Core4, Core1)$, with $w(s_{pt}) = 110 > w_{max}$. Hence, a new session is inevitable, with the test schedule as $S_{pt} = ((Core3, Core4), (Core1))$ and $Cost(T, H) = 29090$. Similarly, Core2 is considered during the fourth iteration, which gives a minimum test cost $Cost(T, H) = 30950$, with a package test schedule $S_{pt} = ((Core3, Core4), (Core1, Core2))$. Eventually, at the fifth itera-

tion, Core5 gives a minimum test cost $Cost(T, H) = 33710$, with a package test schedule $S_{pt} = ((Core3, Core4), (Core1, Core2), (Core5))$. The wafer sort schedules can be obtained as as $S(Chip1) = ((Core3), (Core1, Core2))$, and $S(Chip2) = ((Core4), (Core5))$.

4.5. Experiments

To demonstrate the benefits of the test planning approach proposed in Section 4.4 on non-stacked ICs and 3D Stacked ICs, the test planning algorithm was implemented on several benchmark designs to reduce the test cost. A near optimal test cost for the benchmark designs is achieved by Simulated Annealing, and the two results are compared to demonstrate the performance of the proposed heuristic.

The experiments have been performed on the following six ITC'02 benchmark SoC designs:

p22810, p93791, g1023, d695, h953 and d281.

The following assumptions were made while considering the non-stacked ITC'02 SoC benchmarks as 3D Stacked ICs:

- The modules in the benchmark SoC designs are projected as cores in a non-stacked IC,

- All scan elements (inputs, outputs, and scan cells) in a core are connected to a single scan-chain,

- Modules without any scan chains are not considered,

- 3D Stacked ICs are constructed by stacking any number of the benchmark designs,

- The constant α for all designs is considered to be one, and

- Several experiments were performed with varying values of β for selected benchmark designs, and the most suitable value was found by dividing the test time of the core with the that requires the maximum test time, $t(c)$, by the number of cores, $|C(n)|$.

$$\beta = \frac{t(c)_{max}}{|C(n)|} \tag{4.17}$$

The reduced test cost obtained by the proposed approach is compared against the test cost obtained by Simulated Annealing.

4.5.1. Simulated Annealing

In this section we elaborate the Simulated Annealing approach used in this chapter. Simulated Annealing models the physical process of heating a material and then slowly lowering the temperature to decrease defects, thus minimizing the system energy. Simulated Annealing was used to reach a near optimal test cost in a manner stated below:

1. To begin with, the Simulated Annealing algorithm assumes a random test plan, where each core is allotted to a unique TDR. The cost is calculated by Eq.4.15, which serves as the first trial *point*.

   A starting temperature τ_0 is assumed to be the temperature of the initial state of the system. τ_0 in this case was chosen to be 10 *units*, which provided sufficiently many iterations until the stopping criteria in each case.

   The algorithm iterates in the following manner until the stopping criteria is reached.

2. During each iteration, a new *point* is generated, which is a previously unexplored test plan, in the following way: A core c is selected randomly.

   A random TDR, h, is selected, which is not the same as the TDR to which core c belongs.

   Core c is reallocated to TDR h, and the new cost is calculated by Eq.4.15 iff $w(s|c \in s) \leq w_{max}$.

3. The distance Δ of the new point from the previous point is the difference of the cost calculated at the new point generated and the previous trial point. In this chapter we have assumed temperature to be decreasing exponentially, hence allowing exploration of all possible configurations. The trial point distance distribution, *i.e.*, the extent of the search, equals the current temperature with a uniformly random direction.

4. The algorithm not only accepts all new points that lower the overall test cost, but also, according to the acceptance function, points that raise the overall test cost. By accepting points that raise the overall test cost, the algorithm avoids being trapped in local minima in early iterations and is able to explore globally for better

solutions. The acceptance parameter being

$$\frac{1}{1 + exp\left(\frac{\Delta}{\tau}\right)} \tag{4.18}$$

where,

Δ = new cost - old cost

τ = current temperature.

Since both Δ and τ are positive, the probability of acceptance is between 0 and 0.5. Smaller temperature or larger δ leads to smaller acceptance probability.

5. The algorithm systematically lowers the temperature after each iteration, storing the best point found so far. The temperature function specifies the function the algorithm uses to update the temperature. Let k denote the annealing parameter. The annealing parameter is the same as the iteration number until reannealing.

$$\tau = \frac{\tau_0}{log(k)} \tag{4.19}$$

6. The algorithm reanneals at certain intervals. Reannealing sets the annealing parameters to lower values than the iteration number, thus raising the temperature in each dimension. The annealing parameters depend on the values of estimated gradients of the cost function in each dimension. The basic formula is

$$k_i = log\left(\frac{\tau_0}{\tau_i} \cdot \frac{max_j(g_j)}{g_i}\right) \tag{4.20}$$

where,

k_i = annealing parameter for iteration i.

τ_0 = initial temperature of iteration i.

τ_i = current temperature of iteration i.

g_i = gradient of objective in direction i times difference of bounds in direction i.

Simulated Annealing safeguards the annealing parameter values against improper values.

Table 4.6.: Reduction in cost for non-stacked ICs

No.	Design	Heuristic			Simulated Annealing			% Cost
		TDR	Time	Cost	TDR	Time	Cost	Difference
1	p22810	12	485857	497425	13	479130	491662	1.15
2	p93791	6	598922	605096	9	587300	596561	1.14
3	g1023	5	44950	46270	5	44950	46270	0.00
4	d695	4	37453	38533	5	35063	36413	5.50
5	h953	5	236220	253465	5	236220	253465	0.00
6	d281	3	111210	120306	3	111210	120306	0.00
							Average:	1.34

For this system, reannealing intervals were chosen, as a product of the number of cores in the 3D Stacked IC and the total number of TDRs, which ensured that each core test would traverse all plausible sessions.

7. The algorithm stops when the average change in the objective function is small relative to the function tolerance, or when it reaches any other stopping criterion.

Thus, in this case, when the change in temperature was higher than the change in the test cost, the process was terminated. It refers to the temperature below which no more cores can be allocated to different TDRs.

In case of non-stacked ICs, the reduced test cost under a power constraint is presented in Table 4.6. In the table, each row corresponds to a SoC benchmark design, which is shown in the second column. The number of cores in each design is shown in column three. The first group of three columns, entitled heuristics, show the reduced test cost of the respective designs as obtained by the algorithm proposed in Section 4.4. The later group of columns depict the reduced cost as obtained by Simulated Annealing of the benchmarks. In either group of three columns, the first column shows the number of TDRs required by the obtained test schedule, followed by the test time for the same schedule. The third column is the cost obtained by applying Eq.4.15. The final column shows the percentage difference in the cost obtained by the proposed heuristic to that of the cost obtained by Simulated Annealing. It can be observed that, at an average the proposed approach arrives at a cost which is 1.3% greater than the cost of the test plan obtained by Simulated Annealing.

Table 4.7.: Reduction in cost for 3D Stacked ICs
with 2 and 3 chips in the stack

Design	Heuristic			Simulated Annealing			% Cost
nos.	TDR	Time	Cost	TDR	Time	Cost	Difference
2-chip 3D Stacked ICs							
1,2	18	1087919	1103905	22	1066430	1085968	1.62
2,3	11	643872	655204	17	629525	647038	1.25
3,4	9	82403	84257	13	77288	79966	5.09
4,5	9	275037	286038	10	272647	284871	0.41
5,6	8	348794	368171	8	337646	357023	3.03
						Average:	2.28
3-chip 3D Stacked ICs							
2,3,4	15	681325	690710	22	664588	678352	1.79
3,4,5	14	319987	326132	18	314872	322773	1.03
4,5,6	12	386247	397834	13	372709	385262	3.16
						Average:	1.99

Table 4.8.: Comparison of total execution times
of the proposed heuristic against Simulated
Annealing

No. of chips	Heuristic T_1 (Seconds)	Simulated Annealing T_2 (Seconds)	Ratio T_2/T_1
1	3	376	$1.3 \cdot 10^2$
2	10	20000	$2.0 \cdot 10^3$
3	6	17000	$2.9 \cdot 10^3$

In Table 4.7, the package test cost for various 3D Stacked IC designs
made by stacking the six benchmark designs in Table 4.6 are shown.
The number of chips that have been stacked to make the 3D Stacked
IC is shown in the leftmost column. The group of five rows have 3D
Stacked ICs with two chips in the stack, followed by a group of three
rows having three chips in the stack. The second column from left
shows the benchmark designs that have been used to make the stack,
which correspond to the serial number used in Table 4.6. For instance,
the first 3D Stacked IC design contains two chips in the stack, 1 and 2,
which refers to p22810 and p93791 respectively. The third column lists
the total number of cores in the stack. The groups of three columns
on the left list the number of TDRs, the test time and the test cost re-

spectively for the proposed approach. Similarly, the second group of three columns list the number of TDRs, the test time and the test cost respectively for Simulated Annealing. The rightmost column enlists the percentage reduction in the test cost obtained by using Simulated Annealing. It can be seen that in case of 3D Stacked ICs with 2 chips, the average reduction in test cost obtained is 2.28% and that with 3 chips in the stack is 1.99%. Furthermore, experiments were performed with varying values of β for selected benchmark designs, and it was found that the test cost obtained by the proposed heuristic was higher than the test cost obtained by Simulated Annealing by a maximum of 6.90%. However, since the Simulated Annealing algorithm in this case is required to nearly exhaust the search space, the computation time required for the algorithm rises exponentially with increasing number of cores. On the other hand, the computation time required by the heuristic increases linearly with the number of cores in the 3D Stacked IC. Table 4.8 shows the rounded values of the time taken to execute the heuristic and Simulated Annealing on all designs listed in Table 4.6 and Table 4.7. It can be seen that Simulated Annealing arrives at the desired test plan with considerably longer computation time as compared to the heuristic. The heuristic arrives at test plan within a matter of seconds in case of non-stacked ICs, as well as for 3D Stacked ICs. In case of non-stacked ICs, Simulated Annealing arrives at a test plan for all six designs in just over 6 minutes. However, in case of 3D Stacked ICs with two and three chips in the stack, Simulated Annealing requires almost 6 hours and 5 hours respectively, to determine all test plans.

4.6. Discussion

In this chapter, we define test cost as a function of test time and test hardware for core based ICs provided with the IEEE 1149.1. We have considered non-stacked ICs and 3D Stacked ICs. A test planning algorithm was proposed for scheduling tests while meeting power constraints. The test planning algorithm which addresses the following:

1. For a non-stacked IC, where the same test schedule is applied during wafer sort and package tests, the tests of all the cores are grouped in sessions such that the cost is reduced by co-optimizing test time and the number of TDRs required.

2. For a 3D Stacked IC, where each chip is tested individually during wafer sort and jointly during package test. The cost is re-

duced by forming sessions from different chips concurrently during the package test.

The test planning algorithm was implemented on several ITC'02 SoC benchmarks, and the results were compared against the test cost obtained by Simulated Annealing of the benchmarks. It was observed that the proposed test planning algorithm arrived at a cost with considerably lower computational time, which was in the worst case, 6% (Table 4.6, Case4 and Table 4.7, 3D Stacked IC with designs 3 and 4) higher than the test cost obtained by Simulated Annealing, both in case of non-stacked ICs and 3D Stacked ICs.

5

IEEE 1500 Based Architecture

In this chapter, we address test planning for core-based 3D Stacked ICs supported by a IEEE 1500 based test architecture. The test cost corresponding to the test plan is given as the sum of the cost related to test time and DfT hardware. A mathematical model to calculate the test cost is presented. We propose a scheme for test scheduling and TAM design for 3D stacked ICs based on an ILP method. The ILP model is implemented on several designs constructed from ITC'02 benchmarks. The experimental results show significant reduction in test cost compared to schemes that are optimized for non-stacked ICs.

This chapter proceeds with presenting the assumed mechanism of the system in Section 5.1. A cost model for 3D Stacked ICs using a IEEE 1500 test architecture scheme is discussed in Section 5.2, followed by a motivating example in Section 5.3. A test planning method proposed in Section 5.4 and experiments are performed on several benchmarked circuits, and the results are presented and discussed in Section 5.5.
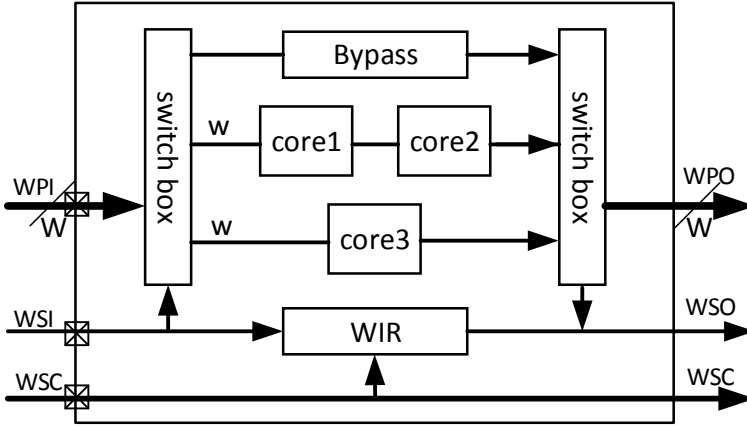
Figure 5.1.: IEEE 1500 based test architecture of a non-stacked IC

5.1. Test Process

In this section we first discuss the test architecture for a non-stacked IC with an IEEE 1500 based test infrastructure as described in [105], followed by a 3D Stacked IC, where each chip of the stack is supported by the IEEE 1500 infrastructure.

5.1.1. Non-stacked IC

A non-stacked IC supported with an IEEE 1500 based test infrastructure is illustrated in Figure 5.1. The chip consists of three cores, *viz.*, Core1, Core2 and Core3, which are accessed by the TAMs. The Wrapper Parallel Port (WPP), comprising the Wrapper Parallel Input (WPI) and the Wrapper Parallel Output (WPO) in the figure, depicts a set of TAMs W, to be decided by the user. The TAM is partitioned among two groups $W = (Gr_1, Gr_2)$. The chip has a TAM width of W during wafer sort, which is split among Gr_1 connecting Core1 and Core2 in series. Gr_1 concatenates the scan chains of Core1 and Core2 serially, while Gr_2 connects Core3 to the switch boxes. The switch boxes select, depending on the instruction in the WIR, the bypass register or all the TAM lines. The Wrapper Serial Port (WSP), comprising of Wrapper Serial Input (WSI), Wrapper Serial Output (WSO), and Wrapper Serial Control (WSC) terminals, as shown in Figure 5.1, supports the serial test mode. The instruction to be executed in a chip is stored in the corresponding Wrapper Instruction Register (WIR).

5.1.2. 3D Stacked IC

Figure 5.2 illustrates a 3D Stacked IC with two chips, where Chip2 is stacked on top of Chip1. Chip1 contains Core1, Core2 and Core3, while Chip2 hosts Core4 and Core5.

At wafer sort, Chip1 and Chip2 each use an IEEE 1500 based test infrastructure as described in Section 5.1.1. Chip1 would require a TAM width of $W(Chip1)$ during wafer sort, which is split among Gr_1 connecting Core1 and Core2 in series, and Gr_2 to Core3, while Chip2 requires an optimal TAM width $W(Chip2)$, which is split among Gr_3 and Gr_4 to Core4 and Core5, respectively.

During package test of the 3D Stacked IC in Figure 5.2 a TAM of width $|W|$ is provided to all the chips in the stack, which concatenates the cores of Chip1 and Chip2. The TAM width $|W|$ in this case can be determined by $\max(W(Chip1), W(Chip2))$, which is the maximum TAM width required by any chip forming the 3D Stacked IC.

In addition, the WPOup, WSOup and WSCup of Chip1, *i.e.*, the lower chip in the stack, is interconnected to the WPIdown, WSIdown and WSCdown of Chip2, respectively, *i.e.*, the chip on top. The WPOup, WSOup and WSCup of the topmost chip, Chip2, are directed out via the WPOdown, WSOdown and WSCdown, respectively, of the lowermost chip in the stack, Chip1. The WPOdown, WPIdown, WSOdown, WSOdown and WSCdown of the lowermost chip, Chip1, serve as the package test interface for the 3D Stacked IC. As illustrated in [52], in this chapter we assume equal number of WPIs and WPOs for each chip.

The TSV interconnect between chips may be tested using the boundary scan registers, which connects all input/output pins and TSVs. Boundary scan registers are implemented on both chips and are used in TSV interconnect test. Test stimuli are applied on out-going TSVs and test responses are captured on in-coming TSVs. Since the boundary scan register is a separate register, testing of TSVs cannot be performed concurrently with core tests.

The TSV interconnect tests contribute with a constant term to the overall test time and could not be scheduled with any core tests. During TSV interconnect testing, the BYPASS mode is selected for each chip in the stack. As illustrated in Figure 5.2, the WIR selects the bypass register for both Chip1 and Chip2 simultaneously. Therefore, the time required to perform TSV interconnect tests are overseen while addressing the total test time in the remainder of the chapter.
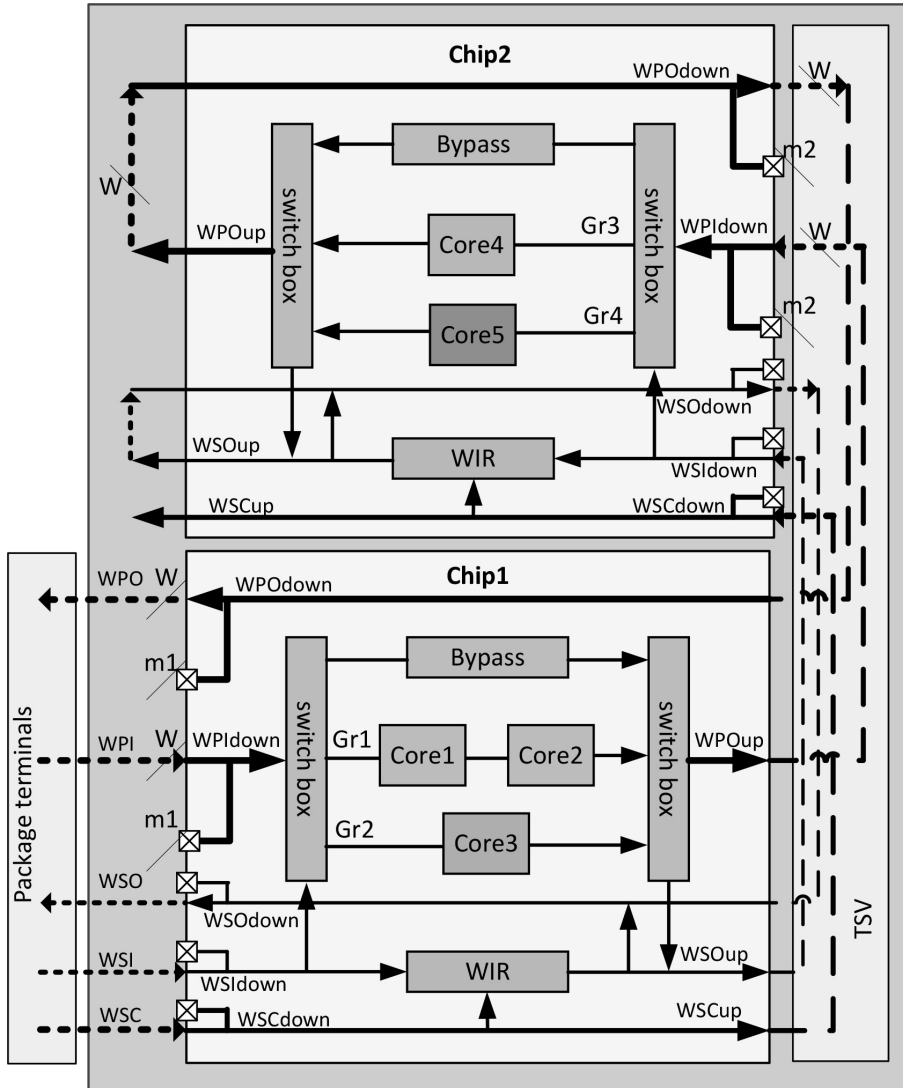
Figure 5.2.: IEEE 1500 based test architecture of a 3D Stacked IC
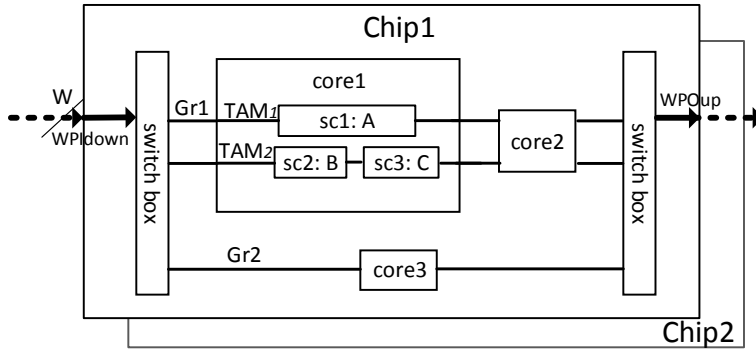
Figure 5.3.: Scan-chains configured into wrapper-chains in Core1 of Chip1

5.2. Problem Formulation

In this section we formulate the problem of test planning for core-based 3D Stacked ICs.

We first introduce the notations and their significance. The notations are listed in Table 5.1, and are grouped among the given variables or the ones to be calculated. The system architecture is illustrated using the corresponding notations in Figure 5.2.

5.2.1. 3D Stacked IC Configuration

A 3D Stacked IC with $N = \{Chip1, Chip2\}$ chips in the stack is shown Figure 5.2. Chip2 is stacked over Chip1. Each chip is denoted as i, $i \in N$, and has a set of $C(i)$ cores, each denoted by c, where $c \in C(i)$. For example, Chip1 comprises of three cores

$C(Chip1) = \{Core1, Core2, Core3\}$. A core c has a set of $S(c)$ scan chains, each denoted by sc scan chains of lengths $l(sc)$, requires $p(sc)$ test patterns, respectively. The scan chains are concatenated into several wrapper chains and accessed through a group of TAM lines.

For a 3D Stacked IC, where each chip is supported by the IEEE 1500 based test infrastructure, the assignment of scan chains to respective TAM lines (core-scan chain configuration) within core1 of Chip1 of Figure 5.2 is shown in Figure 5.3. The routing of the IEEE 1500 test architecture is overlooked in the figure. We assume the following regarding the 3D Stacked IC configuration:

A set of N chips in the stack, where each chip is denoted as i, $i \in N$. In Figure 5.3, the 3D Stacked IC consists of two chips, *viz.*, Chip1 and

Table 5.1.: List of notations

Given	
N	Set of chips in the 3D Stacked IC
i	Each chip $i \in N$
$C(i)$	Set of cores in the chip i
c	Each core $c \in C(i)$
$S(c)$	Set of scan chains in core c
sc	Each scan chain $sc \in S(c)$
$l(sc)$	Length of scan chain sc
$p(sc)$	Patterns required by core c
κ	Designer specified weighting constant
Calculated	
$t(c)$	Time required to test core c
$t(sc)$	Time required on scan chain sc
$t(b)$	Time required on TAM line sc
W	Set of TAM lines
w	Each partitioned group of TAM $w \in W$
$b(w)$	Each TAM line $b \in w$
K	Total number of TAM groups
$y(c,w)$	Binary decision variable
	1; if core c is assigned to partition of TAM w
	0; otherwise
$t(c,w)$	Test time required by core c when the scan-chains are
	concatenated in upto w wrapper-chains
$t(w)$	Total test time required by all cores assigned to TAM partition w
$t(i)$	Wafer sort time of chip i
T_{ws}	Total wafer sort time required for all chips
T_{pt}	Package test time for the complete 3D Stacked IC
T	Total time required to test the 3D Stacked IC
P	DfT hardware cost of the 3D Stacked IC
$Cost(T,P)$	Total test cost
	given as the weighted sum of test time and DfT hardware

Chip2. Chip i has $C(i)$ cores, each denoted as c, $c \in C(i)$. For example, Chip1 has three cores, denoted by Core1, Core2 and Core3 in Figure 5.3. Core c has a set of sc scan chains, and requires $p(sc)$ test patterns. The three scan chains of Core1 of Chip1 are shown in Figure 5.3, labeled as A, B and C, respectively, for ease of reference.

Each functional input or output is equivalent to an internal scan chain of length 1, and are included within sc

Scan chain sc is of length $l(sc)$.

Scan chain sc requires $t(sc)$ time units, and is computed as:

$$t(sc) = (1 + l(sc)) \cdot p(sc) + l(sc) \qquad (5.1)$$

For example, in Figure 5.3, if $l(B) = 2$ and $p(B) = 166$, we have

$$\begin{aligned} t(B) =& (1 + l(B)) \cdot p(B) + l(B) \\ =& (1 + 2) \cdot 166 + 2 = 500 \end{aligned}$$

$$(5.2)$$

For the IEEE 1500 test infrastructure to be optimized, for any given configuration we assume:

A set of W TAM lines as shown in Figure 5.3, where chip i receives a WPIdown of width $|W|$

Width of the TAM W, is divided into smaller sets of TAMs, each denoted by w.

As seen in Figure 5.3, Core1 is accessed by TAM group $w = Gr_1$ which has $|w| = 2$ TAM lines.

Core c is accessed by TAM group, w

Multiple cores may be accessed in series by a single group of TAM, w. As can be seen in Figure 5.3, where $w = Gr_1$ concatenates Core1 and Core2.

Each scan chain sc is accessed by a single TAM line, as visible in Figure 5.3, where $b(Gr_1) = TAM_1$ accesses scan chain A.

Multiple scan chains may be accessed in series by a single TAM line. It is shown by $b(Gr_1) = TAM_2$, which concatenates scan chains B and C in Figure 5.3.

Time required to execute tests of all scan chains on a single TAM line b within a core c is given by $t(b)$, the sum of the time taken by each individual scan chain $t(sc)$, such that $sc \in b$:

$$t(b) = \sum_{\forall sc \in b} t(sc) \qquad (5.3)$$

For example, the time on TAM_2 within core1 is the sum of the time taken by scan chains B and C:

$$t(TAM_2) = t(B) + t(C) = 500 + 500 = 1000 \qquad (5.4)$$

In the rest of the chapter, wherever the length of each scan chain $l(sc)$ and the number of patterns required by each core $p(sc)$ are provided, we have subsequently stated the respective test time obtained $t(sc)$ as given.

The test cost for any given configuration of the IEEE 1500 test infrastructure is obtained as below:

Time taken to test chip i in the stack, $i.\ e.$, the wafer sort time for any chip, $t(i)$, is given by the maximum time at any TAM line b:

$$t(i) = \max_{\forall b \in W} \sum_c t(b) \qquad (5.5)$$

For example, assuming each scan chain is assigned to a dedicated TAM line, the time taken by Chip1 is:

$$t(i) = \max 600, 500, 500, 200, 200, 1000 = 1000 \qquad (5.6)$$

In a similar manner, the package test time T_{pt}, of all the stacked, bonded and packaged chips tested simultaneously is bounded by the maximum time taken at any TAM line b, in the 3D Stacked IC:

$$T_{pt} = \max_{\forall b \in W, i \in N} \sum_c t(b) \qquad (5.7)$$

The overall test time of the 3D Stacked IC, T, is given by the sum of the wafer sort time of each chip in the stack $t(i)$, and the package test time T_{pt}, when all chips are tested jointly after stacking, bonding and packaging:

$$T = \sum_{\forall i \in N} t(i) + T_{pt} \qquad (5.8)$$

Besides the routing, each TAM line requires dedicated chip pins (ATE pins during wafer sort) for transfer of test stimuli. Therefore, the number of TAM lines $|W|$ corresponds directly to the hardware cost of each chip. Since, each chip in the stack has equal number of TAM lines, the total hardware cost P, is the product of the number of chips in the stack N and the width of the TAM $|W|$:

$$P = N \cdot |W| \qquad (5.9)$$

Table 5.2.: Given data for 3D Stacked IC

Chip	Core	Scan chain	Test Time	Label
i	c	sc	$t(sc)$	
	1	1	600	A
		2	500	B
		3	500	C
1	2	4	200	D
		5	200	E
	3	6	1000	F
	4	1	600	X
2		2	500	Y
	5	3	1000	Z
Time equivalent of each TAM, κ				200

Thus, the cost of testing a 3D Stacked IC can be defined as the cumulative cost of the total test time T, as obtained in Eq. 4.4, and the required test hardware P, from Eq. 5.9:

$$Cost(T, P) = T + \kappa \cdot P \qquad (5.10)$$

where κ is the weighting factor for the test hardware, *i.e.*, the time equivalent for each unit of hardware.

5.2.1.1. Objective

Given the 3D Stacked IC supported by the IEEE 1500 test architecture, a test plan is developed to reduce the overall test cost, defined by Eq. 5.10, which is achieved by optimizing the following parameters

1. width of the TAM, $|W|$

2. each partition of W, given by w

3. assigning each core c to a single partition of TAM w

4. assigning each scan chain sc to respective TAM lines b

5.3. Motivating Example

In this section an example is illustrated to motivate the significance of test planning by illustrating the trade-off between test time and test hardware while considering both wafer sort and package tests

Table 5.3.: Test cost variation for each chip

Chip	TAM Width	Test Time	Cost Achieved	Configuration
i	$\lvert W \rvert$	$t(i)$	$t(i)+$ $\kappa \cdot \lvert W \rvert$	
1	1	3000	3200	{A, B, C, D, E, F}
	2	1600	2000	{A, B, C}, {D, E, F}
	3	1400	2000	{A, B}, {C}, {D, E, F}
	4	1000	1800	{A}, {B, C}, {D, E}, {F}
	5	1000	2000	{A}, {B}, {C}, {D, E}, {F}
	6	1000	2200	{A}, {B}, {C}, {D}, {E}, {F}
2	1	2100	2300	{X, Y, Z}
	2	1100	1500	{X, Y}, {Z}
	3	1000	1600	{X}, {Y}, {Z}

simultaneously. The following example shows that an improper co-optimization of the test time and test hardware leads to a higher test cost for the 3D Stacked IC.

In this example, the 3D Stacked IC shown in Figure 5.2 has been considered. Table 5.2 shows the given parameters for the 3D Stacked IC. The first column from the left divides the table for either chip, *viz.*, Chip1 and Chip2, respectively. The second column lists the cores in the 3D Stacked IC, as seen in Figure 5.2. The third column lists the number of scan chains per chip, as well as scan chains per core. As can be seen, in Table 5.2, Core1 has three scan chains, $\{A, B, C\}$ which is also visible in Figure 5.3, Core2 has two: $\{D, E\}$, Core3 contains one scan chain: $\{F\}$, Core4 has two: $\{X, Y\}$, while Core5 has one: $\{Z\}$. In the next column the test time taken by each scan chain is tabulated. The test time is derived from a given length of the respective scan chain and the number of patterns required by the corresponding core, which have been concealed from the table. The rightmost column labels each scan chain in Chip1 as $A - F$ while for Chip2 $X - Z$, for ease of reference. The constant, κ of Eq. 5.10 is assumed to be 200 *units*, as the lowermost row of Table 5.2 depicts.

Table 5.3 depicts the variation in the test cost with increasing TAM width for the wafer sort of each chip. The first column of Table 5.3 divides the table for Chip1 and Chip2. The second column depicts all possible TAM widths for each chip. We can see that the maximum possible number of used TAM lines for Chip1 is 6 while that of Chip2 is 3, when each scan chain has a dedicated TAM line, depicted by the

last row of each chip in Table 5.3. On the other hand, the first row of each chip depicts the scenario where all scan chains of either chip are connected in series on a single TAM line. The third column shows the minimum test times achieved with the corresponding TAM widths. The fourth column lists the cumulative cost obtained from the given TAM width and the minimum test time hence obtained. It is seen that for Chip1 the cost is lowest at a TAM width of 4, for 1800 *units*, while for Chip2 it is 1500 *units* with a 2 TAM lines. Intuitively, the test cost is lowest at the best trade-off with a certain TAM width and the corresponding test time. Increasing the TAM width results in increasing the test cost, as the corresponding reduction in the test time is insufficient to balance the trade-off. The last column in Table 5.3 shows the assignment of scan chains to respective TAMs. For example, in case of Chip1 with two TAM lines, the lowest test time is obtained when scan chains A, B, and C are connected in series in the first TAM line, while scan chains D, E and F are assigned to the second, denoted by: $\{A, B, C\}, \{D, E, F\}$. Thus the wafer sort time in this case is given by:

$$
\begin{aligned}
T_{ws1} &= max\left\{(T_A + T_B + T_C), (T_D + T_E + T_F)\right\} \\
&= max\left\{(600 + 500 + 500), (200 + 200 + 1000)\right\} \\
&= max\left\{1600, 1400\right\} = 1600
\end{aligned}
$$

$$(5.11)$$

While the wafer sort cost for Chip1 can be calculated as:

$$
\begin{aligned}
Cost_{ws}(Chip1) &= t(Chip1) + \kappa \cdot |W| \\
&= 1600 + 200 \cdot 2 = 2000
\end{aligned}
$$

$$(5.12)$$

If the wafer sort schedule providing the lowest test cost of each chip, the TAM width $|W|$ required by the 3D Stacked IC is upper bounded by the maximal TAM width required by any chip in the stack to arrive at the lowest wafer test cost. Therefore, a number of redundant TAM lines must be routed through the chips forming the 3D Stacked IC that require a lower TAM width during wafer sort to arrive at the respective test cost. On the other hand, if the test time of each chip in the 3D Stacked IC is distributed over the whole TAM width $|W|$, the package test time can be minimized by evenly distributing the test time over the total TAM width $|W|$, resulting in a lower overall test

cost. For example, in Table 5.3 it can be seen that the the lowest cost obtained for Chip1 is with 4 TAM lines, whereas that of Chip2 is with 2 TAM lines. Therefore, during wafer sort, if either chip utilizes the TAM width that provides the minimal wafer sort cost, Chip1 would utilize 4 TAM lines, while Chip2 utilizes just 2 of the lines. However, the hardware cost would still be accounted for 4 TAM lines. On the other hand, during wafer sort the test time for Chip2 remains 1100 *time units*, which is longer than the time taken by utilizing all 3 TAM lines, *i.e.*, 1000 *time units*. Furthermore, with a TAM width of 2, the minimum package test time achieved by interconnecting respective TAM lines is 2600 *time units*, obtained as a result of concatenating the TAMs of Core1 and Core5. The lowest package test time is obtained by interconnecting the TAMs of Chip1 and Chip2 in the following manner: $\{A, B, C, Z\}, \{D, E, F, X, Y\}$.

To arrive at the lowest cost for the 3D Stacked IC described above, we have:

$$
\begin{aligned}
Cost_{\#case} &= T + \kappa \cdot P \\
&= \left(\sum_{\forall i \in N} t(i) + T_{pt} \right) + \kappa \cdot (N \cdot |W|) \\
&= \big(t(Chip1) + t(Chip2) + T_{pt} \big) + 200 \cdot (2 \cdot |W|) \\
&= \big(t(Chip1) + t(Chip2) + T_{pt} \big) + 400 \cdot |W| \qquad (5.13)
\end{aligned}
$$

Three possible scenarios are hereby presented, for which the test cost is determined by substituting the values in Eq. 5.13:

1. *TAM width* $|W| = 2$

$$
\begin{aligned}
Cost_i &= (1600 + 1100 + 2600) + 400 \cdot 2 \\
&= 1000 + 1100 + 2600 + 800 = 5500 \ units \qquad (5.14)
\end{aligned}
$$

2. *TAM width* $|W| = 3$

$$
\begin{aligned}
Cost_{ii} &= (1400 + 1000 + 2100) + 400 \cdot 3 \\
&= 1400 + 1000 + 2100 + 1200 = 5700 \ units \qquad (5.15)
\end{aligned}
$$

3. *TAM width* $|W| = 4$

$$
\begin{aligned}
Cost_{iii} &= (1000 + 1000 + 1600) + 400 \cdot 4 \\
&= 1000 + 1000 + 1600 + 1600 = 5200 \ units \qquad (5.16)
\end{aligned}
$$

It is seen that *Case iii*. gives the lowest test cost. It can be seen that, minimizing the wafer sort cost of each chip in the stack may not minimize the overall test cost. The optimized parameters mentioned in Section 5.2, for the given 3D Stacked IC are:

1. bit width of the TAM: $|W| = 4$

2. number of TAM partitions: $K = 3$

3. partition of the total TAM width $|W|$, among K groups of TAMs w, each given as: $|Gr_1| = 2$, $|Gr_2| = 1$ and $|Gr_3| = 1$

4. assigning each core to a single partition of the TAM w: Core1 to Gr_1, Core2 to Gr_2, Core3 to Gr_3, Core4 to Gr_1 and Core5 to Gr_2

5. designing a test wrapper for each core by assigning scan chains to respective TAM lines b: $\{A, X\}$ to TAM_{11}, $\{B, C, Y\}$ to TAM_{12} and $\{D, E, Z\}$ to TAM_{21} and $\{F\}$ to TAM_{31}

It is seen that to arrive at the minimum test cost for a 3D Stacked IC supported by the IEEE 1500 based test infrastructure, it may be insufficient to minimize either the wafer sort cost of each chip or the package test cost. Thus, the test cost is minimized by co-optimizing the test time and hardware cost, while considering both the wafer sort and package test instances simultaneously.

5.4. Proposed Approach Using ILP

In this section the test planning problem for 3D Stacked ICs is formulated as an ILP model, bounds are ascribed to the variables. An algorithm to implement the ILP model is presented.

In [40] [45], for non-stacked ICs, the problem of wrapper-TAM co-optimization was stated to be NP-hard and a solution was proposed using ILP. Since, 3D Stacked ICs are manufactured by stacking non-stacked ICs, the problem can be extended to 3D Stacked ICs to state that, minimizing the cost while considering both the test time cost and the optimized wrapper-TAM as the hardware cost is NP-hard as well. Hence, in the following, an ILP model is detailed.

ILP is applied to minimize a linear objective function on a set of integer variables, while satisfying a set of linear constraints. The ILP model, as explained in [24], can be generalized as :

Minimize: $A \cdot x$

subject to: $B \cdot x \leq C$,   s.t., $x \geq 0$

where *A* is the linear objective function, *B* defines the constraints, *C* is a set of constants, and *x* is a vector of integer variables.

To reach at the minimal cost, the objective function to be minimized is in Eq. 5.10. The function can be expanded to contain the optimizing factors in the following way:

$$
\begin{aligned}
Cost &= T + \kappa \cdot P \\
&= \left(\sum_{\forall i \in N} t(i) + T_{pt} \right) + \kappa \cdot (N \cdot |W|) \\
&= \sum_{\forall i \in N} \max_{\forall b \in W} \left(\sum_{\forall sc(c) \in b} t(sc) \right) \\
&\quad + \max_{\forall b \in W, i \in N} \left(\sum_{\forall sc(c) \in b} t(sc) \right) + \kappa \cdot N \cdot \sum_{\forall w \in W} |w|
\end{aligned}
$$

$$(5.17)$$

subject to the following constraints

- Each scan chain *sc* is assigned to exactly one TAM line *b*, *i.e.*, $sc(c) \cap b$ are unique

- Each core *c* is assigned to exactly one TAM group *w*

- Sum of the TAM width for each chip is *W*, *i.e.*, $\sum_{\forall w \in W} |w| = |W|$

- Sum of the TAM width for the 3D Stacked IC is *W*, *i.e.*, $\sum_{\forall w \in W, i \in N} |w| = |W|$

The large search space of the ILP model can be reduced by an upper and a lower bound to several variables:

- Achieving a trade-off between the test time and the test hardware can be portrayed as dividing the cumulative test time of all the scan chains in the 3D Stacked IC into as many equal (or as nearly equal as possible) groups as the number of TAM lines. It can be formulated as:

$$
Cost = \frac{2 \sum_{\forall b \in W, i \in N} t(sc)}{|W|} + \kappa \cdot N \cdot |W| \tag{5.18}
$$

To find the minima of the cost function in Eq. 5.17, we differenti-
ate Eq.5.18, *wrt W* and equate it to *zero*:

$$0 = \frac{d}{dW} Cost$$

$$= \frac{d}{dW}\left(\frac{2\sum_{\forall b\in W, i\in N} t(sc)}{|W|} + \kappa \cdot N \cdot |W|\right)$$

$$= \frac{-2\sum_{\forall b\in W, i\in N} t(sc)}{|W|^2} + \kappa \cdot N$$

$$\Rightarrow W_{est} = \left(\frac{2\sum_{\forall b\in W} t(sc)}{\kappa \cdot N}\right)$$

$$(5.19)$$

In the best case scenario, the scan chains in each chip can be
categorized into W_{est} equal groups to give the optimal test cost.
Although, in practice it might not always be plausible to comple-
ment Eq. 5.17 with exactly W_{est} to give the optimal cost.

- The maximum TAM width cannot exceed the ratio of the maxi-
  mum possible test time to that of the minimum possible test time,
  i.e.,

$$W_{max} = \frac{\sum_{\forall b\in W, i\in N} t(sc)}{\max_{\forall b\in W, i\in N}(t(sc))} \tag{5.20}$$

- Number of partitions of the TAM width, K does not exceed the
  minimum number of cores in any chip, *i.e.,*

$$K_{max} = \min_{\forall i\in N}(i) \tag{5.21}$$

- Maximum width of any group of TAM $|w|_{max}$ is the lowest ratio
  between the total time taken of all the scan chains to the longest
  scan chain for any chip in the 3D Stacked IC, *i.e.,*

$$|w|_{max} = \min\left(\max\left(\frac{\sum_{\forall b\in W, i\in N} t(sc)}{\max_{\forall b\in W, i\in N}(t(sc))}\right)\right) \tag{5.22}$$

- The test time of any scan chain on any chip may not surpass the
  sum of the maximum time taken by any scan chain in the chip

with the ratio of the total test time on the chip to that of the provided TAM width, *i.e.*,

$$\max_{\forall i \in N}(t(sc)) = \frac{\sum_{\forall sc(c) \in C(i), i \in N} t(sc)}{|W|} \tag{5.23}$$

Algorithm 4 ILP model for test planning of 3D Stacked ICs

1: **Given bounds:** $W_{est}, W_{max}, \alpha, K_{max}, |w|_{max}$
2: **if** $W_{est} \leq W_{max}$ **then**
3:     $W_1 = W_{est}$
4:     $W_2 = W_{max}$
5: **else**
6:     $W_1 = W_{max}$
7:     $W_2 = W_{est}$
8: **end if**
9: **for** $W = (W_1 - \kappa)$ to $(W_2 + \kappa)$ **do**
10:     **if** $W < K_{max}$ **then**
11:         $K^* = W$
12:     **else**
13:         $K^* = K_{max}$
14:     **end if**
15:     **for** $K = 1$ to K^* **do**
16:         **while** All partitions of W have not been explored **do**
17:             **for** $|w| = 1$ to $|w|_{max}$ **do**
18:                 **while** $\max \sum_{\forall sc(c) \in C(i), i \in N} t(sc) \leq \max_{\forall i \in N}(t(sc))$ **do**
19:                     Scan chains of each chip are assigned to each TAM line as per Algorithm 5
20:                 **end while**
21:                 **Save** $|w|$
22:             **end for**
23:         **end while**
24:         **Save** K
25:     **end for**
26:     **if** $Cost \leq Cost_{ILP}$ **then**
27:         $Cost_{ILP} = Cost$
28:     **end if**
29: **end for**

The ILP is implemented on the test function while considering the constraints and bounds mentioned above, as in Algorithm 4 to obtain

---

Algorithm 5 Scan Chain Allocation

1: **Given from Algorithm 4:** K
2: **Sort** Scan chains of each core $sc, \forall sc \in C(i), i \in N$ in descending order of time taken
3: **for** $sc = \max_{\forall sc \in C(i), i \in N}(sc)$ to $\min_{\forall sc \in C(i), i \in N}(sc)$ **do**
4:     **Sort** TAM lines $b, \forall b \in W$ in descending order of time taken at present
5:     **for** $\max(|w|)$ to $\min(|w|)$ **do**
6:         **Concatenate** sc to b to minimize $\max_{\forall b \in W}(t(b))$
7:     **end for**
8:     **Save** core-scan chain configuration
9: **end for**

---

the optimal test cost, $Cost_{ILP}$, optimal TAM width, W_{ILP}, partitions of the TAM width W and the width of each partition, $|w|$. The allocation of scan chains to respective TAM wires is performed by a bin packing approach motivated by [106], *viz.*, the Best Fit Decreasing (BFD) algorithm, as detailed in Algorithm 5.

5.5. Experiments

The objective of the experiments is to demonstrate that the proposed ILP scheme results in a lower test cost compared to when making use of schemes developed for non-stacked ICs. The proposed ILP scheme for 3D Stacked ICs is detailed above and the following two schemes for non-stacked ICs were used.

The proposed ILP model for test planning of 3D Stacked ICs with TSVs was implemented to achieve the minimal test cost for several 3D Stacked ICs with up to four chips in the stack. The wafer sort time for each chip, the package test time, and width of the TAM for the 3D Stacked IC are accounted at the lowest obtained test cost. To highlight the efficiency of the proposed ILP model, experiments are detailed to compare the test cost obtained for 3D Stacked ICs with varying TAM widths.

The 3D Stacked ICs in the experiments are obtained by combining several ITC'02 benchmarks. Each benchmark represents a chip in the stack. The four designs listed in Table 6.8 have been considered.

- Scheme 1, the TAM for each chip is optimized independently of all other chips in the 3D Stacked IC. It means that each chip gets

Table 5.4.: Designs

Label	Design	Cores	Contributor
D	d695	11	Duke University
G	g1023	15	University of Stuttgart
P	p34392	20	Philips Semiconductors
T	t512505	32	Texas Instruments

Table 5.5.: Scheme 1 where test architecture for each chip is optimized individually

Design	TAM architecture		Test time							Test cost
	TAM width	TAM cost	Wafer sort				Package test	Total time		cost
			Chip 1	Chip 2	Chip 3	Chip 4				
DP	30	15201815	23194	545763			2133589	1137914	16339729	
DT	32	51500032	21745	5166376			11858562	10376242	61876274	
GP	30	2210774	24381	545763			814491	1140287	3351061	
GT	32	25240421	22857	5166376			8302773	10378466	35618887	
DGP	30	1917965	23194	24381	711865		773919	1186676	3104640	
DGT	32	16221562	21745	22857	6612961		6670052	10421956	26643517	
DPT	32	10829574	21745	511653	5166376		5699774	11399547	22292026	
GPT	32	77025301	22857	511653	13777003		15202362	11401772	88489977	
DGPT	32	14201482	21745	22857	584746	5904430	6540149	11445261	25709647	

Table 5.6.: Scheme 2 where test architecture is optimized for the lowest chip in the stack and used for all chips

Design	TAM architecture		Test time							Test cost
	TAM width	TAM cost	Wafer sort				Package test	Total time		cost
			Chip1	Chip2	Chip3	Chip4				
DP	8	4053817	86979	2046611			2133589	4267179	8320996	
DT	8	12875008	86979	20665505			20752483	41504966	54379974	
GP	20	1473849	36571	818644			855216	1710431	3184280	
GT	20	15775263	36571	8266202			8302773	16605546	32380809	
DGP	8	511457	86979	91428	2046611		2225017	4450035	4961492	
DGT	8	4055390	86979	91428	20665505		20843911	41687822	45743212	
DPT	8	2707394	86979	2046611	20665505		22799094	45598188	48305582	
GPT	20	48140813	36571	818644	8266202		9121417	18242835	66383648	
DGPT	8	3550370	86979	91428	2046611	20665505	22890522	45781044	49331414	

Table 5.7.: Proposed 3D Stacked IC scheme where test architecture and test planning are co-optimized for all chips

Design	TAM architecture		Test time						Test
	TAM	TAM		Wafer sort			Package	Total	cost
	width	cost	Chip1	Chip2	Chip3	Chip4	test	time	
DP	8	2987025	86979	2046611			2133589	4267179	7254204
DT	12	16601987	49702	11808860			11858562	23717124	40319110
GP	22	1140287	34830	779661			814491	1628982	2769269
GT	18	11623882	36571	8266202			8302773	16605546	28229428
DGP	22	1083487	30253	31801	711865		773919	1547838	2631325
DGT	24	9338072	27833	29257	6612961		6670052	13340103	22678175
DPT	35	7979683	21745	511653	5166376		5699774	11399547	19379230
GPT	11	21283307	60952	1364407	13777003		15202362	30404725	51688032
DGPT	27	9156218	24851	26122	584746	5904430	6540149	13080298	22236517

Table 5.8.: Test cost comparison between schemes for non-stacked ICs and the proposed 3D Stacked IC scheme

Designs	Scheme 1 (Table 5.5)	Scheme 2 (Table 5.6)	3D Stacked IC scheme (Table 5.7)	Test cost (%) 3D Stacked IC versus	
	Test cost	Test cost	Test cost	Scheme 1	Scheme 2
DP	16339729	8320996	7254204	125.24	14.71
DT	61876274	54379974	40319110	53.47	34.87
GP	3351061	3184280	2769269	21.01	14.99
GT	35618887	32380809	28229428	26.18	14.71
DGP	3104640	4961492	2631325	17.99	88.55
DGT	26643517	45743212	22678175	17.49	101.71
DPT	22292026	48305582	19379230	15.03	149.26
GPT	88489977	66383648	51688032	71.20	28.43
DGPT	25709647	49331414	22236517	15.62	121.85
			Average:	40.36	63.23

the TAM that is most suitable for its wafer sort. Note that after the optimization additional TAM wires can be added to a chip. For example, if the top chip requires a very wide TAM while all other chips only need a narrow TAM, the wide TAM is added to all chips to make testing of the top chip possible at package test.

- Scheme 2, the TAM for the lowest chip is optimized and the same test architecture is used for all chips in the 3D Stacked IC. In this case, all chips use the TAM optimized for wafer sort test of the lowest chip.

For the experiments, core-based 3D Stacked ICs were constructed using four ITC'02 benchmarks: d695 (D), g1023 (G), p34392 (P), and t512505 (T), see Table 6.8. To give an indication of the complexity of the designs, Table 6.8 also details the number of cores. Each of the ITC'02 benchmarks form a chip and by combining the four benchmarks in various ways, 3D Stacked ICs with 2, 3 and 4 chips were constructed. For example, the DP design in Table 5.5 is a 3D Stacked IC with 2 chips consisting of d695 and p34392 where d695 is the lowest chip.

For the constant κ in the test cost (Eq. 5.10), we performed some experiments and found that suitable are the following settings: $\kappa = T_{max}/(0.5 \times (TAM_{max} - TAM_{min})^2)$ where T_{max} is the test time when all chips are tested assuming one single wrapper-chain, TAM_{max} is set to T_{max}/s where s is the length of the longest scan-chain of a chip, and $TAM_{min} = 1$. By setting κ in this way, we have a general scheme for all design. Obviously, a designer can set κ in the most appropriate way for a given design. For the search space with respect to TAM width W, we have a general limit that has been used for all experiments. The highest allowed TAM width is limited by computing the sum of the test times of all cores assuming a single wrapper-chain. To obtain the highest TAM limit, this value is divided with the test time of the core with highest test time.

The results from the non-stacked schemes are in Table 5.5 and 5.6. The results from the proposed 3D Stacked IC scheme are in Table 5.7. The comparison between the three schemes is in Table 5.8. Table 5.5, Table 5.6 and Table 5.7 are constructed in the same way. The designs are listed in column one, the TAM width and the TAM cost are in column two, the test time at wafer sort for each chip where applicable, the test time at package test, the total test time are in column three, and the test cost for each design is in column four.

The comparison between the two non-stacked schemes against the 3D Stacked IC scheme is in Table 5.8. Table 5.8 is organized as fol-

lows. Column one lists the designs. Columns two, three and four are organized in the same way with the test cost for the two non-stacked schemes and the proposed 3D Stacked IC scheme. Column five reports the comparison between the non-stacked schemes and the 3D Stacked IC scheme. The results show that the 3D Stacked IC scheme produces the lowest test cost in all cases. For the benchmark DP, the 3D Stacked IC scheme is 125% better than Scheme 1 and 14% better than Scheme 2. At an average over all designs, the 3D Stacked IC scheme is 40% better than Scheme 1 and 63% better than Scheme 2.

5.6. Discussion

As test planning for 3D Stacked ICs is different from test planning for non-stacked ICs, we address in this chapter the problem of test planning for 3D Stacked ICs. We assume that each chip in the stack is supported by an IEEE 1500 based test architecture, and we model the test planning problem, which is known to be *NP*-hard, with ILP. The aim is to minimize the overall test cost given as the test time and TAM for each individual chip at wafer sort and the complete stack of chips at package test. We assume a test flow where each chip is individually tested at wafer sort and all chips (the complete 3D Stacked IC) are jointly tested at package test. As ILP is very time consuming, we reduce the search space by determining a near-optimal TAM width through a bounding scheme. The proposed ILP model and the bounding scheme was simulated on a 3D Stacked IC obtained by stacking up to four ITC'02 benchmarks. In the experiments we compare the proposed scheme against two schemes for non-stacked ICs. A reduction in the test cost was accounted for each instance. The results show that proposed scheme results in a test cost at an average of 40% and 63%, lower than the two schemes for non-stacked ICs, respectively.

Part III

Test Flow Selection

In this part, test flow selection of 3D Stacked ICs is discussed. It is necessary to have a procedure to arrive at a test flow that provides a low test time, in order to reduce the overall test cost. Therefore we first illustrate the problem of test flow selection for 3D Stacked ICs. We put forward a test flow selection algorithm and demonstrate its effectiveness. For experiments we compare the test flows generated by our algorithm against straightforward test flows.

6

Test Flow Selection

In this chapter, the test flow selection of 3D Stacked ICs is discussed. A traditional non-stacked chip is tested twice at the two levels, viz. (i) wafer sort and (ii) package test. Wafer sort is motivated by the fact that packaging the faulty products is more expensive than the test itself. By testing, unnecessary packing of faulty chips is avoided. For non-stacked chips, faults might occur while packaging the same IC. Therefore, the test performed at wafer sort is repeated at the package test.

In case of 3D Stacked ICs, there are four steps in the stacking process when faults can be introduced to any individual chip of the stack: (i) die fabrication, (ii) when the bottom of the chip is bonded to the stack, (iii) when another chip is bonded to the top of the chip, (iv) packaging. Based on these steps, several test repetitions of tests can be considered. For a three-chip stack, these test repetitions of tests can be referred to as wafer sort, test after the first stacking event (for the two chips that are first stacked together), test after the second stacking event and package test. It should be noted that testing after a stacking event or package test includes testing the TSVs.

This chapter proceeds with presenting a cost model for 3D Stacked ICs to calculate the time related to any test flow in Section 6.1, followed by a motivating example in Section 6.2. A test flow selection algorithm is presented in Section 6.3. The proposed algorithm is executed on several 3D Stacked ICs with up to ten chips in the stack. Test flow selection is eventually integrated with test planning on several 3D Stacked IC designs, and the results are presented and discussed in Section 6.4.
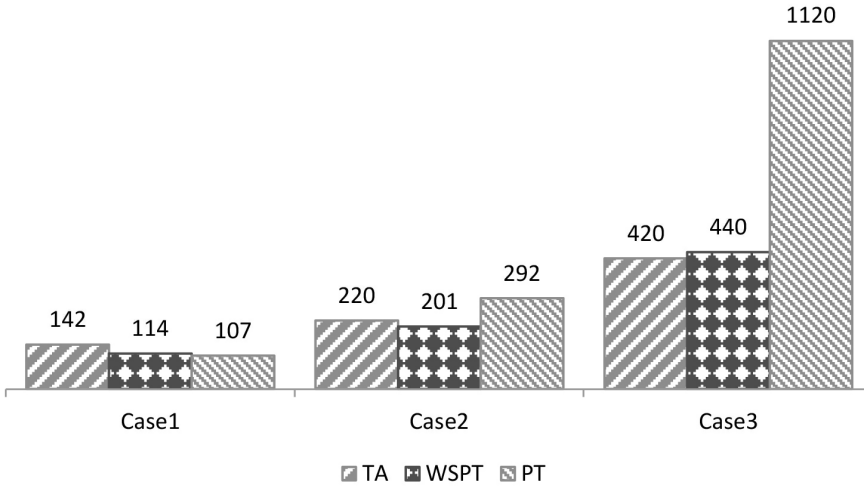
Figure 6.1.: Comparison of expected total test times for three test flows − TA, WSPT and PT respectively − on one design with three sets of yield values: Case 1, Case 2, and Case 3. PT, WSPT and TA require the lowest expected test times for Case1, Case2 and Case3 respectively.

6.1. Problem Formulation

In this section we derive an expression to calculate the expected total test time required to produce one fault-free 3D Stacked IC for any assumed test flow. For reference we assume Figure 6.1 and the design provided in Table 6.3 with yield Case 1.

We elaborate the notations given in Table 6.1 using Figure 6.2 and Figure **??**. First, the given notations corresponding to given data are discussed, followed by the notations that refer to values that need to be calculated for a selected test flow. Finally, we discuss the notations that represent values that hold true for all 3D Stacked ICs considered in this paper. We assume a given 3D Stacked IC with N chips in the stack, where each chip is denoted by i, $1 \leq i \leq N$. A test instance is denoted by I_{ij}, illustrated in Figure 6.3. Each test instance, I_{ij}, requires test time T_{ij} and has a yield y_{ij}.

The wafer sort instances are illustrated by the boxes in the upper row, where $j = 1$, which means I_{11} to I_{N1} are the instances for wafer
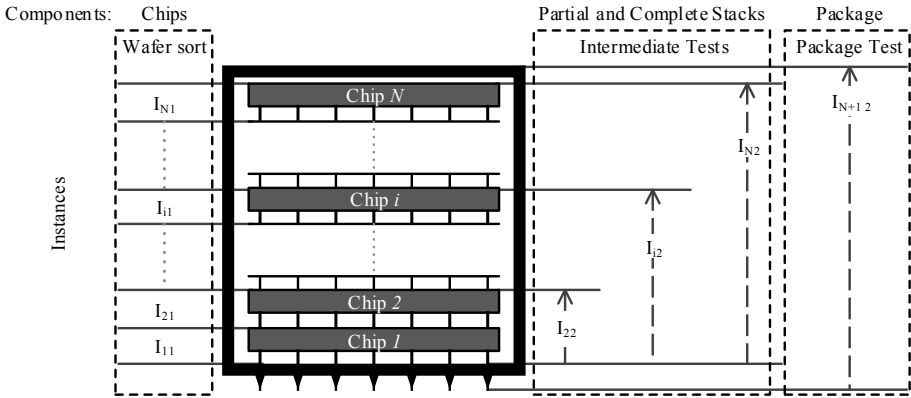
Figure 6.2.: Components of the 3D Stacked IC that are
tested at instance I_{ij}. Wafer sort of individ-
ual chips are marked on the left, from I_{11} to
I_{N1}. Intermediate tests, I_{22} to I_{N2} of the 3D
Stacked IC with 2 to N chips are to the right.
Package test $I_{N+1\ 2}$ is at the right.

sort. The wafer sort instance of a chip i is indicated with I_{i1} to the left
in Figure 6.2. We have $j = 2$ for intermediate tests of partial stacks
with i chips, and for package test instances, I_{i2}. In Figure 6.3, the right
column indicates the intermediate test instances and the package test.
For an intermediate test instance I_{i2}, the intermediate stack consists of
all chips from 1 to i. For example, if $i = 3$, the intermediate test instance
I_{32} consists of the partial stack of chip 1, 2 and 3. The intermediate test
instances and package test are illustrated to the right in Figure 6.2.

A partial stack, with i chips, is tested during the intermediate in-
stance, I_{i2}, which comprises of components from two previous in-
stances − a partial stack with $i − 1$ chips, and chip i. This is illustrated
by arrows connecting instance $I_{i-1\ 2} \rightarrow I_{i2}$, and $I_{i1} \rightarrow I_{i2}$, in Figure 6.3.

We now discuss the values that need to be computed to obtain
the desired test flow. A test flow is represented by the vector $X =$
$(x_{11}...x_{N1}), (x_{22}...x_{N\ 2}), (x_{N+1\ 2})$, with $2N$ elements, that include N wafer
sorts, $N − 1$ intermediate tests and 1 package test. Each element is de-
noted by the binary decision variables x_{ij}, for each box in Figure 6.3,
we set $x_{ij} = 1$ when a test is performed at instance I_{ij}, and $x_{ij} = 0$
when no tests are performed at instance I_{ij}. Let us consider the ex-
ample of a 3D Stacked IC with 2 chips in the stack, as illustrated in
Section 6.2. The vectors for TA, WSPT and PT would be represented
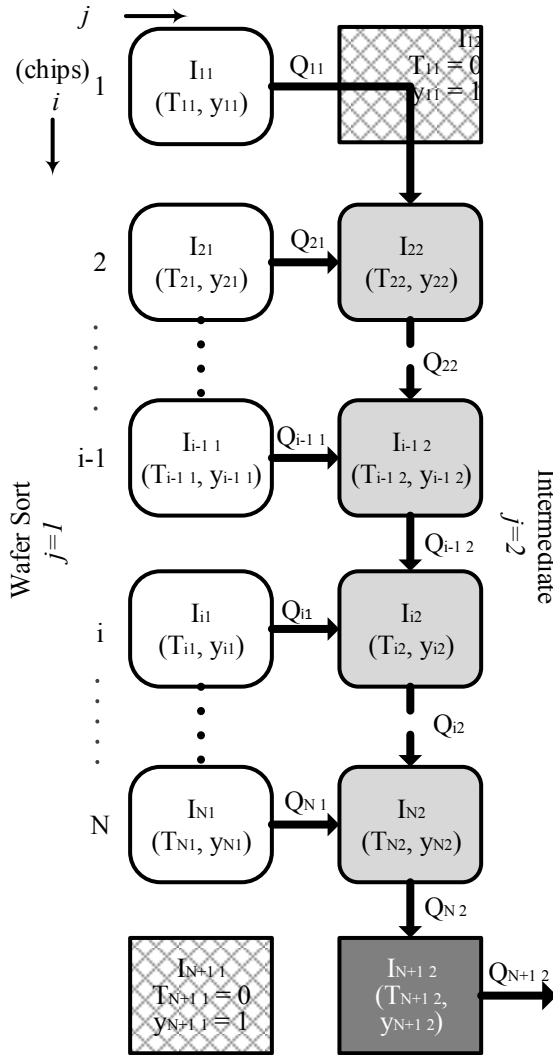
Figure 6.3.: Test instances I_{ij} of a 3D Stacked IC with
N chips in the stack. Upper row indicates
wafer sort instances of chips $i = 1$ to N; from
I_{11} to I_{N1}. Lower row indicates intermediate
test instances, after stacking chips $i = 2$ to
N to the incomplete stack, and package test
instances; I_{22} to I_{N2}. Connecting arrows be-
tween instances (boxes) indicate transfer of
components. Boxes representing instances
I_{12} and $I_{N+1\,1}$ are represented for the sake
of convenience.

Table 6.1.: List of notations

Input data

N	Number of chips constituting the stack
I_{ij}	Test instance corresponding to i^{th} chip
T_{ij}	Time taken to test a single unit at instance I_{ij}
y_{ij}	Yield of the manufacturing stage I_{ij}
i	i^{th} chip in the stack, $1 \leq i \leq N$
j	$j = 1 \implies$ wafer sort
	$j = 2 \implies$ intermediate and package test

Known data

I_{12}	Does not exist
	$T_{12} = 0$
	$y_{12} = 1$
$I_{N+1\,1}$	Does not exist
	$T_{N+1\,1} = 0$
	$y_{N+1\,1} = 1$
$Q_{N+1\,2}$	$= 1$
Q_{i2}	$= Q_{i-1\,2}$

Calculated

x_{ij}	Binary decision variable
	1; if test is performed at instance I_{ij}
	0; otherwise
$\bar{x}_{ij}$	$1 - x_{ij}$
X	Test flow vector composed of x_{ij}
	$X = (x_{i1} \vert 1 \leq i \leq N), (x_{i2} \vert 2 \leq i \leq N), (x_{N+1\,2} = 1)$
Q_{ij}	Number of good units that need to be produced at instance I_{ij}
$T_{eff}(ij)$	Expected time taken to produce each good unit at instance I_{ij}
Y_{ij}	Effective yield at instance I_{ij}
τ	Expected total test time taken by the test flow given by X

as: $X = (1,1),(1),(1)$, $X = (1,1),(0),(1)$ and $X = (0,0),(0),(1)$ respectively. For convenience, we also define $\bar{x}_{ij} = 1 - x_{ij}$.

τ is the total expected time taken by the test flow given by X. The total expected time depends on what tests are applied in a test flow. At each instance, the effective yield is computed depending on the tests that have been previously performed. To enable computation of test time, for a test instance I_{ij}, we let Q_{ij} denote the number of good units that need to be produced at instance I_{ij}, $T_{eff}(ij)$ denote the expected time taken to produce one good unit at instance I_{ij}, and Y_{ij} the effective yield. The objective of this paper is to minimize the expected total test time τ.

Finally, it is given for all 3D Stacked ICs that, instances I_{12} and $I_{N+1\ 1}$ do not exist. Instance I_{12} corresponds to the box at the top right of Figure 6.3, for the intermediate test of only chip 1, which does not exist as at least two stacked chips are tested at any intermediate test. Again, for a 3D Stacked IC comprising of N chips in the stack, instance $I_{N+1\ 1}$ corresponding to the box at the bottom left corner of Figure 6.3, refers to wafer sort of chip $N + 1$, also does not exist. Therefore, it may be assumed that these instances require no test time, *i.e.*, $T_{12} = T_{N+1\ 1} = 0$, and also have a perfect yield, $y_{12} = y_{N+1\ 1} = 1$, for the generic expressions.

In the following subsections 6.1.1, 6.1.2, and 6.1.3 respectively, we elaborate each component of the expression, *viz*, yield Y_{ij}, quantity Q_{ij} and time $T_{eff}(ij)$.

6.1.1. Yield

The effective yield at each test instance is presented here, which is given as a function of the given yield values of the preceding test instances, depending on whether a test was performed during that instance. We will discuss effective yield first for wafer sort and then for intermediate and package tests.

Let us consider an arbitrary wafer sort test instance I_{i1}, which has the given yield of y_{i1}. As there are no prior tests of the chip at wafer sort, the effective yield depends only on the yield at test instance I_{i1}.

For wafer sort instances the effective yield Y_{i1} at test instance I_{i1} is given as:

$$Y_{i1} = y_{i1} \qquad (6.1)$$

In the example the yield of chip 2 for case 1 at wafer sort is $y_{21} = 0.91$, which means that the effective yield Y_{21}, which is wafer sort test

of chip 2, is:

$$Y_{21} = y_{21} = 0.91 \tag{6.2}$$

Next, we discuss the effective yield at intermediate and package test instances. Let us consider an arbitrary intermediate or package test instance I_{i2}. For the intermediate test instances the given yield y_{i2}, represents the yield of producing an intermediate stack with i chips, assuming that both components, *i.e.*, the chip i and the intermediate stack with $i - 1$ chips, are tested and are defect free. Contrary to wafer sort instances, as seen from Figure 6.3, any intermediate test instance, $I_{i2} \forall i > 2$, receives components from the preceding partial stack with $i - 1$ and wafer manufacturing stage of chip i. Now, if no tests were performed at the wafer sort instance I_{i1}, the defective components will be passed on to the corresponding intermediate test I_{i2}, which would decrease the effective yield at the intermediate test instance I_{i2}. This would result in the effective yield to be the product of the yield of the individual test instances I_{i1} and I_{i2}, *i.e.*, $(y_{i2} \cdot y_{i1})$. For example, with the design in Section 6.2, the yield at the intermediate test instance of stacking chip 1 and chip 2, I_{22} is $0.90 \cdot 0.92 = 0.828$, assuming $X = (1,0), (1), (1)$, as wafer sort has not been performed for chip 2 at instance I_{21}. Similarly, the yield at any intermediate instance I_{i2} depends on tests performed during previous instances $I_{i'2}$, where, $i' < i$. Therefore, to calculate the effective yield at any intermediate test instance I_{ij} as a effect of the yield of the instance itself and that of the corresponding wafer sort instance I_{i1}, we use the expression $y_{ij}(y_{i1}^{\bar{x}_{i1}})$. If $x_{ij} = 0$ we get $y_{i1}^{\bar{x}_{i1}} = y_{i1}^1 = y_{i1}$ and $x_{ij} = 1$ gives $y_{i1}^{\bar{x}_{i1}} = y_{i1}^0 = 1$. Therefore, the expression gives y_{ij} when $x_{i1} = 1$ *i.e.*, when test was performed at wafer sort, and $y_{ij} \cdot y_{i1}$ when $x_{i1} = 0$ *i.e.*, when no tests were performed at wafer sort. However, the effective yield at any intermediate instance I_{i2} will also depend on the yield of the preceding intermediate instance $I_{i-1\,2}$ when test was not performed, which in turn will depend on the yield of $I_{i-2\,2}$ and so on, unless a test had been performed at any of these prior instances. In other words, the yield at any intermediate test instance I_{ij} is a consequence of of the preceding intermediate test instances up to $I_{i'j} \quad \forall\ i' < i$, which is the latest intermediate test instance when a test has been performed. Therefore, the first intermediate test instance I_{22} depends only on the preceding wafer sort instances I_{11} and I_{21}, as there are no previous intermediate

tests possible. Thus we can derive the effective yield at instance I_{22} as:

$$Y_{22} = y_{22} \cdot y_{11}^{\bar{x}_{11}} \cdot y_{21}^{\bar{x}_{21}} \tag{6.3}$$

For the 3D Stacked IC in Section 6.2 with yield Case 1 and test flow $X = (0,1),(1),(1)$, we have $x_{11} = 0$, $x_{21} = 1$, $x_{22} = 1$ and $x_{32} = 1$. Hence we can compute Y_{22} as:

$$\begin{aligned} Y_{22} &= y_{22} \cdot y_{11}^{\bar{x}_{11}} \cdot y_{21}^{\bar{x}_{21}} \\ &= 0.92 \cdot 0.90^1 \cdot 0.91^0 \\ &= 0.92 \cdot 0.90 \cdot 1 = 0.8280 \end{aligned} \tag{6.4}$$

Similarly, the effective yield at the following intermediate or package test instance I_{32} depends on all previous tests performed, and can be given as:

$$\begin{aligned} Y_{32} &= y_{32} \cdot (y_{22} \cdot (y_{11}^{\bar{x}_{11}} \cdot y_{21}^{\bar{x}_{21}}))^{\bar{x}_{22}} \\ &= y_{32} \cdot Y_{22}^{\bar{x}_{22}} \end{aligned} \tag{6.5}$$

As seen in Figure 6.3, all intermediate test instances, depicted by the right column, receive components from preceding wafer sort and intermediate test instances. The yield, depending on the preceding instances, is:

$$Y_{i2} = \begin{cases} y_{i2} \cdot y_{i1}^{\bar{x}_{i1}} \cdot Y_{i-1\,2}^{\bar{x}_{i-1\,2}}, & \text{for } 2 < i \leq N+1 \\ y_{22} \cdot y_{11}^{\bar{x}_{11}} \cdot y_{21}^{\bar{x}_{21}}, & \text{for } i = 2 \end{cases} \tag{6.6}$$

Therefore, in case of package tests, we will have:

$$Y_{N+1\,2} = y_{N+1\,2} \cdot y_{N+1\,1}^{\bar{x}_{N+1\,1}} \cdot Y_{N2}^{\bar{x}_{N2}} \tag{6.7}$$

where, it is given that $y_{N+1\,1} = 1$, as noted in Table 6.1 and we set $\bar{x}_{N+1\,1} = 1$.

For the given package test instance I_{32}, in this example, the preceding wafer sorts were performed, while the intermediate test instance was avoided; such that $X = (1,1),(0),(1)$. Therefore, the yield of only the intermediate test instance, I_{22}, affects the package test instance:

$$Y_{32} = y_{32} \cdot y_{31}^{\bar{x}_{31}} \cdot Y_{22}^{\bar{x}_{22}}$$

$$=y_{32} \cdot y_{31}^{\bar{x}_{31}} \cdot \{y_{22}^{\bar{x}_{22}} \cdot (y_{11}^{\bar{x}_{11}} \cdot y_{21}^{\bar{x}_{21}})^{\bar{x}_{22}}\}$$
$$=0.93 \cdot 1^1 \cdot \{0.92^1 \cdot (0.90^0 \cdot 0.91^0)^1\}$$
$$=0.93 \cdot 1 \cdot 0.92 \cdot 1 \cdot 1 = 0.8556 \tag{6.8}$$

6.1.2. Quantity

At any test instance, the number of units tested is greater than the number of good units obtained, due to imperfect (< 1) yield. Therefore, we calculate the expected quantity of good units required at the end of each instance such that a fixed number of good units are obtained from a succeeding manufacturing stage.

Let us start with the package test instance, illustrated by the bottom right box in Figure 6.3. With a yield of $y_{N+1\,2}(< 1)$, to obtain $Q_{N+1\,2} = 1$ good packages we need to test $Q_{N+1\,2}/y_{N+1\,2}$ packages. Therefore, at the preceding instance I_{N2}, we need to produce $Q_{N2} = Q_{N+1\,2}/y_{N+1\,2}$ good units. Now, to produce Q_{N2} good units at the intermediate test instance I_{N2}, we need to test $Q_{N2}/(y_{N2} \cdot y_{N1}^{\bar{x}_{N1}})$ units. It is useful to note here that the number of instances that need to be tested during the intermediate test instance I_{N2} increases by $1/y_{N1}$ times if test was not performed at the wafer sort instance I_{N1}. This is due to the share of the defective wafers that pass on to the intermediate stack. Consequently, we need to produce $Q_{N1} = Q_{N2}/(y_{N2} \cdot y_{N1}^{\bar{x}_{N1}})$ good intermediate stacks and back calculate the number of good units that need to be produced after each test instance up to Q_{11}. The quantity of good units required after each test instance Q_{ij} is formulated as follows.

For wafer sort instances:

$$Q_{i1} = \begin{cases} \frac{Q_{i2}}{y_{i2}}, & 2 < i \le N \\ \frac{Q_{22}}{y_{22} \cdot y_{11}^{\bar{x}_{11}} \cdot y_{21}^{\bar{x}_{21}}}, & i \le 2 \end{cases} \tag{6.9}$$

For the 3D Stacked IC in Section 6.2 with yield Case 1 and $X = (0,0),(0),(1)$, the number of units required at the end of instance I_{11} is:

$$Q_{11} = \frac{1}{y_{11}^{\bar{x}_{11}} \cdot y_{21}^{\bar{x}_{21}} \cdot y_{22}^{\bar{x}_{22}} \cdot y_{32}^{\bar{x}_{32}}}$$
$$= \frac{1}{0.90^1 \cdot 0.91^0 \cdot 0.92^1 \cdot 0.93^1} = 1.299 \tag{6.10}$$

Table 6.2.: Effective test time at each test instance Table 6.3

Test Flow	Teff(11)	Teff(21)	Teff(22)	Teff(32)	τ
Case 1					
TA	12.99	12.84	35.06	75.27	136.16
WSPT	12.99	12.84	0.00	81.81	107.64
PT	0.00	0.00	0.00	99.89	99.89
Case 2					
TA	27.18	26.80	57.08	95.89	206.94
WSPT	27.18	26.80	0.00	133.18	187.16
PT	0.00	0.00	0.00	267.97	267.97
Case 3					
TA	72.57	71.15	108.85	132.08	384.64
WSPT	72.57	71.15	0.00	253.99	397.71
PT	0.00	0.00	0.00	996.04	996.04

For intermediate test instances:

$$Q_{i2} = \begin{cases} \frac{Q_{i+1\,2}}{y_{i+1\,2} \cdot y_{i+1\,1}^{\bar{x}_{i+1\,1}}}, & 2 \le i \le N-1 \\ \frac{Q_{N+1\,2}}{y_{N+1\,2}}, & i = N \end{cases} \tag{6.11}$$

In the example, the quantity required for package test, when only the wafer sorts have been performed, such that $X = (1,1),(0),(1)$ is:

$$Q_{22} = \frac{Q_{32}}{y_{32}} = \frac{1}{0.93} = 1.075 \tag{6.12}$$

It should be noted that at any intermediate test instance I_{i2}, for each intermediate stack comprising of chips 1 to $i-1$ obtained from instance $I_{i-1\,2}$, one unit of chip i from instance I_{i1} is stacked. Therefore, we need equal number of units from preceding instances I_{i1} and $I_{i-1\,2}$, giving $Q_{i1} = Q_{i-1\,2}$.

6.1.3. Time

An expression to calculate the expected total time taken by any test flow to produce one fault-free packaged 3D Stacked IC is formulated here. The time expected at each test instance depends on the time taken to test each unit at the instance, the effective yield, as well as

the number of units required at successive test instances to produce the desired number of good packages. Table 6.2 is used to list the expected test time at each instance for different test flows required by the 3D Stacked ICs mentioned in Table 6.3.

The effective test time, $T_{eff}(ij)$, spent at any instance, I_{ij}, depends on the given test time, T_{ij}, effective yield, Y_{ij}, at the instance and the number of defect-free units that need to be obtained, Q_{ij}, and the binary decision variable, x_{ij}, is given by:

$$T_{eff}(ij) = \frac{T_{ij}}{Y_{ij}} \cdot Q_{ij} \cdot x_{ij} \qquad (6.13)$$

For example, the effective time spent at the intermediate test instance I_{22} when all tests are performed, as seen in the first row of Case 1; such that $X = (1,1), (1), (1)$ is:

$$
\begin{aligned}
T_{eff}(22) =& \frac{T_{22}}{Y_{22}} \cdot Q_{22} \cdot x_{22} \\
=& \frac{30}{0.92 \cdot 0.90^0 \cdot 0.91^0} \cdot 1.075 \cdot 1 = 35.06 \qquad (6.14)
\end{aligned}
$$

The sum of the effective test times, $T_{eff}(ij)$, at each instance, I_{ij}, gives the expected total test time required, τ, by any selected test flow, X, to produce each fault-free packaged 3D Stacked IC as shown below.

$$\tau = \sum_{i=1}^{N+1} T_{eff}(ij) \qquad \forall j = 1, 2 \qquad (6.15)$$

The expected total test time, assuming a test flow when all instances are tested, as seen in the topmost row of Case 1, gives:

$$
\begin{aligned}
\tau =& T_{eff}(11) + T_{eff}(21) + T_{eff}(22) + T_{eff}(32) \\
=& 12.99 + 12.84 + 35.06 + 75.27 = 136.16 \qquad (6.16)
\end{aligned}
$$

The objective is to find a suitable test flow for any given 3D Stacked IC, such that the expected total test time τ is minimized.

6.2. Motivating Example

In this section we show with an example the effect of test flow on the expected total test time. The example considers a 3D Stacked IC with

Table 6.3.: SIC with three different sets of yield

Test instance	WS1	WS2	IT	PT
I_{ij}	I_{11}	I_{21}	I_{22}	I_{32}
Test time T_{ij}	10	10	30	70
Yield y_{ij}(Case 1)	0.90	0.91	0.92	0.93
Yield y_{ij}(Case 2)	0.70	0.71	0.72	0.73
Yield y_{ij}(Case 3)	0.50	0.51	0.52	0.53

two chips. Four test instances exist for the 3D Stacked IC, *viz*, wafer sort of each individual chip (WS1 and WS2, respectively), intermediate test (IT) of the two chips, and package test (PT) of the final 3D Stacked IC.

Table 6.3 details the given testing times for each instance. To demonstrate the impact of test flows, we consider three sets of yield values for each test instance, as shown in Table 6.3. For example, for yield Case 1, the yield at the wafer sort of Chip 1 is 0.90.

We use three straightforward test flow schemes:

- Test all (TA): tests are applied at every possible test instance;

- Wafer sort and package test (WSPT): each individual chip is tested at wafer sort and the complete 3D Stacked IC is tested at package test;

- Package test (PT): testing is only applied to the complete 3D Stacked IC at the final test instance, package test;

We compare the expected total test time required to obtain one good 3D Stacked IC, by assuming TA, WSPT and PT as the test flows. Figure 6.1 illustrates the expected total test time required by the three test flows with the three sets of yield values. Computation details of the expected test time required by the test flows are elaborated in the following Section.

The results show that for Case 1 PT has the lowest expected test time, while in Case 2, the test flow with the lowest expected test time is WSPT, and that in Case 3 TA results in the lowest expected test time. Thus, it can be concluded from the results that a predetermined test flow may not provide the lowest expected test time for any given 3D Stacked IC. In this paper we present a method to obtain a test flow for any given 3D Stacked IC, such that the expected total test time is minimized.

6.3. **Proposed Approach: TFSA**

In this section, we first detail the Test Flow Selection Algorithm (TFSA) and then we detail the computational complexity of the algorithm.

Given the test time T_{ij} and yield y_{ij} at all test instances I_{ij}, the TFSA generates a test flow, X, by iteratively trying to reduce the expected total test time τ. At each iteration, the test instance that contributes to most reduction in τ is selected. As discussed in the previous section, we represent a test flow with the vector $X = (x_{11}...x_{N1}), (x_{22}...x_{N\,2}), (x_{N+1\,2})$, where $(x_{N+1\,2}) = 1$, since package test is always performed.

TFSA, which is detailed in Algorithm 6, in line 1, takes as input N chips where for each test instance I_{ij} ($1 \leq i \leq N$, $1 \leq j \leq 2$) the test time T_{ij} and y_{ij} are given. We use the 3D Stacked IC described in Section **??**, with yield Case 3 in Table 6.3 for illustration. In the example, there are 2 chips ($N = 2$); hence, there are $2 \cdot N = 4$ possible test instances.

Algorithm 6 Test Flow Selection Algorithm (TFSA)

1: **Input:** A 3D Stacked IC with N chips
       for each test instance I_{ij}: test time T_{ij} and yield y_{ij}
2: **Initialize:**
       X as $x_{ij} = 0$ for $i \in 1..N; j \in [1,2]$ and $x_{N+1\,2} = 1$
       Compute expected total test time τ using Eq. 6.15
3: **for** $Counter = 1$ to $(2N - 1)$ **do**
4:    **Set:** $\acute{i} = 0$ and $\acute{j} = 0$
5:    **for** $i = 1$ to N **do**
6:       **for** $j = 1$ to 2 **do**
7:          **if** $x_{ij} = 0$ **then**
8:             **Set:** $x_{ij} = 1$
9:             **Compute:** $\acute{\tau}$
10:            **if** $\acute{\tau} < \tau$ **then**
11:               **Set:** $\tau = \acute{\tau}$
12:               **Set:** $\acute{i} = i$ and $\acute{j} = j$
13:            **end if**
14:            **Revert:** $x_{ij} = 0$
15:          **end if**
16:       **end for**
17:    **end for**
18:    **Set:** $x_{\acute{i}\acute{j}} = 1$
19: **end for**
20: **Output:** X, τ

The test flow vector X, and the corresponding test cost τ are initialized in line 2. All binary decision variables, x_{ij}, are initialized such that only package test is applied. For the example, we set $X = (0,0), (0), (1)$. After initialization, the expected time is computed with Eq. 6.15 to:

$$
\begin{aligned}
\tau &= \sum_{\forall i,j} T_{eff}(ij) \\
&= T_{eff}(11) + T_{eff}(21) + T_{eff}(22) + T_{eff}(32) \\
&= 0 + 0 + 0 + \frac{T_{32}}{Y_{32}} \cdot Q_{32} \cdot x_{32} \\
&= \frac{T_{32}}{y_{32} \cdot (y_{22} \cdot (y_{11}^{\bar{x}_{11}} \cdot y_{21}^{\bar{x}_{21}}))^{\bar{x}_{22}}} \cdot 1 \cdot 1 \\
&= \frac{70}{0.53 \cdot (0.52 \cdot (0.50^1 \cdot 0.51^1))^1} \cdot 1 \cdot 1 \qquad\qquad = 996 \qquad (6.17)
\end{aligned}
$$

As noted in Table 6.1, $T_{eff}(12) = 0$ and $T_{eff}(31) = 0$.

When only package test is applied, the actual yield at package test takes the yield at all instances into account.

A variable, *Counter*, is active between lines $3 \to 19$, to ascertain $2N - 1$ iterations. In this example, *Counter* iterates from $1 \to 3$. Variables $\acute{i}$ and $\acute{j}$ are reset for the iteration, in line 4.

To scan through all $2N - 1$ test instances I_{ij}, variables i and j are defined between lines $5 \to 17$ and lines $6 \to 16$, respectively, where $1 \le i \le N$ and $1 \le j \le 2$.

During an iteration, each inactive test instance $x_{ij} = 0$ is set to $x_{ij} = 1$, in line $7 \to 8$. The corresponding test cost $\acute{\tau}$ is computed in line 9, as a result of the modified test flow, to evaluate if there is a benefit to include the test instance in the test flow. In the first iteration, the matrix is updated to $(1,0), (0), (1)$.

At line 9, the effective total test time $\acute{\tau}$ for the current test flow $X = (1,0), (0), (1)$ is computed as:

$$
\begin{aligned}
\acute{\tau} &= \sum_{\forall i,j} T_{eff}(ij) \\
&= T_{eff}(11) + T_{eff}(21) + T_{eff}(22) + T_{eff}(32) \\
&= \frac{T_{11}}{Y_{11}} \cdot Q_{11} \cdot x_{11} + 0 + 0 + \frac{T_{32}}{Y_{32}} \cdot Q_{32} \cdot x_{32}
\end{aligned}
$$

$$
=\frac{T_{11}}{y_{11}} \cdot \frac{1}{y_{12} \cdot y_{22} \cdot y_{32}} \cdot 1 + 0 + 0
$$

$$
+ \frac{T_{32}}{y_{32} \cdot (y_{22} \cdot (y_{11}^{\bar{x}_{11}} \cdot y_{21}^{\bar{x}_{21}}))^{\bar{x}_{22}}} \cdot 1 \cdot 1
$$

$$
=\frac{10}{0.50} \cdot \frac{1}{0.51 \cdot 0.52 \cdot 0.53} \cdot 1
$$

$$
+ \frac{70}{0.53 \cdot (0.52 \cdot (0.50^0 \cdot 0.51^1))^1} \cdot 1 \cdot 1 \qquad\qquad = 640 \qquad (6.18)
$$

Note, in this case, wafer sort is applied to chip 1, which means y_{11} is used at test instance I_{11}, and not in test instance I_{32}.

If the new test cost $\acute{\tau}$ is lower than all previously computed test costs τ (line 10), the test cost is updated as $\tau = \acute{\tau}$ and the indices i and j are recorded, at lines 11 and 12, respectively. In the example, $\acute{\tau} = 640$ and $\tau = 996$. Hence, a better solution is found, thus replacing τ by $\acute{\tau}$. The algorithm continues with the test flow $(0,1),(0),(1)$ which gives $\acute{\tau} = 640$, same as the present value of $\tau = 640$. Hence, $\acute{i}$ and $\acute{j}$ are not updated. For the test flow $(0,0),(1),(1)$, we get $\acute{\tau} = 558 < \tau = 640$, and update indices $\acute{i} = 2$ and $\acute{j} = 2$.

Hence, at the end of the first iteration, in line 18, we update the test flow to $X = (0,0),(1),(1)$.

Similarly, at the end of the second iteration $Counter = 2$, we will have $X = (1,0),(1),(1)$ and $\tau = 487$. Eventually, at the third and final iteration $Counter = 2N - 1 = 3$, we will have $X = (1,1),(1),(1)$ and $\tau = 384$. Thus, for this example, $(1,1),(1),(1)$, means wafer sort is applied to both chip 1 and chip 2, intermediate test is applied to the stack of chips $1 + 2$ and package test is applied to the complete stack. The expected total test time is 384.

6.3.1. Complexity Estimation

There are two nested iterations in algorithm 6. The outer iteration, *for* loops between lines $3 \rightarrow 19$, iterates *Counter* from 1 to $2N - 1$, and the inner iteration, *for* loops between lines $5 \rightarrow 17$, iterates i from 1 to N. For each i, j takes two values 1 and 2. Thus, the complexity is the product of the number of iterations of each loop, *i.e.*, $(2N - 1) \cdot N \cdot 2 = 4N^2 - 2N$, which is of order $O(N^2)$.

6.4. Experiments

In this section we present two sets of experiments. First we compare the expected total test times obtained from TFSA with respect to three predetermined test flows and the test flow obtained by exhaustive search. Next, the TFSA is integrated with test planning of core-based 3D Stacked ICs with a IEEE 1500 based test architecture, to optimize the test time.

6.4.1. Test Flow Selection

The objective is to compare the expected total test time by applying TFSA and that required for the three straight forward test flows (TA, WSPT, and PT) as well as against exhaustive search.

Experiments were performed on two sets of 3D Stacked IC designs with 2 to 10 chips in the stack. The 3D Stacked IC designs are detailed in Table 6.5. For example, in case of both Set 1 and Set 2, SIC_2 consists of three chips in the stack, chip 1, 2, and 3. The test time and yield values of each chip in the 3D Stacked IC designs at wafer sort are given for Set 1 and Set 2 in Table 6.4. In Set 1, chip 1 has a test time of 1000 time units and a yield of 0.62. The test times are kept constant for all chips in the stack for both Set 1 and Set 2, to emphasize the difference among the expected total test times. However, the yield values for Set 1 and Set 2 are changed to emphasize the differences among the test flows obtained. For intermediate tests and package test, we assume an additional test time of 1000 units and the given yield at the test instance to be 0.70. For instance, the test time assumed during the package test of SIC_3 is the sum of the the test times of each individual chip − chips 1, 2 and 3 −, two layers of interconnects − between chips 1 and 2, and chips 2 and 3 − give an additional 2×1000, and finally 1000 time units for testing the package itself.

The results of the comparison between TFSA, TA, PT, WSPT and exhaustive search are collated in Table 6.6. Table 6.6 is organized as follows. The leftmost column lists the 3D Stacked IC designs. The following group of five columns list the expected total test times required by each method. The rightmost group of four columns depicts for each method the overhead in expected total test time time compared to the optimal expected total test time obtained by exhaustive search. The most significant points that can be drawn from Table 6.6 are:

- TFSA generates test flows and corresponding test times very close to exhaustive search in most cases.

Table 6.4.: Test times and yields of Chips 1 to 10

Chip	1	2	3	4	5	6	7	8	9	10
	Set 1									
Test time	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
Yield	0.62	0.66	0.70	0.74	0.78	0.82	0.86	0.90	0.94	0.98
	Set 2									
Test time	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
Yield	0.98	0.94	0.90	0.86	0.82	0.78	0.74	0.70	0.66	0.62

Table 6.5.: Designs

3D Stacked IC designs	Chips in the 3D Stacked ICs as detailed in Table 6.4. First chip is lowermost.
SIC_1	1, 2
SIC_2	1, 2, 3
SIC_3	1, 2, 3, 4
SIC_4	1, 2, 3, 4, 5
SIC_5	1, 2, 3, 4, 5, 6
SIC_6	1, 2, 3, 4, 5, 6, 7
SIC_7	1, 2, 3, 4, 5, 6, 7, 8
SIC_8	1, 2, 3, 4, 5, 6, 7, 8, 9
SIC_9	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

- PT has a low expected total test time for 3D Stacked ICs with up to three chips in the stack: for SIC_1 of Set 1, the result is only 1% away from optimum, whereas the optimal is obtained for SIC_1 and SIC_2 of Set 2. However, for all other cases, PT produces results that are far from optimum. As the number of chips in the 3D Stacked IC increases, the performance of PT deteriorates.

- TA results give expected total test times that are about 40% more than optimum for Set 1 and over 80% worse at an average for Set 2.

- WSPT is not as efficient as the TFSA. However, it is interesting to note that WSPT produces optimal results when the number of chips is less than 4, and WSPT is only a few % away from optimum when the number of chips in the stack is less than eight.

Table 6.6.: Comparison of expected total test times

3D Stacked	Total Test Time (τ)					Difference(%)			
ICs	Exhaustive	TFSA	TA	PT	WSPT	TFSA	TA	PT	WSPT
	Set 1								
SIC_1	10874	10874	13812	10972	10874	0	27	1	0
SIC_2	21656	21656	29402	33588	21656	0	36	55	0
SIC_3	38277	38277	53088	86456	38277	0	39	126	0
SIC_4	63842	63842	88354	197930	63842	0	38	210	0
SIC_5	101631	101631	140179	413792	103042	0	36	302	1
SIC_6	159216	178426	215669	801922	162933	12	35	404	2
SIC_7	184017	184017	324978	1454734	254110	0	77	691	38
SIC_8	346572	346572	482610	2487197	392444	0	39	618	13
SIC_9	503730	503730	709280	4028502	601658	0	41	700	19
	Set 2								
SIC_1	4874	4874	11682	4874	8743	0	140	0	79
SIC_2	11604	11604	25711	11604	17965	0	122	0	55
SIC_3	25702	25702	47430	25702	32619	0	85	10	27
SIC_4	45946	45946	80144	55971	55632	0	74	22	21
SIC_5	75558	75558	128577	123013	91441	0	70	63	21
SIC_6	121359	127854	199481	277057	146744	5	64	128	21
SIC_7	188408	188408	302500	646197	231631	0	61	243	23
SIC_8	284257	284257	451419	1573533	361253	0	59	454	27
SIC_9	421316	421316	665931	4028502	558309	0	58	856	33

Table 6.7 lists the test flows obtained by the exhaustive search and the TFSA, respectively. It is interesting to note, first of all, the test flows proposed by the TFSA are, in most cases, the same as the test flows obtained from exhaustive search. However, for exhaustive search the expected total test time for 2^{2N-1} test flows need to be evaluated, whereas TFSA only compares $(2N-1)^2$ test flows. Therefore, TFSA requires lower computation time, as compared to exhaustive search, to determine a test flow for 3D Stacked ICs with more than 2 chips in the stack. Secondly, the optimal test flow does not follow a regular pattern. For example, SIC_6 and SIC_7 of Set 1 differ by only one chip. But the test flows are very different. In addition, it is also observed that performing wafer sort pays off in most cases.

Table 6.7.: Comparison of test flows obtained with exhaustive search against TFSA.

Design	Test flow for exhaustive search	Test flow for TFSA
	Set 1	
SIC_1	(1,1), (0), (1)	(1,1), (0), (1)
SIC_2	(1,1,1), (0,0), (1)	(1,1,1), (0,0), (1)
SIC_3	(1,1,1,1), (0,0,0), (1)	(1,1,1,1), (0,0,0), (1)
SIC_4	(1,1,1,1,1), (0,0,0,0), (1)	(1,1,1,1,1), (0,0,0,0), (1)
SIC_5	(1,1,1,1,1,1), (0,1,0,0,0), (1)	(1,1,1,1,1,1), (0,1,0,0,0), (1)
SIC_6	(0,0,1,1,1,1,1), (1,0,1,0,0,0), (1)	(1,1,1,1,1,1,1), (0,1,0,0,0,0), (1)
SIC_7	(1,1,1,1,1,1,1,1), (0,0,0,0,0,1,0), (1)	(1,1,1,1,1,1,1,1), (0,1,0,0,0,0,0), (1)
SIC_8	(1,1,1,1,1,1,1,1,1), (0,1,0,0,1,0,0,0), (1)	(1,1,1,1,1,1,1,1,1), (0,1,0,0,1,0,0,0), (1)
SIC_9	(1,1,1,1,1,1,1,1,1,1), (0,1,0,0,1,0,0,0,0), (1)	(1,1,1,1,1,1,1,1,1,1), (0,1,0,0,1,0,0,0,0), (1)
	Set 2	
SIC_1	(0,0), (0), (1)	(0,0), (0), (1)
SIC_2	(0,0,0), (0,0), (1)	(0,0,0), (0,0), (1)
SIC_3	(0,0,1,1), (1,0,0), (1)	(0,0,1,1), (0,1,0), (1)
SIC_4	(0,0,0,1,1), (0,0,1,0), (1)	(0,0,0,1,1), (0,0,1,0), (1)
SIC_5	(0,0,0,1,1,1), (0,0,1,1,0), (1)	(0,0,0,1,1,1), (0,0,1,1,0), (1)
SIC_6	(0,0,1,1,1,1,1), (0,0,1,1,0,0), (1)	(0,0,0,1,1,1,1), (0,0,1,1,0,0), (1)
SIC_7	(0,0,0,1,1,1,1,1), (0,0,1,1,1,0,0), (1)	(0,0,0,1,1,1,1,1), (0,0,1,1,1,0,0), (1)
SIC_8	(0,0,0,1,1,1,1,1,1), (0,0,1,1,1,0,0,0), (1)	(0,0,0,1,1,1,1,1,1), (0,0,1,1,1,0,0,0), (1)
SIC_9	(0,0,0,1,1,1,1,1,1,1), (0,0,1,1,1,0,0,0,0), (1)	(0,0,0,1,1,1,1,1,1,1), (0,0,1,1,1,0,0,0,0), (1)

Table 6.8.: Experimental data

Label	Design	Cores	Time	Yield
D	d695	11	695828	0.65
G	g1023	15	731423	0.65
P	p34392	20	16372887	0.75
T	t512505	32	165324037	0.75
Intermediate test			10000	0.65
Package test			10000	0.75

Table 6.9.: Expected total test times required by 3D Stacked ICs for different test flow and test architecture schemes, part 1

Test flow	Test architecture scheme			vs SIC (%)	
	Scheme SIC	Scheme 1	Scheme 2	Scheme 1	Scheme 2
	SIC: DP; TAM width = 10				
TFSA	5.3E+6	5.9E+6	6.7E+6	11	14
WSPT	5.3E+6	5.9E+6	6.7E+6	11	14
TA	1.0E+7	1.2E+7	1.4E+7	20	17
PT	8.3E+6	9.5E+6	1.1E+7	14	16
	vs TFSA (%)				
WSPT	0	0	0		
TA	94	102	104		
PT	56	62	65		
	SIC: DT; TAM width = 16				
TFSA	3.0E+7	3.7E+7	4.5E+7	23	22
WSPT	3.0E+7	3.7E+7	4.5E+7	23	22
TA	6.2E+7	7.6E+7	9.6E+7	23	26
PT	5.0E+7	5.9E+7	7.2E+7	18	22
	vs TFSA (%)				
WSPT	0	0	0		
TA	106	105	111		
PT	66	60	57		
	SIC: GP; TAM width = 22				
TFSA	2.5E+6	2.8E+6	3.1E+6	12	11
WSPT	2.5E+6	2.8E+6	3.1E+6	12	11
TA	4.7E+6	5.3E+6	5.6E+6	13	6
PT	3.8E+6	4.4E+6	4.9E+6	16	11
	vs TFSA (%)				
WSPT	0	0	0		
TA	92	88	82		
PT	56	56	60		

Table 6.10.: Expected total test tmes required by 3D Stacked ICs for different test flow and test architecture schemes, part 2

	Test architecture scheme			vs SIC (%)	
Test flow	Scheme SIC	Scheme 1	Scheme 2	Scheme 1	Scheme 2
	SIC: GT; TAM width = 22				
TFSA	2.2E+7	2.6E+7	3.2E+7	18	23
WSPT	2.2E+7	2.6E+7	3.2E+7	18	23
TA	4.5E+7	5.1E+7	5.6E+7	13	10
PT	3.6E+7	4.2E+7	5.2E+7	17	24
	vs TFSA (%)				
WSPT	0	0	0		
TA	106	94	76		
PT	65	61	64		
	SIC: DGP; TAM width = 25				
TFSA	4.1E+6	4.5E+6	5.2E+6	10	16
WSPT	4.4E+6	5.0E+6	5.8E+6	14	16
TA	5.2E+6	6.0E+6	7.0E+6	15	17
PT	4.6E+6	5.3E+6	6.2E+6	15	17
	vs TFSA (%)				
WSPT	10	12	12		
TA	25	31	32		
PT	12	16	17		
	SIC: DGT; TAM width = 25				
TFSA	3.7E+7	4.1E+7	4.7E+7	11	15
WSPT	4.1E+7	4.6E+7	5.3E+7	12	15
TA	4.7E+7	5.4E+7	6.2E+7	15	15
PT	4.2E+7	4.8E+7	5.6E+7	14	17
	vs TFSA (%)				
WSPT	12	14	15		
TA	27	33	34		
PT	14	18	20		

Table 6.11.: Expected total test times required by 3D Stacked ICs for different test flow and test architecture schemes, part 3

	Test architecture scheme			vs SIC (%)	
Test flow	Scheme SIC	Scheme 1	Scheme 2	Scheme 1	Scheme 2
	SIC: DPT; TAM width = 30				
TFSA	3.5E+7	3.8E+7	4.4E+7	9	16
WSPT	3.7E+7	4.2E+7	4.9E+7	14	17
TA	4.4E+7	5.1E+7	5.9E+7	16	16
PT	3.8E+7	4.3E+7	5.1E+7	13	19
	vs TFSA (%)				
WSPT	7	9	10		
TA	25	30	33		
PT	9	13	15		
	SIC: GPT; TAM width = 16				
TFSA	6.5E+7	7.2E+7	8.3E+7	11	15
WSPT	7.0E+7	7.9E+7	9.1E+7	13	15
TA	8.2E+7	9.5E+7	1.1E+8	16	16
PT	7.1E+7	8.1E+7	9.5E+7	14	17
	vs TFSA (%)				
WSPT	7	9	10		
TA	26	32	33		
PT	9	13	15		
	SIC: DGPT; TAM width = 30				
TFSA	3.5E+7	4.3E+7	6.3E+7	23	47
WSPT	5.0E+7	6.4E+7	9.9E+7	28	55
TA	5.2E+7	6.2E+7	9.2E+7	19	48
PT	6.8E+7	9.0E+7	1.3E+8	32	44
	vs TFSA (%)				
WSPT	45	48	57		
TA	45	45	45		
PT	97	108	110		

Table 6.12.: Test flow used with SIC Scheme to minimize
test time

Design	Exhaustive	TFSA
DP	(1, 1), (0), (1)	(1, 1), (0), (1)
DT	(1, 1), (0), (1)	(1, 1), (0), (1)
GP	(1, 1), (0), (1)	(1, 1), (0), (1)
GT	(1, 1), (0), (1)	(1, 1), (0), (1)
DGP	(1, 1, 0), (1, 0), (1)	(1, 1, 0), (1, 0), (1)
DGT	(1, 1, 0), (1, 0), (1)	(1, 1, 0), (1, 0), (1)
DPT	(1, 1, 0), (1, 0), (1)	(1, 1, 0), (1, 0), (1)
GPT	(1, 1, 0), (1, 0), (1)	(1, 1, 0), (1, 0), (1)
DGPT	(1, 1, 1, 0), (1, 1, 0), (1)	(1, 1, 1, 0), (1, 1, 0), (1)

6.4.2. Test Architecture Design

In the second set of experiments, the goal is to further reduce test cost by integrating test flow selection and test planning. It is achieved by comparing the expected total test times obtained by integrating test architecture design and test planning schemes with different test flows. We evaluate (1) TFSA against three straightforward test flows (TA, WSPT, and PT) against an exhaustive search of all possible test flows, and (2) the test flows on three test architecture designs and test planning schemes. The objective here is to integrate test flow selection and test architecture design to obtain the minimal test cost.

We assume that the given 3D Stacked ICs are core-based, and each core is provided with a IEEE 1500 based core test wrapper as presented in 5.1. The problem at system-level is given a TAM width (W) to find the most suitable number of TAM groups, their widths, and assign the cores to the TAM groups such that test time is minimized. The three test planning schemes that we assume are:

- Scheme 1, the TAM for each chip is optimized independently of all other chips in the 3D Stacked IC. It means that each chip gets the TAM that is most suitable for its wafer sort. Note that after the optimization additional TAM wires can be added to a chip. For example, if the top chip requires a wide TAM while all other chips only need a narrow TAM, the wide TAM is added to all chips to make testing of the top chip possible at package test.

- Scheme 2, the TAM for the lowest chip is optimized and that TAM architecture is used for all chips in the 3D Stacked IC. In

this case, all chips use the TAM optimized for wafer sort test of the lowest chip in the 3D Stacked IC.

- Scheme SIC, ILP is used to optimize the test architecture for a given test flow. Our ILP scheme [77] is extended from only accepting WSPT to allow an arbitrary test flow.

The four ITC'02 benchmarks used to construct the core-based 3D Stacked ICs are: d695 (D), g1023 (G), p34392 (P), and t512505 (T), as detailed in Table 6.8. To give an indication of the complexity of the designs, Table 6.8 details the number of cores in the third column. The test time for each design, when all scan chains within the design are concatenated to a single wrapper chain *i.e.* the TAM width is 1, is tabulated in the fourth column. The yield for each design, as assumed for experiments in this paper, are shown in the last column. For example, d695, represented as D, contains 11 cores, requires a test time of 695828 clock cycles and has a yield of 0.65. The two bottom rows indicate the additional test time incurred during each intermediate test, and during the package test respectively. At each intermediate test an additional 10000 time units are added to the total time at the test instance. For instance, the test time assumed during the package test of DGPT is the sum of the test times of each chip − D, G, P and T − three layers of interconnects in between correspond to 3×10000, and finally 10000 for the package itself. The yield at intermediate test is set to 0.65 and the yield at package test is set to 0.75.

Each ITC'02 benchmark in Table 6.8 represents a chip in a 3D Stacked IC. By combining the four benchmarks in various ways, we constructed 3D Stacked ICs with 2, 3 and 4 chips. In total we created 9 designs (DP, DT, GP, GT, DGP, DGT, DPT, GPT, and DGPT) where for example the DP design is a 3D Stacked IC with 2 chips consisting of d695 and p34392 where d695 is the lowest chip. The test time required to test each unit at any test instance is obtained from test architecture design and test planning using either Scheme 1, Scheme 2 or Scheme SIC, detailed in [77]. Note that the test time obtained from test architecture design and test planning of the 3D Stacked IC does not change with yield or the output quantity [77].

In the experiments, for each design we applied the four test flows and at each test flow we used the three test architecture design schemes. The results are collated in Tables 6.9 6.10 6.11, which is organized as follows. There is a sub-table for each of the 9 designs (DP, DT, GP, GT, DGP, DGT, DPT, GPT, and DGPT). Each sub-table is organized in the same manner. As an example, we take design DP in Table 6.9. The

four test flows are listed to the left and for each test flow the test time using the three test architecture schemes are reported. For example, the test time using Scheme SIC with a test flow obtained with TFSA is 5326513. Each test time is compared in two ways. First, at a given test flow the test times are compared against Scheme SIC. For example, the test time obtained with Scheme SIC is 10% lower than that of Scheme 1, where each test architecture scheme is applied on the corresponding test flow obtained with TFSA. Second, for a given test architecture the test times at different test flows are compared. For example, with the SIC Scheme, PT requires 56% higher test time as compared to TFSA.

The results indicate that Scheme SIC is best for all cases. In some cases, Scheme SIC versus Scheme 2 on design GP is 9% better, but in some cases, for example DGPT Scheme SIC is 56% better than Scheme 2, with each test architecture scheme using the test flow obtained by TFSA.

The results indicate that WSPT is as good as TFSA in many cases; however, overall TFSA is close to exhaustive search. In Table 6.12 the test flows with the lowest test times on any test architecture scheme produced by TFSA is compared against exhaustive search on the designs. For example, the test flow for design DP using exhaustive search is $X = (1,1), (0), (1)$, which means wafer sort of D and P and package test of DP (for details, refer to Section 6.1).

6.5. Discussion

Here we summarize Part III of this thesis. Chapter 6 addresses test flow selection for 3D Stacked ICs. The objective is to obtain a test flow that minimizes the total time taken to produce each good package. Given are the testing time and yield values for all test instances. A cost model is presented for calculating the total time taken to produce each good package of a 3D Stacked IC, for any given test flow.

A test flow selection algorithm (TFSA) is proposed for finding a test flow that lowers the test time for a given 3D Stacked IC. The heuristic is executed on several 3D Stacked ICs with up to ten chips in the stack. The test time obtained by TFSA was also compared against three static test flows: (i) tests at all instances, (ii) test only at package test, (iii) wafer sort followed by package test. Results indicated that the test time for performing only package test increases with each additional chip in the stack, as compared to the time taken with the test flow obtained by TFSA. On the contrary, the test time required for testing at all test instances reduces with more chips in the stack, as compared to that of TFSA. However, wafer sort followed by package test provides a test time closest to that of TFSA.

Part IV

Epilogue

This part of the thesis is divided into two sections. A summarization of the thesis along with some concluding remarks are presented in Section 7.1, followed by a discussion of possible future works in Section 7.2.

7

Concluding Remarks

7.1. Summary

In this section we summarize the thesis and present some conclud-
ing remarks. First we discuss about test planning for core-based 3D
Stacked ICs followed by test flow selection.

Test planning for 3D Stacked ICs is addressed in Part II of the thesis.
The goal of a test plan is to minimize the overall test cost, given as
the added cost of test time and the cost related to DfT hardware. Cost
models are presented and methods are proposed to obtain the intended
test plan for three different test architectures *viz.*, BIST, IEEE 1149.1 and
IEEE 1500.

In Chapter 3, the objective is to minimize the test application time
for core-based BISTed 3D Stacked ICs under power constraints.

It is assumed that the BIST engines of several cores share a common
TDR, which is then connected to an on-chip JTAG TAP. To ensure ses-
sion based testing only one TDR from each chip can be accessed at any
time. The test schedule for 3D Stacked ICs includes wafer sort of each
chip, followed by the package test.

The test application time of core-based 3D Stacked ICs can be re-
duced using different approaches. Instead of applying a basic package
test schedule, where the unaltered wafer sort schedules of the chips
are run serially, *aka* Serial Processing, power compatible sessions from
different chips can be executed in parallel, *i.e.*, Partial Overlap. Exper-
iments demonstrate that Partial Overlap could reduce the test time by
up to 12% over Serial Processing.

Further reduction in test application time may be achieved at the
cost of hardware constraints, if the full session-based test schedule,
including the wafer sort and package tests, is optimized globally. With
such an approach, termed as ReScheduling, the test time can be found
to be reduced up to 22% over Serial Processing in the designs assumed
in our experiments.

Rigid ways of test scheduling for non-stacked ICs are not sufficient
to minimize tests for 3D Stacked ICs. Unlike non-stacked ICs, where
the same chip was tested both at wafer sort and package test, in case
of 3D Stacked ICs, different chips or stacks are tested at different test
instances. Test scheduling for 3D Stacked ICs must consider the test
schedules of all test instances simultaneously to minimize test time.

For core based 3D Stacked ICs, with fixed test time and power con-
sumption for each core, tests can be scheduled with varying test appli-
cation times depending on the limitations for the resource constraints.
At the cost of DfT hardware and power consumption the total test time
can be reduced to more than 22% by scheduling core tests simultane-

ously, as demonstrated by the experiments.

The goal of Chapter 4 is to minimize the overall test cost by test planning for scan-based 3D Stacked ICs with an IEEE 1149.1 test architecture. A test cost model is defined as the sum of the cost corresponding to test time and DfT hardware. The test time is given as the sum of the wafer sort and package test times, while the number of TDRs correspond to the DfT hardware.

With increasing number of cores and scan chains, the problem of attaining the optimal trade-off between the components affecting the test cost becomes NP complex. Traditional optimization methods, such as Simulated Annealing, require longer times to obtain the desired test plan with increasing number of scan-chains in the 3D Stacked IC. Where Simulated Annealing may take days to arrive at the optimal test plan, the proposed algorithm provides a near-optimal solution within seconds. Implementations on various ITC'02 benchmark designs suggest that the cost corresponding to the proposed test plan, as compared to Simulated Annealing, was 6% higher in the worst case, and 2% higher at an average. However, Simulated Annealing may require more than 10^3 times the computation time required to arrive at the desired test plan.

Similar to the previous section, the objective in Chapter 5 is to minimize the test cost for core-based 3D Stacked ICs supported by a IEEE 1500 based test architecture.

Here the test cost is given as the sum of the cost related to test time and DfT hardware. The test time is calculated as the total time taken to execute all wafer sort schedules and the package test schedule. The width of the TAM determines the hardware cost. A mathematical model to calculate the test cost is presented.

Several schemes may be explored to reduce the test cost for IEEE 1500 based 3D Stacked ICs, which is known to be NP-hard problem. We propose an ILP based approach, that takes into account both wafer sort and package test instances simultaneously to optimize the test cost. In contrast, multiple straightforward schemes may be adapted, that require lower computation times. For the first straightforward scheme, the test plan for 3D Stacked ICs is obtained by optimizing the test plan for each individual chip in the stack. This minimizes the test cost during wafer sort, but a sub-optimal package test cost may lead to a higher overall test cost. Experiments on 3D Stacked ICs constructed from ITC'02 benchmarks indicate a 40% reduction in the total test cost at an average for test plans obtained with ILP against this straightforward scheme. For the second approach, the TAM partition

that is optimal for the lowermost chip can be applied to all chips in the stack. Due to different TAM partitioning in each chip, it is necessary that the test schedules of the chips are run serially during package test. This considerably increases the package test time, thereby providing a sub-optimal test schedule. In this case, at an average, ILP provides a reduction of 63% of the test cost when compared against this test planning scheme. Thus, optimizing the test cost, taking into account all test instances could result in considerable reduction in the test cost. However, the computation time for ILP may be extremely long, so here we reduce the search space by determining a near-optimal TAM width through a bounding scheme.

Part II discusses test planning for 3D Stacked ICs for three test architectures, *viz.*, BIST, IEEE 1149.1, IEEE 1500. Approaches have been proposed to minimize the test cost by co-optimizing the test time and DfT hardware. As compared to traditional approaches used for non-stacked ICs, the proposed approaches in all cases show considerable reduction in test costs. As compared to standard optimization methods, the proposed approaches require considerably less computation time and provide test plans with costs close to the optimal test plan. We have assumed the traditional test flow, applied for non-stacked ICs, comprising of wafer sort of each chip followed by package test. However, the approaches can be suitably modified for any given test flow. Although we assume test architectures traditionally used for non-stacked ICs, test plans with even lower test costs may be obtained with other test architectures.

Chapter 6 of Part III addresses test flow selection for 3D Stacked ICs. The objective is to minimize the time required to produce each good package, given the test time and yield of each test instance.

Taking into account the yield at each instance, if tests are performed during fewer test instances then more units must be tested at the instances when tests are performed. The trade-off between the time spent during a test instance to the additional time spent on other test instances on testing the added faulty units contributes to the final test time spent on each good package. The search space for the optimal test flow increases exponentially with the number of cores and chips in the stack. Therefore, generic test flows may be employed for the testing of 3D Stacked ICs, such as wafer sort followed by package test (WSPT), testing at all instances (TA), or only package test (PT). A near-optimal test flow may be obtained with significantly lower computation time for optimizing the test flow with the TFSA. Experiments demonstrate that the proposed TFSA can reduce the test time by more than 61% over TA for 3D Stacks with fewer chips, though the test time overhead

reduces wwith a larger number of chips in the stack. On the other hand, PT may prove to be the optimal test flow with up to 3 chips in the stack, but for larger number of chips TFSA can again provide a lower test time by up to 270% at an average. WSPT however performs best among predefined test flows, with a 21% higher test time than TFSA at an average. Subsequently, TFSA generates test flows and the corresponding test times very close to exhaustive search, with an average 2% higher test time, but with up to about a 100 times lower computation time.

Overall, in this thesis we discuss test cost reduction of 3D Stacked ICs with two different approaches: test planning and test flow selection. Test time and DfT hardware is co-optimized to achieve a test plan that minimizes the total test cost. We provide test planning approaches for core-based 3D Stacked ICs based on different test architectures. In addition, test time required to produce each good packaged chip is minimized using the TFSA. Experiments show considerable reductions in test cost.

7.2. Future Work

Reduction of test cost has proven to be a persistent challenge for the large scale manufacture of ICs. With new emerging technologies like 3D integration, solutions developed for previous generations of integration technologies has drawbacks. 3D Stacked ICs require an entirely different manufacturing process with several additional steps. Each additional step introduces the possibility of new and previously unseen defects. Hence, 3D Stacked ICs needs a different approach to the problem of test cost optimization as compared to its predecessors.

This thesis focuses on test cost reduction of 3D Stacked ICs by co-optimizing test time and DfT hardware and the selection of an efficient test flow. Either problem has a large solution space, for which methods to arrive at the exact optimal solutions are yet to be devised. Therefore, the work in this matter is far from over.

Test cost based on test time, DfT hardware or test flow of 3D Stacked ICs can be further reduced by considering additional factors.

Possible issues are hereby discussed.

Test Planning: With IEEE P1838 and test architectures used for non-stacked ICs

Test architecture standard for 3D Stacked ICs, namely the IEEE P1838,

is being developed to obtain test access between different layers within the stack [70]. The IEEE P1838 standard is based on the existing IEEE 1149.1 standard and IEEE 1500 standard. The standard facilitates TSV test and circuit test after die bonding. [71] [72] explores the emerging IEEE P1838 standard with regard to SICs and test interfaces and corresponding change of test perspective as compared to traditional ICs and wafer probing of TSVs and the testing of die logic before and after TSV breakpoints, respectively. In addition, several articles acknowledge the developing IEEE P1838 standard for test of SICs. However, due to lack of finalization utilization of the standard in [28] [73–76] is limited. In [73], a new Memory Based Interconnect Test (MBIT) approach for SIC memories is proposed, that has no area overhead, detects both static and dynamic faults and performs at speed testing, has flexibility in applying any test pattern and extremely short test execution time. [28] [76] overviews and discusses the challenges and emerging solutions in testing three classes of memories: 3D stacked memories, Resistive memories and Spin-Transfer-Torque Magnetic memories, followed by corresponding defect mechanisms and fault models. Special applications of IEEE P1838 are proposed in [74] [75]. [74] proposes an efficient multicast test architecture for targeting defects in dies, where multiple chips can be tested simultaneously to reduce the test-application time under power and fault coverage constraints followed by test scheduling and optimization using the proposed architecture, while [75] proposes an architecture for an FPGA-based tester for a SICs. The standard involves a die level wrapper on each chip in the stack. In addition, an IEEE 1149.1 based TAP controller is provided in the bottom die that controls the WIRs of the die wrappers. [77] [78] addresses test scheduling for SICs to minimize the test cost. In [77] a scalable test architecture is assumed where each chip is provided with a IEEE 1500 based wrapper, in accordance with the developing IEEE P1838 standard. Similarly, [78] assumes a IEEE 1149.1 based test architecture for the SICs for test planning.

The developing test architecture standard for 3D Stacked ICs, IEEE P1838, may be employed for test planning to further reduce test cost. IEEE P1838 focuses on standardizing the inter-chip test access, *i.e.*, for TSVs between chips. However, the standard overlooks test architecture definition for intra-chip logic. In other words, the infrastructure to test the cores within a chip is not defined by IEEE P1838. The standard however provides a IEEE 1149.1 compliant TAP and a FPP. The widely employed test architecture standard for non-stacked ICs, IEEE 1500, does not consider test of TSVs. IEEE 1500, however, also complies with the IEEE 1149.1 TAP. In addition, the standard also allows a WPP

analogous to the FPP of IEEE P1838. Thus, by integrating both IEEE P1838 and IEEE 1500 standards, a standardized test architecture spanning both inter-chip and intra-chip circuitry for 3D Stacked ICs can be obtained. Test scheduling, considering such an architecture may lead to further reductions in test cost for 3D Stacked ICs.

For 3D Stacked ICs with TSVs, Marinissen *et. al.* in [52], proposed a scalable test architecture based on IEEE 1149.1 and IEEE 1500 connected on wide TAMs, with the assumption that each chip is equipped with dedicated probe pads for wafer sort. In addition, all signals from a chip are transferred to the chip on top of it via TSVs, which are situated on the top side of the chip, and hierarchical wrapper instruction register (WIR) chains are used to prevent unbridled growth of length. In [52], Marinissen *et al.* have proposed a test access architecture for 3D Stacked ICs which is based on [40]. The test architecture is based on IEEE 1149.1, and supports both wafer sort and package testing, using a modular approach. Marinissen *et. al.* in [52] highlight the importance of standardization and optimization of test architecture of 3D Stacked ICs, which is addressed in this thesis.

Test Flow Selection: Test schedule dependent test flow selection

The optimal test cost of a given 3D Stacked IC can be obtained by comparing the optimal test plans for each possible test flow. It is seen that the test time and yield of each component constituting the 3D Stacked IC are decisive factors in determining a cost-efficient test flow. It has also been illustrated that the test time at each test instance varies greatly with the test schedule, *i.e.*, the test flow varies with the chosen test schedule. On the other hand, to determine a test schedule for reducing the total test time should consider all enabled test instances simultaneously, *i.e.*, the optimal test schedule depends on the chosen test flow. Thus there exists a dilemma of the causality due to the interdependency of the selected test flow and test schedule. Hence, test planning algorithms should include test flow selection methods in order to obtain the minimal test cost.

A possible approach may include a predetermined test flow as the starting point to arrive at a cost-efficient test schedule. The process can be iterated with a range of test schedules against different test flows to arrive at a integrated test plan with reduced test cost.

Part V

Appendix

Appendix A

Test Mechanism

Here we describe the test mechanisms of the different test architectures used in the thesis. First we start with BIST in Section A.1, followed by IEEE 1149.1 (Boundary Scan) in Section A.2. Finally, in Section A.3, the IEEE 1500 test architecture is discussed.

A.1. Built-In Self-Test (BIST)

A generic test architecture of a BISTed device is presented in Figure A.1 [2] [20–23] [29] [30] [55] [107–109]. The test pattern generator provides the required patterns for testing the DUT using either (i) exhaustive, (ii) pseudo-exhaustive, or (iii) pseudo-random patterns. The patterns are shifted in using the wrapper, and the response is relayed to the output response analyzer. The output analyzer compares the signature obtained from the device under test (DUT) output to the expected signature. The comparator uses one among the several techniques (i) ones count, (ii) transition count, (iii) parity checking, (iv) syndrome checking, or (v) signature analysis. The test controller, illustrated by the large block at the left in Figure A.1, communicates the clock, enable and other signals to the different blocks of the architecture.
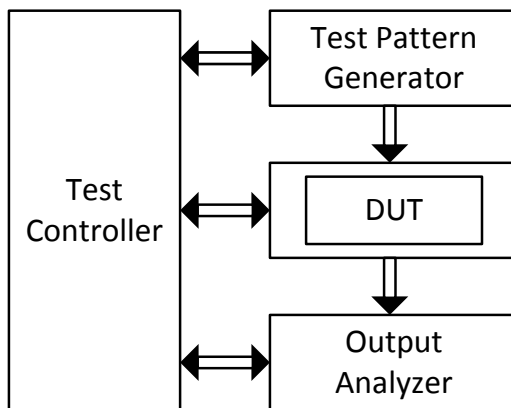

Figure A.1.: Test architecture of BIST

A.2. IEEE 1149.1

In this section we describe the IEEE 1149.1 standard. First we present the general test architecture, and in detail, describe the main components of the IEEE 1149.1 standard along with the instruction set used.

This standard was developed for testing ICs and interconnects between ICs on a PCB. It is achieved by inserting a boundary-scan cell at every I/O pin if the ICs and connecting them to form a boundary-scan register. Multiple ICs on a PCB may have daisychained registers to form a scan-chain.
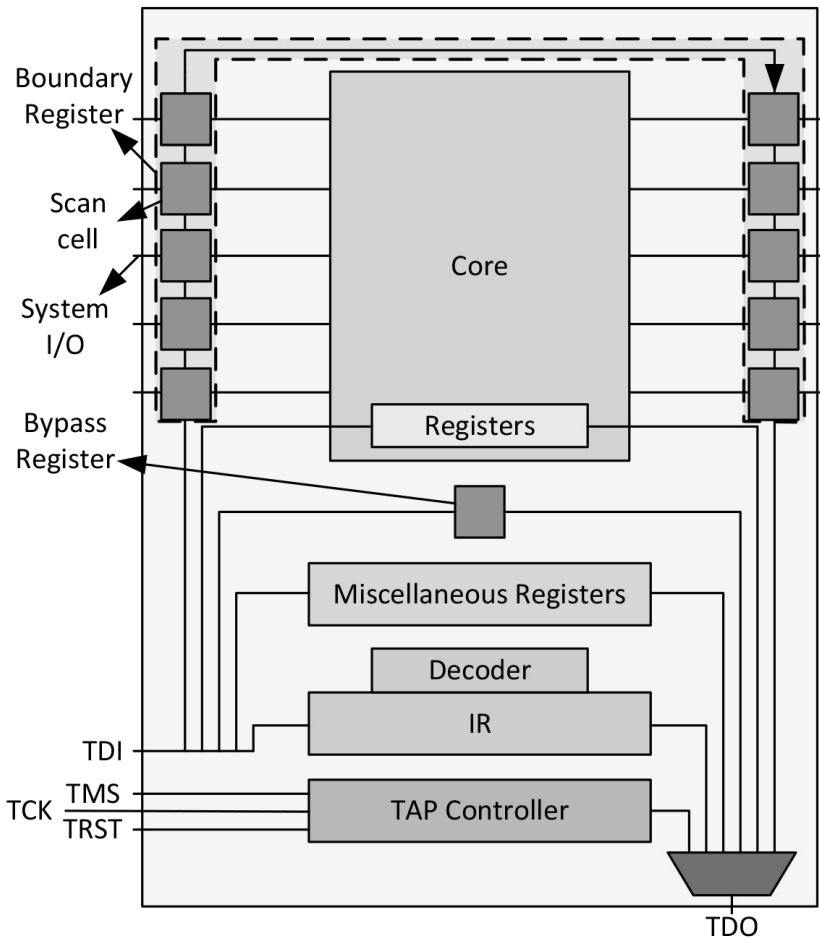
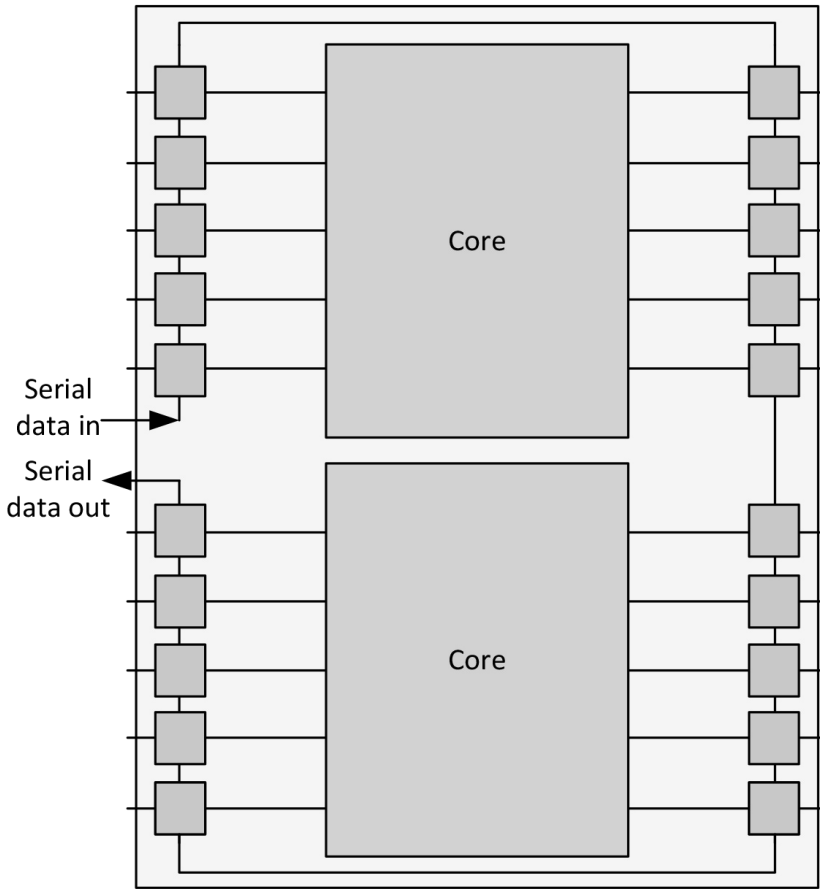**Figure A.2.:** Test mechanism of IEEE 1149.1

Figure A.3.: Serial access of cores using IEEE 1149.1

The IEEE 1149.1 standard is illustrated in Figure A.2 [2] [21–23] [31–33] [56–59]. The main components of the standard are:

- Test Access Port (TAP)

- TAP controller

- Instruction Register (IR)

- Test Data Registers (TDRs)

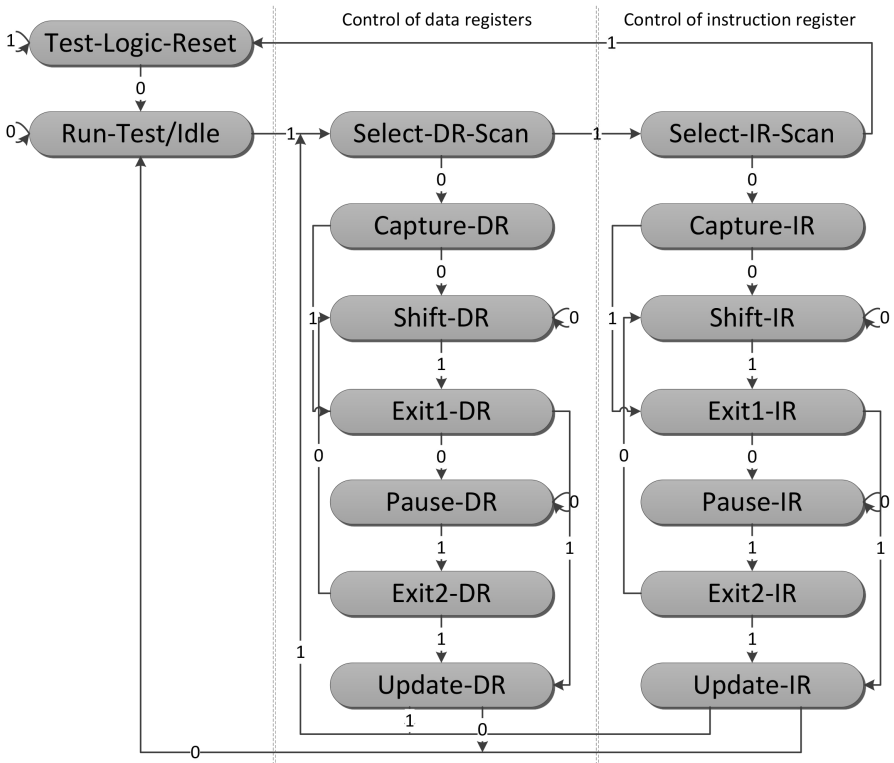The standard functions as follows.

Figure A.4.: State diagram of the IEEE 1149.1 TAP controller

- To generate control signals for configuring the circuit under test, an instruction is sent in via the TDI to the IR

- Each test pattern is shifted in via the TDI to the TDRs and applied to the circuit under test. The response is captured in a TDR and shifted out via the TDO while a new test pattern is shifted in

Next we discuss the TAP and TAP controller, followed by the IR and TDRs. The TAP controller acts as the interface for the standard, and uses the following signals:

- Test Data In (TDI) serially shifts in the test data or instructions to the core at the rising edge of TCK.

- Test Data Out (TDO) shifts out the test response from the core at the falling edge of TCK.

- Test Mode Select (TMS) is used to determine the next state at the rising edge of TCK. The pin switches the system from functional to test mode.

- Test Clock (TCK) is used to synchronize operations of the internal state machine. The system clock and TCK rates may not necessarily be the same.

- Test Reset (TRST) is a optional pin used to reset the state machine of the TAP controller.

The TAP controller, illustrated by the lowermost block in Figure A.2, may operate at any of the following stages illustrated in Figure A.4:

- Test-Logic-Reset is the normal mode of operation.

- Run-Test/Idle waits for internal tests to be performed.

- Select-DR-Scan initiates data-scan sequence.

- Capture-DR enables loading of test data.

- Shift-DR loads test data in series.

- Exit1-DR ends phase-1 shifting of data.

- Pause-DR, as bus master reloads data, the mode temporarily holds the scan operation.

- Exit2-DR ends phase-2 shifting of data.

- Update-DR loads from corresponding shift registers in parallel.

The instruction is stored in the IR. There are four obligatory instructions:

- BYPASS relays the signal from TDI to TDO thus enabling other components in the series to be tested Figure A.3

- EXTEST chooses the boundary register and the input is sampled using Capture-DR. Values are shifted in during Shift-DR, followed by Update-DR

- SAMPLE/PRELOAD uses the boundary register while the component is in the functional mode. Data scanning can be used to access the boundary registers to observe incoming or outgoing data from the DUT

Additionally, several other instructions may be loaded to the IR, such as RUNBIST, USRCODE, HIGHZ, IDCODE, CLAMP.

IEEE 1149.1 uses at least two kinds of TDRs including the boundary register, shown with dotted lined in Figure A.2, and the single bit bypass register, that directly relays signal from TDI to TDO with minimal overhead. The device ID, is included among the miscellaneous registers.

The boundary scan description language (BSDL) is used by the test generator to develop the test for the DUT.

A.3. IEEE 1500

In this section we describe the IEEE 1500 test architecture standard, as illustrated in Figure A.5 [2] [22] [23] [35] [62] [63] [65] [66]. In the IEEE 1500 standard each core is provided with a wrapper, that provides a common platform for the test commands and to perform the test. A TAM enables propagation and reception of test data.

A core wrapper compliant to the IEEE 1500 standard is illustrated in Figure A.5. The IEEE 1500 standard wrapper can be categorized into the following five broad segments:

1. **Wrapper instruction register (WIR)**: The WIR is used for storing instructions that are to be executed.
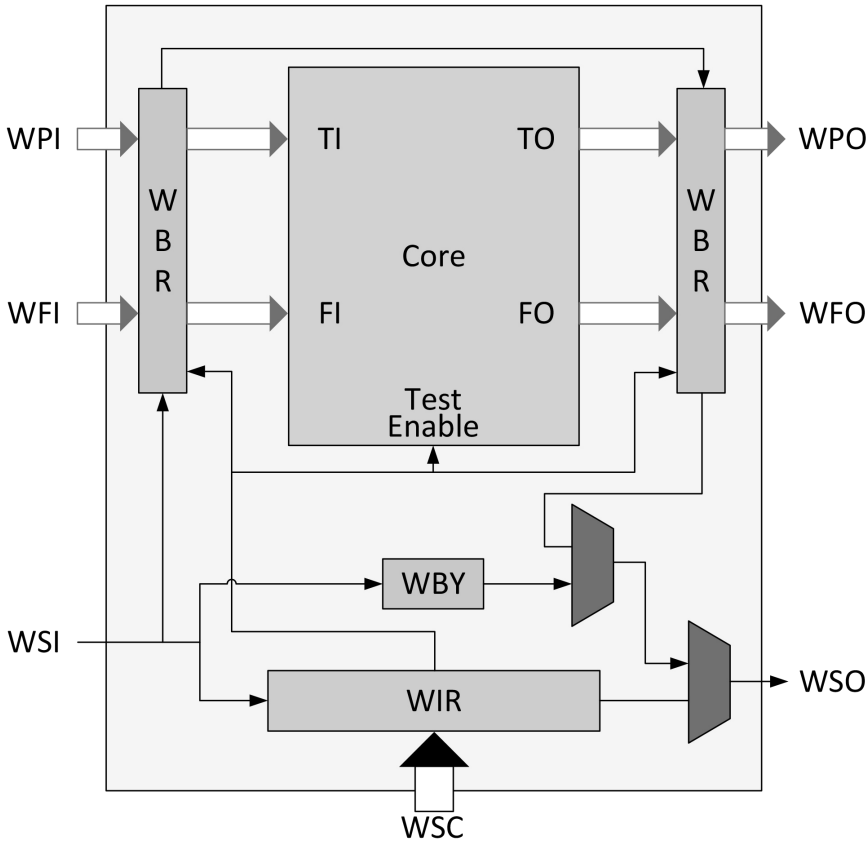

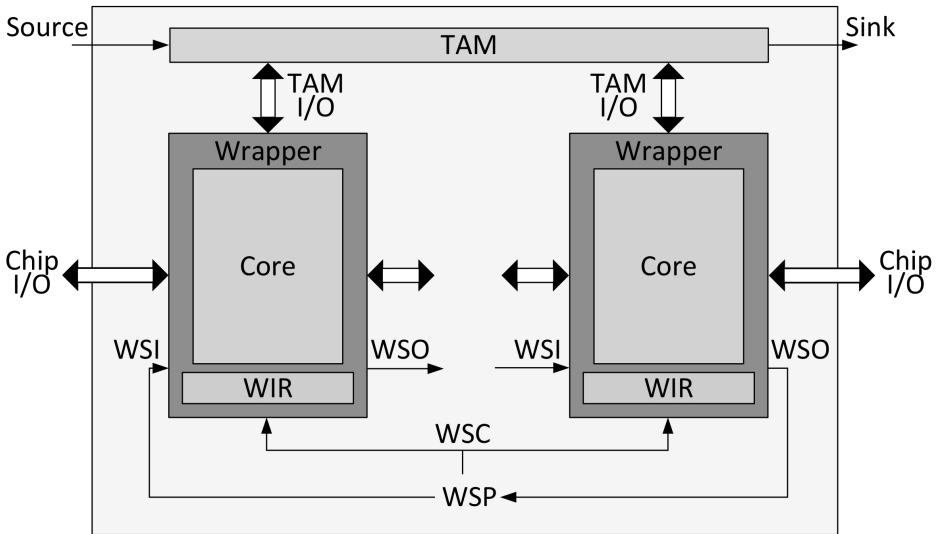**Figure A.5.:** The IEEE 1500 standard test wrapper for a core

Figure A.6.: Test access of cores provided with the IEEE
1500 wrapper

2. **Wrapper boundary register (WBR)**: The WBR is composed of
   multiple wrapper boundary cells (WBCs) connected serially.

   Each WBC comprises of four data terminals, *viz*, cell functional
   input (CFI), cell functional output (CFO), cell test input (CTI),
   and cell test output (CTO).

3. **Wrapper bypass register (WBY)**: The WBY directly connects the
   WSI and the WSO, and is used to bypass test signals. WBY gets
   selected by default when no other register is specifically selected
   by the WIR.

4. **Wrapper serial port (WSP)**: The WSP comprises of three sets of
   terminals, *viz*, a wrapper serial input (WSI), a wrapper serial out-
   put (WSO), and multiple wrapper serial controls (WSCs).

   The WSC in turn may comprise up to seven terminals for control,
   viz, wrapper reset WRSTN, wrapper clock WRCK, SelectWIR,
   CaptureWR, ShiftWR, UpdateWR, and a optional TransferDR.
   WSC may also contain multiple AUXCKn clock terminals.

5. **Wrapper parallel port (WPP)**: The WPP is an optional compo-
   nent of the IEEE 1500 standard. It consists of three categories of

terminals that can be configured by the manufacturer, *viz*, wrapper parallel input (WPI), wrapper parallel output (WPO), and wrapper parallel control (WPC).

The DfT of a chip provided with the IEEE 1500 test architecture is shown in Figure A.6. The chip contains several cores, each provided with the IEEE 1500 wrapper and a WIR. The WIR stores the instruction to be executed for the respective cores. A mandatory WSP is present at the chip, that connects each core with the WSI, the WSO, and multiple WSC terminals. Although the control signals for each core can be applied directly to the WSP if the IEEE 1500 standard, the interface is also compatible to with the TAP controller of the IEEE 1149.1 standard.

A major advantage of the IEEE 1500 standard is the support of parallel test. This is done by a optional test access mechanism (TAM), as shown in Figure A.6, that is user defined. Multiple TAM input and output terminals connect each core to the TAM source and sink respectively, to communicate parallel test information. The TAM inputs correspond to WPC and WPI, while the TAM outputs to the WPO in Figure A.5. The provision of parallel TAM is mainly responsible for the reduction of test time.

The WBR may operate in any of the following functional modes defined by the IEEE 1500 standard:

- *Normal*: In the normal mode, the core performs its usual functions.

- *Inward facing*: As the name suggests, circuitry internal to the core is tested during the inward facing mode. The WBR controls and observes the WFI and WFO terminals during this mode of operation, respectively.

- *Outward facing*: As the name suggests, interconnects and other circuitry outside to the core is tested during the outward facing mode. The WBR controls the WFO and WPO terminals and observes the WFI and WPI terminals during this mode of operation.

- *Nonhazardous* (safe): In the nonhazardous operational mode, the WBR controls the FI and WFO in a safe state.

Each WBC may operate in any of the following events:

- *Shift*: During the shift event the data in the WBCs, forming the WBR, that connects the WSI to the WSO shifts by one WBC towards the WSO. Meanwhile, the first WBC, of the WBR, at the WSO is loaded with new data. *This is a mandatory event.*

- *Capture*: During the capture event the data on the CFI or CFO of a WBC is captured and stored in the WBC. *This is a mandatory event.*

- *Update*: At the update event, data stored in the shift path storage element of a WBC, that lies closest to the CTO, is loaded to another storage element of the WBC that lies away from the shift path. *This is a optional event.*

- *Transfer*: The transfer event might perform one of the two tasks: (i) if the data during the capture event is not present in a storage element, it is moved near the STI of the WBC. This maximizes the captured data (ii) data is transferred nearer to the CTO by one step. This facilitates delay testing *This is a optional event.*

- *Apply*: The apply event activates test data to make it effective as stimuli. During the inward facing or the outward facing modes, this event controls data at the corresponding terminals. The Apply event is a virtual event inferred from other events. *This is a derivative event.*

All devices compliant to the IEEE 1500 standard includes a defined instruction set as described below:

- **W<S/P/H>_<Command>_<Configuration>**: In this instruction, *W* is the preface for all IEEE 1500 instructions. *S*, *P*, and *H* correspond to serial, parallel, and hybrid test modes, respectively.

  For serial *S* instructions, only the elements of WSP, *i.e.*, WSI, WSO and WSC terminals are utilized, while the WPP is unaffected.

  The parallel *P* instructions would target the WPP elements, *i.e.*, WPI, WPO, and WPC terminals. While parallel instructions are executed, only WBY is considered between the WSI and WSO terminals.

  The hybrid *H* instructions utilizes both WSP and WPP simultaneously.

  Command stands for the operations to be executed for the instruction (e.g., EXTEST or BYPASS). Configuration describes the test configuration selected by the instruction.

  Examples include: WS_INTEST_RING, WS_INTEST_SCAN

  These are discussed below in further detail:

- **WS_INTEST_RING**: During the execution of this instruction, the core is in the inward facing mode and performs one step of the operation as data is shifted into the WBR each time. This is a optional instruction among IEEE 1500 instruction set.

- **WS_INTEST_SCAN**: Similar to WS_INTEST_RING, this instruction enables core operation while the scan chains of the core are serially connected. This is a optional instruction among IEEE 1500 instruction set.

- **W<S/P/H>_EXTEST**: For S/P, this instruction is the same as the WS_EXTEST or WP_EXTEST instructions. For WH_EXTEST, the instruction becomes hybrid, and may execute EXTEST with a mixed series and parallel configuration.

- **W<S/P/H>_INTEST**: At least one INTEST instruction is mandatory in the standard IEEE 1500 instruction set.

  Subcategories include: WS_INTEST_RING,

  WS_INTEST_SCAN, W<S/P/H>_TEST

- **WS_BYPASS**: The bypass instruction transfers all information directly from the WSI to the WSO terminal, thus letting the core run in its normal functional mode. This instruction is mandatory in the standard IEEE 1500 instruction set.

- **WS_EXTEST**: The serial extest instruction is used to enable the outward facing mode, and thus interconnects and other circuitry outside to the core is selected to be tested. As the instruction targets only the serial ports, it concatenates all external circuitry serially into a single scan chain. This instruction is mandatory in the standard IEEE 1500 instruction set.

- **WP_EXTEST**: The parallel extest instruction is used to enable test of circuitry outside the core. Unlike WS_EXTEST, the external circuitry forms multiple single scan chains for WP_EXTEST, corresponding to the width of the WPI/O. This is a optional instruction among IEEE 1500 instruction set as it only involves the optional parallel terminals.

- **WS_SAFE**: This instruction lets the wrapper in a safe state by setting the CTOs at predetermined values. This is a optional instruction among IEEE 1500 instruction set.

- **WS_PRELOAD**: With the serial preload instruction, the WBRs can be loaded without changing the data on the terminals of the external circuitry. The instruction may also substitute WS_SAFE, or used to load data before executing the WS_EXTEST instruction. Therefore, this instruction is mandatory in the standard IEEE 1500 instruction set if the WBC form a shift path keeping the WFO at a constant.

- **WP_PRELOAD**: Similar to the WS_PRELOAD nstruction, this instruction can be used to load data to the WBR. However, in case of parallel preload, the WBR can be visualized as several independent parts, thus making the test more efficient. This is a optional instruction among IEEE 1500 instruction set as a more efficient way of executing WS_PRELOAD.

- **WS_CLAMP**: This instruction enables the WBR to drive WFO, while the core is put to reset or clock off. This is a optional instruction among IEEE 1500 instruction set.

Appendix

B

3D Stacked IC Construction

Here we describe the 3D Stacked IC designs used to implement the
test planning and test flow algorithms proposed in this thesis. Section
B.1 details all the ITC'02 benchmarks used, while in Section B.2 we
describe the 3D Stacked ICs.

Table B.1.: Non-stacked benchmark circuits
used in Chapter 3

Label	SoC	Cores	Time, Power
Z	*ASIC Z*	9	{160, 352}, {134, 295}, {102, 279}, {102, 279}, {69, 282}, {61, 241}, {38, 213}, { 23, 96}, {10, 95 }
M	*Muresan*	10	{675, 675}, {600, 300}, {600, 75}, {450, 450}, {375, 375}, {300, 150}, {225, 75}, {150, 300}, {75, 900}, {75, 525}
L	*System L*	14	{387, 285}, {120, 54}, {83, 18}, {46, 43}, {29, 21}, {22, 90}, {5, 10}, {3, 7}, {3, 7}, {164, 4}, {174, 285}, {31, 38}, {54, 154}, {78, 30}

B.1. Benchmarks

Several benchmark SoCs have been used to achieve 3D Stacked IC designs. Experiments are performed for test planning approaches proposed in Part II. The rest of this section details the data used for test planning with a BISTed architecture (B.1.1), followed by IEEE 1149.1 based architecture (B.1.2) and finally IEEE 1500 based test architecture (B.1.3).

B.1.1. BIST

The benchmarks considered in Chapter 3 are: ASIC Z [22][29], System L [22] and Muresan's design [27]. The assumed data is tabulated in Table B.1. The power constraint assumed in all cases is $W_{max} = 900$ units.

B.1.2. IEEE 1149.1

The ITC'02 benchmark SoCs used in Chapter 4 are: p22810, p93791, g1023, d695, h953 and d281 [68][110]. The assumed data is tabulated in Table B.2. The length of each scan chain and the corresponding number of patters required by each module are listed in the last column serially as per the modules. The power constraint assumed in all cases is $W_{max} = 500$ units.

For each benchmark, the inputs, outputs and bidirectionals are ig-

Table B.2.: ITC'02 SoC benchmark circuits
used in Chapter 4

Label	SoC	Cores	{Scan-chain length, Patterns}
1	*p22810*	22	{130, 785}, {68, 644}, {186, 465}, {181, 215}, {214, 202}, {400, 181}, {122, 175}, {77, 124}, {73, 108}, {88, 94}, {82, 93}, {77, 59}, {43, 58}, {115, 40}, {99, 38}, {80, 37}, {101, 27}, {100, 26}, {89, 25}, {109, 8}, {34, 2}, {104, 1}
2	*p93791*	13	{68, 916}, {181, 416}, {168, 409}, {93, 391}, {175, 234}, {521, 218}, {150, 216}, {100, 210}, {219, 194}, {219, 194}, {82, 187}, {189, 172}, {5, 11}
3	*g1023*	12	{54, 268}, {43, 134}, {84, 74}, {64, 68}, {53, 57}, {32, 51}, {47, 36}, {47, 34}, {52, 31}, {13, 29}, {13, 18}, {9, 15}
4	*d695*	8	{41, 234}, {45, 110}, {54, 105}, {46, 97}, {34, 95}, {12, 75}, {55, 68}, {54, 12}
5	*h953*	7	{348, 341}, {189, 305}, {185, 182}, {121, 110}, {21, 49}, {32, 39}, {327, 9}
6	*d281*	5	{32, 2048}, {9, 1082}, {32, 374}, {8, 346}, {9, 336}

nored. Additionally, modules that have no scan-chains are not counted. Only the longest scan-chain is used in each module for test planning. This helps avoid any drastic differences in the test time of each core, calculated by Eq. 4.2, in order to ensure unbiased test planning.

B.1.3. IEEE 1500

ITC'02 benchmark SoCs have been considered in Chapter 5, that include: d695, g1023, p34392 and t512505 [68][110]. Each module defined in the benchmarks are serially listed as cores for the experiments performed. The total number of scan flip-flops (including inputs and outputs) are listed along with the number of patterns required by each core in the last column. The assumed data is tabulated in Table B.3.

In each benchmark considered for test planning, the length is calculated as the sum of the inputs, outputs, bidirectionals and all scan-chains within the module. Modules that have zero total tests are not

taken into account for the purpose of the experiments.

Table B.3.: ITC'02 SoC benchmark circuits
used in Chapter 5

Label	SoC	Cores	{Scan-chain length, Patterns}
D	*d695*	10	{64, 12}, {315, 73}, {67, 75}, {286, 105}, {1768, 110}, {852, 234}, {122, 95}, {77, 97}, {73, 12}, {82, 68}
G	*g1023*	14	{130, 134}, {68, 74}, {186, 57}, {181, 268}, {214, 51}, {400, 36}, {82, 34}, {77, 31}, {43, 68}, {99, 29}, {80, 15}, {101, 16}, {89, 512}, {104, 1024}
P	*p34392*	19	{130, 210}, {68, 514}, {186, 3108}, {181, 6180}, {214, 12336}, {400, 1965}, {122, 512}, {77, 9930}, {88, 228}, {82, 454}, {77, 9285}, {115, 173}, {99, 2560}, {80, 432}, {100, 4440}, {99, 128}, {80, 786}, {100, 745}, {89, 12336}
T	*t512505*	31	{746, 157}, {392, 330}, {1003, 154}, {946, 408}, {25, 3}, {204, 127}, {672, 608}, {4671, 1025}, {734, 195}, {8304, 788}, {637, 188}, {173, 42}, {187, 68}, {2357, 278}, {924, 151}, {1901, 370}, {568, 80}, {123, 153}, {118, 79}, {118, 77}, {582, 242}, {582, 233}, {2966, 532}, {1980, 429}, {678, 148}, {216, 13}, {99, 10}, {7, 3}, {307, 67}, {410, 151}, {43922, 3370}

B.2. 3D Stacked IC Design Formation

In this section we illustrate how we construct and test 3D Stacked ICs for the experiments presented in this thesis. We assume that 3D Stacked ICs are formed by stacking multiple SoCs from the ITC'02 test benchmarks provided. The formation of 3D Stacked ICs used in this thesis is explained with the help of an example.

3D Stacked ICs can be bonded face-to-face, back-to-back and face-to-back [6]. Face-to-face and back-to-back bondings are not scalable to more than two chips in the stack. Therefore, we assume face-to-back bonding for experiments performed in this thesis for 3D Stacked ICs with any number of chips in the stack. Chips in the stack may be bonded face-to-back to the I/O pins in the package either face-up (Figure B.1) or face-down (Figure B.2).

Figure B.1 shows a 3D Stacked IC with N chips in the stack, bonded face-to-back with face-up. The chips are stacked serially with Chip 1 as the bottom chip and Chip N as the topmost chip.

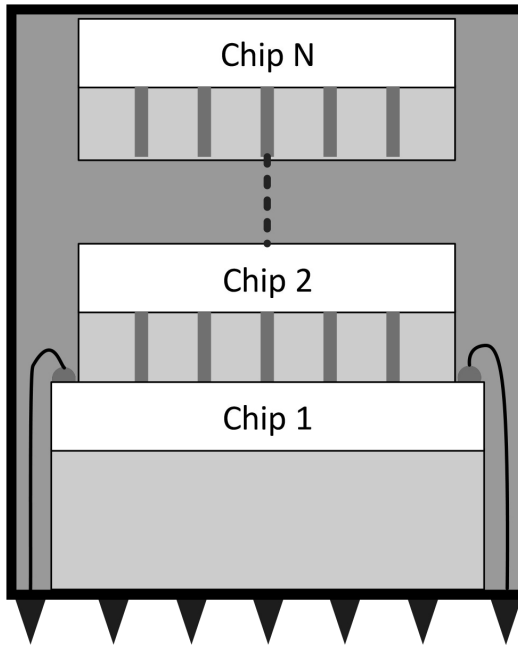A stack of three chips: Chip 1 $\equiv X$, Chip 2 $\equiv Y$ and Chip 3 $\equiv Z$

Figure B.1.: 3D Stacked IC constituting N chips, with face-to-back face-up bonding

respectively, would be called XYZ by the system of nomenclature used in this thesis.

In this case, X, Y and Z are the labels assigned to the corresponding benchmarks.

For testing purposes, each individual chip behaves as a non-stacked SoC during wafer sort. During intermediate and package tests the entire stack under test behaves as an individual unit. The accumulation of all cores within the stack may be scheduled in a manner similar to scheduling of cores within a non-stacked IC. The TSV interconnects between the chips can be also considered as cores with the additional constraint on scheduling that TSVs may not be tested simultaneously
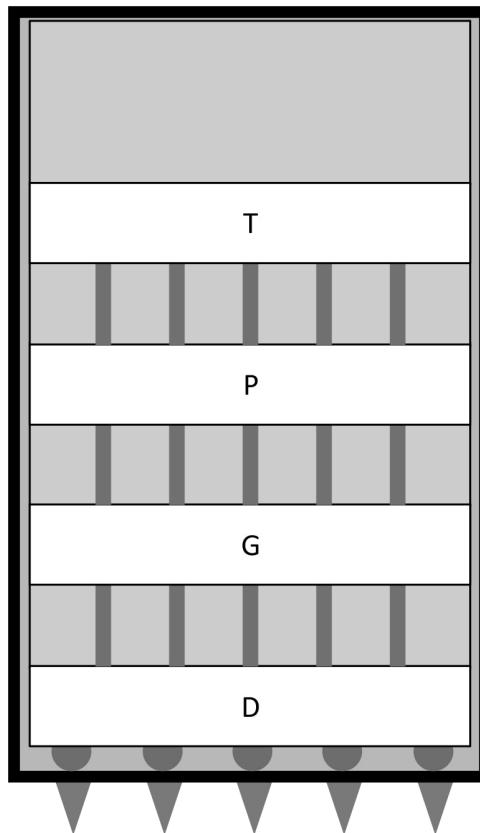

Figure B.2.: 3D Stacked IC constituting D (d695), G (g1023), P (p22810) and T (t512505) stacked face-down

with cores within the chip.

For example, the design DGPT used in Section 5.5 of Chapter 5, refers to a 3D Stacked IC formed by stacking four chips d695, g1023, p22810 and t512505, with the corresponding labels D, G, P and T respectively as seen in Table 6.8. Furthermore, the order of the chips in the stack from bottom to top is D, G, P and T. The design is illustrated in Figure B.2 with a face-down face-to-back bonding.

During wafer sort, D, G, P and T are tested individually akin to non-stacked ICs. For intermediate or package tests, DG, DGP and DGPT are considered as a unified unit. Thus, for the intermediate test of DGP, tests of all cores in D, G and P can be scheduled simultaneously as long as power, resource or other constraints are not violated. TSVs between D and G, as well as G and P may be tested separate from the cores within either D, G, P or T.

References

[1] Nobelprize.org, "The History of the Integrated Circuit," in *Nobel Media AB 2014.*, 2015.

[2] L.-T. Wang, C.-W. Wu, C.-W. Wu, and X. Wen, "VLSI test principles and architectures: Design for Testability," in *Academic Press*, 2006.

[3] G. Moore, "Cramming More Components Onto Integrated Circuits," in *Electronics*, pp. 671–680, 1965.

[4] A. Brás, "Systems on chip: Evolutionary and revolutionary trends," in *Internal Conference on Computer Architecture (ICCA)*, 2002.

[5] V. Malhi, "Pop, sip, mcm, mcp or soc: Assessing the tradeoffs," *EE Times-India*, pp. 1–2, 2006.

[6] E. J. Marinissen and Y. Zorian, "Testing 3D Chips Containing Through-Silicon Vias," in *IEEE International Test Conference (ITC)*, pp. 1–11, 2009.

[7] H.-H. S. Lee and K. Chakrabarty, "Test Challenges for 3D Integrated Circuits," in *IEEE Design and Test of Computers, SpecialIssue on 3D IC Design and Test*, pp. 26–35, Oct. 2009.

[8] L. Patrick, F. Crécy, M. Fayolle, B. Charlet, T. Enot, M. Zussy, B. Jones, J.-C. Barbé, N. Kernevez, N. Sillon, S. Maitrejean, D. Louis, and G. Passemard, "Challenges for 3D IC integration: bonding quality and thermal management," in *IEEE Design and Test of Computers, Special Issue on 3D IC Design and Test*, pp. 210–212, 2007.

[9] J. H. Lau, "Evolution, challenge, and outlook of TSV, 3D IC integration and 3D silicon integration," in *International Symposium on Advanced Packaging Materials (APM)*, pp. 462–488, 2011.

[10] R. S. Patti, "Three-Dimensional Integrated Circuits and the Future of System-on-Chip Designs," in *Proceedings of the IEEE*, vol. 94, pp. 1214–1224, June 2006.

[11] P. Emma and E. Kursun, "Opportunities and challenges for 3D systems and their design," in *Design Test, IEEE*, vol. PP, pp. 1–11, 2013.

[12] K. Bernstein, P. Andry, J. Cann, P. Emma, D. Greenberg, W. Haensch, M. Ignatowski, S. Koester, J. Magerlein, R. Puri, and A. Young, "Interconnects in the third dimension: Design challenges for 3D ICs," in *Proceedings of the 44th Annual Design Automation Conference*, pp. 562–567, 2007.

[13] B. Noia, S. K. Goel, K. Chakrabarty, E. J. Marinissen, and J. Verbree, "Test-Architecture Optimization for TSV-Based 3D Stacked ICs," in *IEEE European Test Symposium (ETS)*, pp. 24–29, May 2010.

[14] A. Sheibanyrad, F. Pétrot, and A. Jantsch, *3D integration for NoC-based SoC Architectures*. Springer, 2011.

[15] J. H. Lau, "Evolution and Outlook of TSV and 3D IC/Si Integration," in *Electronics Packaging Technology Conference*, pp. 560–570, 2010.

[16] P. Batude, M. Vinet, B. Previtali, C. Tabone, C. Xu, J. Mazurier, O. Weber, F. Andrieu, L. Tosti, L. Brevard, B. Sklenard, P. Coudrain, S. Bobba, H. Ben Jamaa, P. Gaillardon, A. Pouydebasque, O. Thomas, C. Le Royer, J. Hartmann, L. Sanchez, L. Baud, V. Carron, L. Clavelier, G. De Micheli, S. Deleonibus, O. Faynot, and T. Poiroux, "Advances, challenges and opportunities in 3D cmos sequential integration," in *Electron Devices Meeting (IEDM), 2011 IEEE International*, pp. 7.3.1–7.3.4, Dec 2011.

[17] P. Ramm, A. Klumpp, J. Weber, and M. M. V. Taklo, "3D System-on-Chip technologies for More than Moore systems," in *Microsystem Technologies*, vol. 16, pp. 1051–1055, July 2010.

[18] B. Black, M. Annavaram, N. Brekelbaum, J. DeVale, L. Jiang, G. Loh, D. McCauley, P. Morrow, D. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. Shen, and C. Webb, "Die stacking (3D) microarchitecture," in *Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 469–479, 2006.

[19] "Tezzaron Semiconductor," in *http://www.tezzaron.com/*, 2002.

[20] R. M. Chou, K. K. Saluja, and V. D. Agrawal, "Scheduling tests for VLSI systems under power constraints," in *IEEE Transactions on VLSI Systems*, vol. 5, pp. 175–185, June 1997.

[21] M. Bushnell and V. Agrawal, *Essentials of electronic testing for digital, memory and mixed-signal VLSI circuits*, vol. 17. Springer Science & Business Media, 2000.

[22] E. Larsson, *Introduction to advanced system-on-chip test design and optimization*, vol. 29. Springer Science & Business Media, 2006.

[23] S. Goel, "Test-Access Planning and Test Scheduling for Embedded Core-Based System Chips," in *University Press, Eindhoven, The Netherlands*, 2005.

[24] F. Corsi, S. Martino, and T. Williams, "Defect level as a function of fault coverage and yield," in *European Test Conference, 1993. Proceedings of ETC 93., Third*, pp. 507–508, IEEE, 1993.

[25] R. A. Frohwerk, "Signature analysis: A new digital field service method," *Hewlett-Packard Journal*, vol. 28, no. 9, pp. 2–8, 1977.

[26] "Semiconductor Industry Association (SIA)," in *International Technnology Roadmap for Semiconductors (ITRS)*, 2011.

[27] V. Muresan, X. Wang, V. Muresan, and M. Vladutiu, "Greedy Tree Growing Heuristics on Block-Test Scheduling Under Power Constraints," in *Journal of Electronic Testing: Theory and Applications*, pp. 61–78, 2004.

[28] S. Adham and E. Marinissen, "IEEE P1838," in *grouper.ieee.org/groups/3Dtest*, 2011.

[29] Y. Zorian, "A Distributed BIST Control Scheme for Complex VLSI devices," in *IEEE VLSI Test Symposium (VTS)*, pp. 6–11, Apr. 1993.

[30] Y.-J. Huang, J.-F. Li, J.-J. Chen, D.-M. Kwai, Y.-F. Chou, and C.-W. Wu, "A built-in self-test scheme for the post-bond test of TSVs in 3D ICs," in *IEEE VLSI Test Symposium (VTS)*, pp. 20–25, May 2011.

[31] A. Dahbura, M. Uyar, and C. Yau, "An optimal test sequence for the JTAG/IEEE P1149.1 test access port controller," in *IEEE International Test Conference (ITC)*, pp. 55–62, Aug. 1989.

[32] J. Andrews, "An embedded JTAG, system test architecture," in *IEEE Electro International Conference*, pp. 691–695, May 1994.

[33] L. Whetsel, "An IEEE 1149.1 based test access architecture for ICs with embedded cores," in *IEEE International Test Conference (ITC)*, pp. 69–78, Nov. 1997.

[34] X. Zhang, Y. Jiang, and J. Ju, "Specific Design and Optimization of JTAG IP Core," in *IEEE Circuits and Systems International Conference on Testing and Diagnosis (ICTD)*, pp. 1–4, Apr. 2009.

[35] B. I. Dervisoglu, "A Unified DFT Architecture for use with IEEE 1149.1 and VSIA/IEEE P1500 Compliant Test Access Controllers," in *IEEE Transactions on VLSI Systems*, pp. 53–58, June 2001.

[36] B. Mullane, M. Higgins, and C. MacNamee, "IEEE 1500 Core Wrapper Optimization Techniques and Implementation," in *IEEE International Test Conference (ITC)*, no. 29.2, pp. 1–10, Nov. 2008.

[37] C.-W. Wang, J.-R. Huang, Y.-F. Lin, K.-L. Cheng, C.-T. Huang, C.-W. Wu, and Y.-L. Lin, "Test scheduling of bisted memory cores for soc," in *Test Symposium, 2002. (ATS '02). Proceedings of the 11th Asian*, pp. 356–361, Nov 2002.

[38] Y. Huang, W.-T. Cheng, C.-C. Tsai, N. Mukherjee, O. Samman, Y. Zaidan, and S. Reddy, "Resource allocation and test scheduling for concurrent test of core-based SOC design," in *IEEE Asian Test Symposium (ATS)*, pp. 265–270, Nov. 2001.

[39] H. H-S, J.-R. Huang, K.-L. Cheng, W. C-W, C.-T. Huang, C.-W. Wu, and Y.-L. Lin, "Test scheduling and test access architecture optimization for system-on-chip," in *IEEE Asian Test Symposium (ATS)*, pp. 411–416, Nov. 2002.

[40] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test Wrapper and Test Access Mechanism Co-Optimization for System-on-Chip," in *Journal of Electronic Testing: Theory and Applications*, vol. 18, pp. 213–230, 2002.

[41] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test Access Mechanism Optimization, Test Scheduling, and Tester Data Volume Reduction for System-on-Chip," in *IEEE Transactions on Computers*, vol. 52, pp. 1619–1632, Dec. 2003.

[42] H. Yi, J. Song, and S. Park, "Low-Cost Scan Test for IEEE-1500-Based SoC," in *IEEE Transactions on Instrumentation and Measurement*, vol. 57, pp. 1071–1078, May 2008.

[43] M. A. Amory, E. Briao, E. Cota, M. Lubaszewski, and F. G. Moraes, "A Scalable Test Strategy for Network-on-Chip Routers," in *IEEE International Test Conference (ITC)*, no. 25.1, pp. 1–9, Nov. 2005.

[44] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Wrapper/TAM Co-Optimization, Constraint-Driven Test Scheduling, and Tester Data Volume Reduction for SOCs," in *IEEE VLSI Test Symposium (VTS)*, no. 44.3, pp. 685–690, June 2002.

[45] K. Chakrabarty, "Optimal test access architectures for system-on-a-chip," in *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 6, pp. 26–49, Jan. 2001.

[46] E. Larsson and Z. Peng, "An Integrated Framework for the Design and Optimization of SOC Test Solutions," in *Journal of Electronic Testing: Theory and Applications (JETTA)*, vol. 18, pp. 385–400, Aug. 2002.

[47] D. L. Lewis and H.-H. S. Lee, "A Scan-Island Based Design Enabling Pre-bond Testability in Die-Stacked Microprocessors," in *IEEE International Test Conference (ITC)*, pp. 1–8, 2007.

[48] X. Wu, P. Falkenstern, and Y. Xie, "Scan Chain Design for Three-Dimensional Integrated Circuits (3D ICs)," in *International Conference on Computer Design (ICCD)*, pp. 208–214, 2007.

[49] Y.-J. Lee and S. K. Lim, "Co-Optimization of Signal, Power, and Thermal Distribution Networks for 3D ICs," in *Electrical Design of Advanced Packaging and Systems Symposium*, pp. 163–166, 2008.

[50] L. Jiang, L. Huang, and X. Qiang, "Test Architecture Design and Optimization for Three-Dimensional SoCs," in *Design, Automation and Test in Europe (DATE)*, pp. 220–225, Apr. 2009.

[51] L. Jiang, X. Qiang, K. Chakrabarty, and T. M. Mak, "Layout-Driven Test-Architecture Design and Optimization for 3D SoCs under Pre-Bond Test-Pin-Count Constraint," in *International Conference on Computer-Aided Design (ICCAD)*, pp. 191–196, Nov. 2009.

[52] E. J. Marinissen, J. Verbree, and M. Konijnenburg, "A Structured and Scalable Test Access Architecture for TSV-Based 3D Stacked ICs," in *IEEE VLSI Test Symposium (VTS)*, pp. 1–6, Apr. 2010.

[53] J. Rajski, J. Tyszer, M. Kassab, N. Mukherjee, R. Thompson, K.-H. Tsai, A. Hertwig, N. Tamarapalli, G. Mrugalski, G. Eide, *et al.*, "Embedded deterministic test for low-cost manufacturing test," in *IEEE International Test Conference (ITC)*, pp. 916–922, 2002.

[54] A. Stroele, "Reducing bist hardware by test schedule optimization," in *IEEE Asian Test Symposium (ATS)*, pp. 253–258, 1992.

[55] T. Hiraide, K. Boateng, H. Konishi, K. Itaya, M. Emori, H. Yamanaka, and T. Mochiyama, "Bist-aided scan test-a new method for test cost reduction," in *IEEE VLSI Test Symposium, 2003. Proceedings. 21st*, pp. 359–364, 2003.

[56] "IEEE 1149.1: JTAG," in *http://grouper.ieee.org/groups/1149/1/*, 1990.

[57] K. P. Parker, "The Boundary-Scan Handbook," in *Kluwer Academic Publishers*, no. 3, 2003.

[58] H. Bleeker, P. V. D. Eijnden, and F. D. Jong, "Boundary-Scan Test: A Practical Approach," in *Kluwer Academic Publishers*, 1993.

[59] S. Oakland, "Considerations for implementing IEEE 1149.1 on system-on-a-chip integrated circuits," in *IEEE International Test Conference (ITC)*, pp. 628–637, June 2000.

[60] T. Caldwell, B.; Langford, "Is IEEE 1149.1 Boundary Scan Cost Effective: A Simple Case Study ," in *IEEE International Test Conference (ITC)*, pp. 106–109, Sept. 1992.

[61] L. Whetsel, "An approach to accelerate scan testing in IEEE 1149.1 architectures," in *IEEE International Test Conference (ITC)*, pp. 314 – 322, Oct. 1994.

[62] "IEEE 1500," in *http://grouper.ieee.org/groups/1500*, 2005.

[63] Y. Zorian, "Test Requirements for Embedded Core-based Systems and IEEE P1500," in *IEEE International Test Conference (ITC)*, no. 8.4, pp. 191–199, Nov. 1997.

[64] A. Benso, S. D. Carlo, P. Prinetto, and Y. Zorian, "IEEE Standard 1500 Compliance Verification for Embedded Cores," in *IEEE Transactions on VLSI Systems*, vol. 16, pp. 397–407, Apr. 2008.

[65] M. Higgins, C. MacNamee, and B. Mullane, "IEEE 1500 Wrapper Control using an IEEE 1149.1 Test Access Port," in *Irish Signals and Systems Conference (ISSC)*, pp. 198–203, June 2008.

[66] E. J. Marinissen and Y. Zorian, "IEEE Std 1500 Enables Modular SoC Testing," in *IEEE Design and Test of Computers*, pp. 8–16, Feb. 2009.

[67] M. Higgins, C. MacNamee, and B. Mullane, "Design and implementation challenges for adoption of the IEEE 1500 standard," in *IET Computers and Digital Techniques*, vol. 4, pp. 38–49, 2010.

[68] E. J. Marinissen, K. Chakrabarty, and V. Iyengar, "A Set of Benchmarks for Modular Testing of SOCs," in *International Test Conference (ITC)*, no. 19.1, pp. 519–528, 2002.

[69] E. Hopper and B. C. H. Turton, "An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem," in *European Journal of Operational Research*, vol. 128, pp. 34–57, Jan. 2001.

[70] "Tezzaron Semiconductor," in *http://www.tezzaron.com/*, 2002.

[71] B. Noia and K. Chakrabarty, "Post-Bond Test Wrappers and Emerging Test Standards.," in *IEEE European Test Symposium (ETS)*, pp. 159–180, 2014.

[72] B. Noia and K. Chakrabarty, "Pre-bond testing of die logic and TSVs in high performance 3D-SICs," in *IEEE International 3D Systems Integration Conference (3DIC)*, pp. 1–5, Jan. 2012.

[73] M. Taouil, M.andMasadeh, S. Hamdioui, and E. J. Marinissen, "Interconnect test for 3D stacked memory-on-logic,"

[74] S. Wang, K. Wang, R. andChakrabarty, and M. B. Tahoori, "Multicast Testing of Interposer-Based 2.5D ICs: Test-Architecture Design and Test Scheduling.," in *IEEE 25th Asian Test Symposium (ATS)*, pp. 86–91, Nov. 2016.

[75] F. Zhang, Y. Sun, X. Shen, K. Nepal, J. Dworak, T. Manikas, P. Gui, R. Bahar, A. Crouch, and J. Potter, "Using Existing Reconfigurable Logic in 3D Die Stacks for Test," in *IEEE 25th North Atlantic Test Workshop (NATW)*, pp. 46–52, May 2016.

[76] E. I. Vatajelu, P. Prinetto, M. Taouil and S. Hamdioui, "Challenges and Solutions in Emerging Memory Testing," in *IEEE Transactions on Emerging Topics in Computing*, 2018.

[77] B. SenGupta and E. Larsson, "Test Planning and Test Access Mechanism Design for Stacked Chips using ILP," in *IEEE VLSI Test Symposium (VTS)*, pp. 1–6, Apr. 2014.

[78] B. SenGupta, D. Nikolov, U. Ingelsson, and E. Larsson, "Test Planning for Core-based Integrated Circuits under Power Constraints," in *Journal of Electronic Testing: Theory and Applications (JETTA)*, vol. 33, pp. 7–23, Feb. 2017.

[79] M. Agrawal, K. Chakrabarty, and B. Eklow, "Test-Cost Optimization and Test-Flow Selection for 3D-Stacked ICs," in *IEEE VLSI Test Symposium (VTS)*, pp. 1–6, Apr. 2013.

[80] M. Richter and K. Chakrabarty, "Optimization of test pin-count, test scheduling, and test access for noc-based multicore socs," vol. 63, pp. 691–702, March 2014.

[81] M. Taouil, S. Hamdioui, K. Beenakker, and E. J. Marinissen, "Test impact on the overall die-to-wafer 3D stacked IC cost," vol. 28, pp. 15–25, Springer US, 2012.

[82] M. Taouil and S. Hamdioui, "Yield Improvement for 3D Wafer-to-Wafer Stacked Memories," in *Journal of Electronic Testing: Theory and Applications (JETTA)*, vol. 28, pp. 523 – 534, Jan. 2012.

[83] M. Taouil, S. Hamdioui, and E. Marinissen, "How significant will be the test cost share for 3D die-to-wafer stacked-ICs?," in *Design Technology of Integrated Systems in Nanoscale Era (DTIS), 2011 6th International Conference on*, pp. 1–6, April 2011.

[84] M. Taouil, S. Hamdioui, J. Verbree, and E. J. Marinissen, "On Maximizing the Compound Yield for 3D Wafer-to-Wafer Stacked ICs," in *International Test Conference (ITC)*, no. 6.2, pp. 1–10, Sept. 2010.

[85] M. Taouil and S. Hamdioui, "On optimizing test cost for wafer-to-wafer 3D-stacked ICs," in *Design Technology of Integrated Systems in Nanoscale Era (DTIS), 2012 7th International Conference on*, pp. 1–6, May 2012.

[86] S. Hamdioui and M. Taouil, "Yield Improvement and Test Cost Optimization for 3D Stacked ICs," in *Asian Test Symposium (ATS)*, pp. 480 – 485, Nov. 2011.

[87] J. Lau, "Tsv manufacturing yield and hidden costs for 3D IC integration," in *Electronic Components and Technology Conference (ECTC), 2010 Proceedings 60th*, pp. 1031–1042, June 2010.

[88] E. Marinissen, "Challenges in testing tsv-based 3D stacked ICs: Test flows, test contents, and test access," in *Circuits and Systems (APCCAS), 2010 IEEE Asia Pacific Conference on*, pp. 544–547, Dec 2010.

[89] G. Smith, L. Smith, S. Hosali, and S. Arkalgud, "Yield considerations in the choice of 3D technology," in *Semiconductor Manufacturing, 2007. ISSM 2007. International Symposium on*, pp. 1–3, Oct 2007.

[90] E. J. Marinissen, C.-C. Chi, J. Verbree, and M. Konijnenburg, "3D DfT architecture for pre-bond and post-bond testing," in *IEEE 3D Systems Integration Conference (3DIC)*, pp. 1–8, Nov. 2010.

[91] M. Taouil, S. Hamdioui, and E. J. Marinissen, "On modeling and optimizing cost in 3D Stacked-ICs," in *IEEE International Design and Test Workshop (IDT)*, vol. 6, pp. 24–29, Dec. 2011.

[92] M. Taouil and S. Hamdioui, "Layer redundancy based yield improvement for 3D wafer-to-wafer stacked memories," in *European Test Symposium (ETS), 2011 16th IEEE*, pp. 45–50, May 2011.

[93] Y. Chen, D. Niu, Y. Xie, and K. Chakrabarty, "Cost-effective integration of three-dimensional (3D) ICs emphasizing testing cost analysis," in *Proceedings of the International Conference on Computer-Aided Design (ICCAD)*, pp. 471–476, 2010.

[94] J. Xie, Y. Wang, and Y. Xie, "Yield-aware time-efficient testing and self-fixing design for tsv-based 3D ICs," in *Design Automation Conference (ASP-DAC), 2012 17th Asia and South Pacific*, pp. 738–743, Jan 2012.

[95] K. Sakuma, P. Andry, C. Tsang, K. Sueoka, Y. Oyama, C. Patel, B. Dang, S. Wright, B. Webb, E. Sprogis, R. Polastre, R. Horton, and J. Knickerbocker, "Characterization of stacked die using die-to-wafer integration for high yield and throughput," in *Electronic Components and Technology Conference, 2008. ECTC 2008. 58th*, pp. 18–23, May 2008.

[96] C. Ferri, S. Reda, and R. Bahar, "Strategies for improving the parametric yield and profits of 3D ICs," in *IEEE/ACM International Conference on Computer-Aided Design, ICCAD*, pp. 220–226, Nov 2007.

[97] C. Ferri, S. Reda, and R. I. Bahar, "Parametric yield management for 3D ICs: Models and strategies for improvement," vol. 4, pp. 19:1–19:22, ACM, Nov. 2008.

[98] E. Singh, "Exploiting rotational symmetries for improved stacked yields in w2w 3D-SICs," in *VLSI Test Symposium (VTS), 2011 IEEE 29th*, pp. 32–37, May 2011.

[99] X. Dong and Y. Xie, "System-level cost analysis and design exploration for three-dimensional integrated circuits (3D ICs)," in *Asia and South Pacific Design Automation Conference, ASP-DAC*, pp. 234–241, Jan 2009.

[100] P. Mercier, S. R. Singh, K. Iniewski, B. Moore, and P. O'Shea, "Yield and cost modeling for 3D chip stack technologies," in *Custom Integrated Circuits Conference, 2006. CICC '06. IEEE*, pp. 357–360, Sept 2006.

[101] J. Zhao, X. Dong, and Y. Xie, "Cost-aware three-dimensional (3D) many-core multiprocessor design," in *Proceedings of the 47th Design Automation Conference*, DAC '10, pp. 126–131, ACM, 2010.

[102] S. Fortune and C. J. Van Wyk, "Static analysis yields efficient exact integer arithmetic for computational geometry," vol. 15, pp. 223–248, ACM, July 1996.

[103] M. Taouil, S. Hamdioui, K. Beenakker, and E. Marinissen, "Test cost analysis for 3D die-to-wafer stacking," in *Test Symposium (ATS), 2010 19th IEEE Asian*, pp. 435–441, Dec 2010.

[104] B. SenGupta, U. Ingelsson, and E. Larsson, "Scheduling Tests for 3D Stacked Chips under Power Constraints," in *Journal of Electronic Testing: Theory and Applications (JETTA)*, vol. 28, pp. 121–135, Jan. 2012.

[105] E. J. Marinissen, C.-C. Chi, M. Konijnenburg, and J. Verbree, "A DfT Architecture for 3D-SICs Based on a Standardizable Die Wrapper," in *Journal of Electronic Testing: Theory and Applications (JETTA)*, vol. 28, pp. 73–92, Jan. 2012.

[106] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[107] Y. Zorian, "Built-in self-test," vol. 49, pp. 135 – 138, 1999.

[108] B. Jansen and A. Vandegoor, "Built-in self test," vol. 89, p. 18468, Nov. 1988.

[109] A. Miczo, "Built-in self-test," in *Digital Logic Testing and Simulation*, pp. 451–512, John Wiley and Sons, Inc., 2003.

[110] E. J. Marinissen, V. Iyengar, and K. Chakrabarty, "ITC'02 SOC Test Benchmarks (http://itc02socbenchm.pratt.duke.edu/)," in *International Test Conference (ITC)*, 2002.