ACHIEVING OPTIMAL SYSTEM-ON-CHIP TEST SCHEDULES

by

Spencer K. Millican

A thesis submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy
(Electrical Engineering)

at the
UNIVERSITY OF WISCONSIN–MADISON
2015

Date of the final oral examination: 02/13/15

The dissertation is approved by the following members of the Final Oral Committee:
Kewal K. Saluja, Professor, Electrical and Computer Engineering
Parameswaran Ramanathan, Professor, Electrical and Computer Engineering
Mikko H. Lipasti, Philip Dunham Reed Professor, Electrical and Computer Engineering
Azadeh Davoodi, Associate Professor, Electrical and Computer Engineering
Jing Li, Assistant Professor, Electrical and Computer Engineering
Jeffrey T. Linderoth, Professor, Instrial and Systems Engineering

ACKNOWLEDGMENTS

First and foremost, I would like to thank my Adviser, Professor Kewal K. Saluja. For the past five years and the majority of my time at UW-Madison, his guidance has been critical in completing my research, and his ideas have immeasurably contributed to the subject matter of my studies. I would also like to thank his countless contributions to my writing process, which in turn have lead to several conference and journal papers, as well as this thesis.

I would also like to thank the members of my defense committee, Professors Parameswaran Ramanathan, Mikko H. Lipasti, Azadeh Davoodi, Jing Li, and Jeffrey T. Linderoth, for taking their time to contribute to the quality of this thesis with their indispensable knowledge and suggestions.

I would also like to thank all the faculty and staff of the Electrical and Computer Engineering Department and the Computer Science Department at the University of Wisconsin - Madison for the highest quality of education they have provided, for without the high quality of education I have received from my time as a graduate and undergraduate student here, the completion of this thesis would not be possible.

Finally, I would like to thank the support my close friends and family. It was their inspiration which guided me into this field, and it was their love and support which guided me to complete this thesis.

ABSTRACT

The development and manufacturing of microprocessor integrated circuits (ICs) is becoming an ever more difficult task with decreasing IC feature sizes. Ever since the introduction of silicon-based ICs, it has been predicted that the feature sizes of these ICs would continue to decrease, and therefore lead to more transistors being packed onto a single die. Although a natural consequence of this increased transistor density has been the reduced cost of IC-based systems, new design methods had to be introduced to make effective use of the increased IC density. Along with the design of microprocessor ICs, the manufacturing of ICs has also seen new challenges as a result of transistor-dense dies. During the manufacturing process, each transistor must be thoroughly tested to ensure its correct functionality, which becomes more difficult as more transistors are packed on a die.

As the feature sizes of ICs continue to scale downwards, the power and temperature density of ICs has risen to the point where it can no longer be ignored. In past, the power density of ICs was small enough that it was an afterthought of microprocessor design, but as ICs have become more transistor-dense, this is no longer the case, since increased IC power density can lead to device failure, device damage, or devices not suitable for their desired application (e.g., mobile computing). During test, these problems are especially problematic since the power density during test is higher than during normal device operation, and the violation of power constraints during test can lead to higher IC manufacturing costs. Along with increased power density, increased device temperatures have also become a problem for IC design with recent technologies, as high device temperature can damage a device. High temperature is especially problematic during test, since the environment of test is less temperature-friendly than normal device operation.

The goal of this work is to address power and temperature constraints in a manufacturing test environment. During test, not only must power and temperature constraints be met, but also the time required to apply all tests must be kept at a minimum in order to reduce testing costs. To reduce the testing time under these constraints, this work will make use of technology previously used in advanced microprocessors (dynamic voltage and frequency scaling). This work will also present a scheduling formulation that incorporates many different constraints to allow the scheduling of tests under many different environments, whereas previous formulations concentrated on specific environments. By doing so, the cost of applying tests to a device is kept at a minimum while not violating modern design constraints.

TABLE OF CONTENTS

Page

ABSTRACT ii

LIST OF TABLES vii

LIST OF FIGURES viii

GLOSSARY x

1 Introduction 1

1.1 IC Manufacturing and Test 2
1.2 Contributions 3
1.2.1 Utilizing DVFS For Power-Constrained SoC Test Scheduling 4
1.2.2 Optimal Scheduling of Tests Under Various Constraints 4
1.2.3 Optimal Scheduling of Tests Under Temperature Constraints 4
1.2.4 Temperature-Aware Test Partitioning 5
1.3 Organization 5

2 Role of Test in IC Manufacturing 6

2.1 The Imperfect Nature of Silicon IC Manufacturing 6
2.2 Test Methodology 8
2.3 Test Time Economics 10
2.4 Test Application Time Reduction Techniques 11
2.5 The Increasing Complexity of IC Test 13

3 SoC Design and Test 14

3.1 SoC Design Model 14
3.2 SoC Testing Model 16
3.3 SoC Test Constraints 17

4 Utilizing DVFS For Power-Constrained SoC Test Scheduling 21

4.1 Introduction 21
4.2 Past Work 22

Page

4.2.1 Session Based & Sessionless Test Scheduling . . . 22
4.2.2 Power & DFVS Test Scheduling . . . 24
4.3 Session Based Testing . . . 25
4.3.1 Test Environment and modeling . . . 26
4.3.2 Formulation . . . 28
4.4 Sessionless Test Scheduling . . . 30
4.5 Experimental Setup . . . 32
4.5.1 Benchmarks . . . 32
4.5.2 Implementation . . . 33
4.6 Results . . . 34
4.7 Conclusions . . . 37

5 Optimal Scheduling of Tests Under Various Constraints . . . 39

5.1 Introduction . . . 39
5.2 Related Works . . . 40
5.3 Test Scheduling Formulation . . . 41
5.4 Experiments . . . 45
5.5 Results . . . 47
5.6 Extensions for Other Constraints . . . 50
5.7 Conclusion . . . 51

6 Optimal Scheduling of Tests Under Temperature Constraints . . . 52

6.1 Introduction . . . 52
6.2 Past Work . . . 53
6.3 Methodology . . . 55
6.4 Optimistic MILP Formulation . . . 57
6.4.1 Previously Used Constraints . . . 57
6.4.2 Temperature Constraint . . . 57
6.5 Pessimistic MILP Formulation . . . 59
6.6 Results . . . 61
6.7 Conclusions . . . 64

7 Temperature-Aware Test Partitioning . . . 66

7.1 Introduction . . . 66
7.2 Test Partitioning . . . 67
7.3 Proposed Partitioning Method . . . 68
7.3.1 Partition Quality . . . 68

Appendix

Page

7.3.2 Partitioning Method 70
7.4 Experiment 71
7.5 Results 73
7.6 Conclusion 76

8 Summary 77

8.1 Conclusions 77
8.2 Future Directions 78
8.2.1 Heuristic Benefit Analysis 79
8.2.2 DVFS Benefit Analysis 79
8.2.3 Power Reduction Hardware 81

LIST OF REFERENCES 82

LIST OF TABLES

Table Page

4.1 Benchmark Information 34

4.2 Benchmark Results 35

5.1 Benchmark Information 46

5.2 Schedule Results 47

6.1 Benchmarks Composing Different Stacks 62

6.2 Benchmark Results Under Different Formulations 63

7.1 Benchmark Details 74

7.2 Schedule results (TAT) for T_{PART} = 40° C in ms 74

7.3 Schedule results (TAT) for T_{PART} = 30° C in ms 75

LIST OF FIGURES

Figure Page

2.1 A typical MOSFET transistor, and one with a defect in silicon oxide layer. 7

2.2 Ideal metal layers, common metal layers, an metal open defect, and a metal short defect. 8

2.3 An example of test wrapper and scan chain implementation. 13

3.1 Example of a modern SoC (©Analog Devices) 15

4.1 Presuming that test 2 and 5 are compatible, test time can be reduced by eliminating session requirements. 23

4.2 By scaling V_{DD} with frequency as opposed to frequency alone, better power reduction can be achieved. 27

4.3 As more V_{DD} options become available, better schedules can be obtained. 36

4.4 As more V_{DD} options become available, sessionless scheduling will still give better schedules than session-based schedules. 37

5.1 A true maximum power of this schedule is 15 W, but Section 4.4 will record it as 25 W. 43

5.2 Normalized TAT results with invalid schedules marked. 49

5.3 TAT result of two stacked f2126 benchmarks as more pins and TSVs are allowed for scheduling. 49

6.1 An example of partitioning a single test into regions. 59

6.2 Scheduled times of various formulations. 63

7.1 By partitioning, greater overlap between tests t_1 and t_2 can be achieved. 68

7.2 By partitioning before T_{PART} is met, the temperature of all partitions is reduced. . . . 71

Figure Page

7.3 Effect of T_{PART} on the number of partitions. 73

7.4 Normalized schedule results of different partitioning methods. 76

8.1 Examples of homogeneous and heterogeneous designs. 81

GLOSSARY

3D-IC 3D-stacked integrated circuit. 39–41, 44, 46, 50, 53, 56, 61, 64, 65, 67, 71, 72, 74, 76

ATE automatic test equipment. 3, 10, 80

ATPG automatic test pattern generation. 9, 12, 13

BIST built-in self-test. 12, 40, 50

DUT design under test. 52, 54, 67

DVFS dynamic voltage and frequency scaling. 4, 5, 21, 22, 24–26, 28, 31, 33, 34, 36–40, 42, 44, 45, 47, 48, 57, 78–81

IC integrated circuit. 1–14, 17, 19–21, 24, 25, 66, 74, 77, 78

IP intellectual property. 15–17

JTAG Joint Test Action Group. 12

MILP mixed-integer linear programming. 19, 21, 23, 24, 26, 28, 33, 35, 38, 47, 52, 53, 55, 64, 72

SoC system-on-chip. 5, 14–22, 32, 39–41, 48, 51, 67, 77–81

TAM test access mechanism. 18, 40–48, 50, 51, 57

TAT test application time. 4, 5, 10, 11, 13, 17, 18, 20–23, 28–31, 33–41, 46–48, 50, 51, 66–68, 73, 75, 77–81

TSV through-silicon via. 44–48, 50, 51

TTSV thermal through-silicon via. 55, 56, 64

Chapter 1

Introduction

The ever downward scaling of integrated circuit (IC) feature sizes in recent decades has led to an abundance of microprocessors in the consumer marketplace. In the fairly recent past, the cost of a single computing system was large enough to restrict their use to large industrial applications. The high cost of computation came from the large number of ICs required to implement a single computing system, as the cost per IC was high, as was the cost to interconnect multiple ICs. However, as the feature sizes of ICs scaled downwards in compliance with Moore's Law [1], the number of ICs required to implement a given computing system decreased, which in turn decreased the cost to implement computing systems. The ever decreasing cost of implementing such systems also makes advanced digital devices more available to consumers. The availability of such devices has become common enough that many consider high-performance microprocessors as a low-cost appliance, and even expect the performance of such devices to increase in the future with little or no added cost.

However, as digital microprocessors become ever more common, new issues arise in the development and manufacturing of these devices to meet the demand of consumers. To keep the cost of ICs down, the cost to manufacture a single correctly operating transistor must also decrease [1]. However, recent challenges in IC manufacturing has made the compliance of Moore's law difficult. A challenge came in the form of the unforeseen consequences of smaller IC feature sizes, which have lead to devices operating outside of Moore's predicted trends [2]. Another challenge to modern IC design is in the manufacture the device, which is ever more difficult as transistors become smaller and as more transistors are compacted onto a single die.

This study focuses on one particular issue in the manufacturing of ICs, i.e. the testing for defects in manufactured dies. This challenge must be met during the manufacturing process, but the fulfillment of this task becomes more difficult as more transistors are packed onto a die. Hence, the goal of this research is to fulfill this task while keeping the cost of fulfillment low, thereby allowing consumers to benefit from ever more powerful microprocessors.

1.1 IC Manufacturing and Test

The manufacturing of silicon-based ICs has never been a perfect process. IC manufacturing is a process of changing the chemical makeup of regions of a silicon material, which in turn create individual devices. However, chemical impurities in the silicon material, or imperfections in the size and shape of the features made, can cause the operation of the resulting devices to be imperfect. Such imperfect devices in an IC can create a circuit which does not function according to specification, which in turn can cause the system which uses such circuits to fail. As the feature sizes of ICs decrease, imperfections and impurities unfortunately become more common due to smaller margins of error in the manufacturing process, which in turn leads to more device defects. Although literature and techniques in the reduction of manufacturing defects are ongoing [3, 4], the complete elimination manufacturing defects does not appear to be possible by any means.

Due to the imperfect nature of ICs manufacturing, test is a critical stage in the manufacturing process. Catching defective ICs is important, as the release and subsequent use of a defective IC can have disastrous consequences, or at best, can damage the reputation of the manufacturer. Although it is impossible to prevent all manufacturing defects from occurring, it is still possible to catch defects in a device before it is released. Although catching and discarding all defective ICs will reduce the yield of produced devices, it is still a favorable alternative compared to releasing a faulty device. For this reason, IC test today is performed on all varieties of ICs, ranging from single-gate dies to the largest of microprocessors [5].

Testing has a critical role in IC manufacturing, and fulfilling test requirements is becoming more difficult as IC feature sizes continue to scale downwards. Since the number of defects in an IC is proportional to the number of transistors on a die, the increasing number of transistors on

any die increases the number of defects that must be tested for. At the same time, the complexity of modern ICs is making "access" to individual transistors more complex, and therefore applying tests to hard-to-reach transistors is becoming difficult. Such difficult transistors, which again are growing in number, must still be tested to guarantee a non-faulty device, which in turn requires a larger investment in test resources to test such transistors.

IC testing is not only becoming more difficult due to transistor-dense ICs, but also due to new design constraints not seen in previous technologies. As IC feature sizes scale downwards, the number of transistors on a single die is not the only parameter that is increasing. Moore's Law [1] originally predicted that other parameters, such as power density, would scale at the same rate as the number of transistors on die, but unfortunately, this trend did not hold true [6]. Ignoring such parameters an lead to temporary device malfunctions or even device destruction. In the case of testing, ignoring such parameters can lead to false failures or destruction of a good device, which leads to a reduction in manufacturing yield and therefore increased manufacturing cost.

Another task that has become more difficult is reducing testing costs, in terms of testing hardware and time requirements. The application of tests to a manufactured IC is by no means free, as extra development time and hardware is required to create and apply such tests. The automatic test equipment (ATE) required to apply tests to devices is costly due to their automated and precise nature, and many manufacturers are reluctant to invest in more than the minimal amount of ATE [7]. Unfortunately for the manufacturer, high throughput demands are placed on IC production, and therefore the manufacturer must test each IC in as little time as possible to avoid investment in additional testing equipment. Although IC design alternatives, such as making IC "self-testing", can reduce the need for extra test equipment, such hardware is an investment on the part of the designer, because such hardware requires extra die area to implement [8]. Therefore, effort to reduce the need for such testing equipment is cost effective.

1.2 Contributions

Due to new challenges in IC test, certain issues must be addressed in order to fulfill IC test requirements. The main contributions of this dissertation are as follows.

1.2.1 Utilizing DVFS For Power-Constrained SoC Test Scheduling

As new hardware is introduced to allow more efficient processor operation, the same hardware can be utilized to more efficiently apply tests. Also, the reduction of test application time (TAT), and consequently test cost, has become more difficult under power constraints, especially as ICs become more transistor dense. A new technology, dynamic voltage and frequency scaling (DVFS), was introduced in recent decades to reduce the power consumption of microprocessors while not under full utilization, but such technology has only marginally been used for TAT reduction. In this study, DVFS is utilized to its full potential to reduce the TAT of an IC while still enforcing a power limit to prevent false device failures.

1.2.2 Optimal Scheduling of Tests Under Various Constraints

As new environments and constraints are added to manufacturing tests, all these constraints must be met to guarantee the tests being applied are valid. For the purpose of reducing TAT, presumptions on the testing environment have often been less-than-optimal. By making such poor presumptions, it is possible to either obtain a manufacturing test schedule that is invalid (thereby possibly causing damage to a device), or to generate a higher TAT than required. In both cases, the cost of testing is increased. This study presents a formulation which has the ability to meet many different environmental constraints, thereby obtaining optimal test schedules for many different testing applications.

1.2.3 Optimal Scheduling of Tests Under Temperature Constraints

The introduction of smaller IC feature continues to introduce more testing constraints that must be met to guarantee a thorough and valid test. As the power density of ICs continues to increase, so does the temperature dissipation of such ICs. Violating a temperature bound, much like violating a power bound, can cause the destruction of a device, and therefore such a bound must be enforced both during normal operation and during test. However, enforcing such a bound is more difficult during test due to the unique environment of test. Also, although there is a direct link between power and temperature, the modeling of temperature is an inherently more difficult problem, and

therefore special considerations must be made for enforcing a temperature constraint during test. This study formulates an optimal testing method for enforcing a temperature bound for tests while still minimizing TAT.

1.2.4 Temperature-Aware Test Partitioning

Several methods methods for reducing the TAT of tests have been introduced across many different studies, ranging from extra hardware to different test generation methods. These methods have built upon previous methods, which in turn has increased the complexity of thoroughly testing a device. However, with decreasing IC feature sizes, the complexity of implementing such methods has been brought to the point where implementing them is becoming infeasible. As an alternative, simpler methods for testing are desired for testing which do not increase the complexity of test while still reducing TAT. This study introduces a method of utilizing previous test scheduling formulations to reduce TAT under temperature-constrained environments through the partitioning of tests.

1.3 Organization

This dissertation is organized as follows. Chapter 2 explains the role test has in IC manufacturing and the methods used to test ICs. Chapter 3 explains the system-on-chip (SoC) design model, which is frequently used in IC test, and explains how it is applied to TAT reduction. Chapter 4 gives a formulation for the optimal test scheduling of SoC under power constraints while utilizing DVFS, while Chapter 5 introduces many more constraints to allow SoC test scheduling under many new environments. Chapter 6 introduces temperature to the SoC test scheduling problem. Chapter 7 introduces a method of temperature-aware portioning for the purpose of reducing TAT while not increasing test complexity. This dissertation concludes with Chapter 8, which summaries this work and gives future directions that can be taken.

Chapter 2

Role of Test in IC Manufacturing

This chapter discusses the IC manufacturing process and the reasons test is a critical part of the manufacturing process. This chapter will explain the imperfect nature of silicon IC manufacturing and the steps required to detect imperfections.

2.1 The Imperfect Nature of Silicon IC Manufacturing

In theory, the IC manufacturing process should always produce ICs which meet the specifications of the designer, but in reality, steps in the IC manufacturing process can introduce defect(s). The process of silicon IC manufacturing uses a series of masks to create layers of distinct features, with the width of these features ranging from a millimeter to a few nanometers [9]. These features form changing silicon types, creating oxide layers, creating metal interconnects, and many other processes used to create individual devices and interconnect them. Ideally, the features created by these masks are perfect, right-angle shapes determined by the masking process, but because of impurities of the chemistry involved and the small feature sizes being worked with, abnormalities can occur which fall outside the margins of error for proper device construction. This section will show several such defects and discuss the effect they can have on a digital circuit.

One type of flaw present in MOSFET transistors (the most common kind transistor used in modern digital circuits) is a silicon-oxide short [10]. The principle behind the functionality of field-effect transistors is to create an electrical field between two nodes which will either block the transmission of current (p-type) or allow the transmission of current (n-type), whereas not creating the electrical field will have the opposite effect. The field which allows for this functionality is

generated by applying voltage between the bottom silicon substrate and top metal layer, with a silicon oxide layer in between to block current flow. However, if there is a defect in the silicon oxide layer (as illustrated in Figure 2.1), then current will flow, which will in turn cause a voltage change on the transistor source/drain. For a digital circuit, this can cause the logic level on the output of a gate to be equal to the gate connection of the faulty transistor.

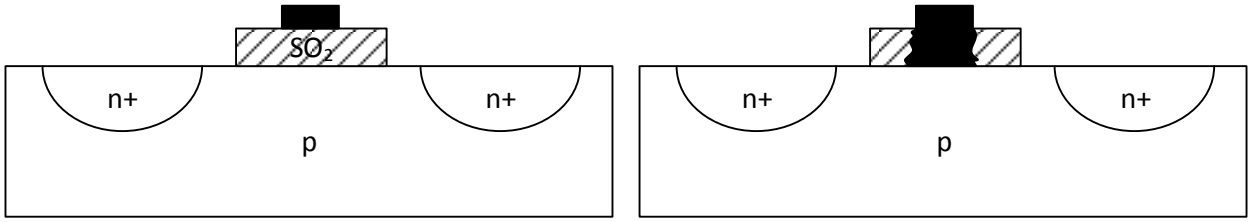

Figure 2.1 A typical MOSFET transistor, and one with a defect in silicon oxide layer.

Two more types of defects are metal-layer shorts and metal-layer opens. In silicon-based ICs, the interconnection of individual transistors is done through several layers of copper conduits. As the manufacturing technique of these metal layers is similar to the method used to generate silicon features, these metal errors are prone to errors, particularly errors in geometry. Imperfect geometries in the manufacturing process are created by imperfect "right angle features", as illustrated in Figure 2.2. As IC feature sizes continue to decrease, these imperfect right-angles can fall outside the margins of error to create defects. One such defect occurs when two or more metal interconnects touch each other, which creates a short [11], which in turn will cause a digital signal to drive unintended elements. Another such defect is when a metal interconnect is broken to create an open [12], which in turn will prevent desired signals from reaching their intended destination. Unfortunately for modern technology designers, such defects can be "partial", since having "thin" metal interconnects can change the resistance of metal elements and thus cause a device to fail when operating at their intended speed [12].

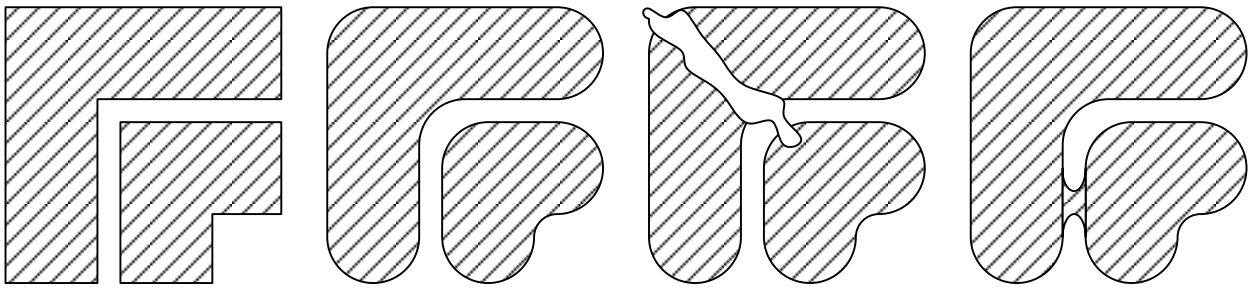

Figure 2.2 Ideal metal layers, common metal layers, an metal open defect, and a metal short defect.

2.2 Test Methodology

Although the act of testing an IC is conceptually simple, in practice applying tests to an IC can be cumbersome. This section will discuss the increasing complexity of IC test and why simple functional tests are inadequate to test an IC.

Although one would ideally model defects using accurate electrical simulation, such models are often infeasible due to their computational complexity. Since ICs are electrical circuits, any defect in an IC can be modeled using accurate electrical simulation, for instance by using SPICE [13]. Using such simulations would be ideal, since most defects would manifest themselves as a change in electrical parameters, such as increased/decreased resistance. However, such a method for modeling faults is impractical for two reasons. First, simulating a defect in such a manner is computationally intensive, and therefore the time to confirm whether or not a test will detect a defect is impractical. For example, if microprocessor test is 4000 instructions long, and each state of the machine takes 1 minute to simulate, than it would take 4000 minutes to simulate the "normal" operation of the circuit, and 4000 more minutes to confirm if the test will detect any given fault or defect. Second, the number of faults to model in such a matter is too large, since a single defect can manifest itself in many different ways. For example, a metal-open defect can manifest itself with marginally higher resistance, significantly higher resistance, or practically infinite resistance, and the same kind of "parametrization" can be applied to any other defect in the circuit.

To cope with cumbersome defect modeling, simpler fault models, such as the stack-at-fault model [14], were developed to create tests for all faults. Although the electrical properties of all

possible defects are diverse, the manner in which they manifest themselves in digital circuits are often similar. For instance, many defects of a single transistor can leave the output of a gate stuck at a particular logic value, or a metal-close fault can cause one gate's input to always be equal to another's. By creating a test which finds if the input/output of a gate is stuck at a given logic value, many defects in the circuit can be found at the same time by finding a test for the modeled fault. For this reason, the stuck-at-fault model [14] has been adopted as an industry standard. The advantage of adapting such a fault model is that it not only simplifies the simulation to that of a logic simulation, but it also greatly reduces the number of defects to model since test for a single stuck-at-fault can detect multiple manufacturing defects.

Although one would ideally apply simple functional tests to an IC to detect all stuck-at-faults, such functional tests rarely capture all possible defects in a circuit without becoming impractical. The ideal test case would be to apply a sample input to a circuit (e.g., a test program), and then evaluate the output to confirm the circuit as non-faulty. However, this rarely, if ever, will detect all possible defects in a circuit. With modern complex multiprocessors containing billions of transistors, it is infeasible to stimulate every transistor in a circuit using a test program, let alone capture the faulty output of every transistor. Although it was possible in the past to apply every possible input/state combination to the circuit to guarantee the circuit's correctness, this is impossible for large circuits. A basic example of this is a 32-bit adder, which has 64 inputs, which in turn has 2^{64} possible input combinations, which for a tester running at 10 GHz would require $2^{64}/(10^{10}) = 58.5$ years to test.

To assist in detecting faults not found through simpler testing methods, new structural testing methods were developed to find tests for specific faults. Although applying all possible circuit input/state combinations would certainly detect all stuck-at-faults in a circuit, doing so is not necessary. If the structure of the circuit is known, an automatic test pattern generation (ATPG) program [15] can find specific test which can detect a given fault. By knowing the structure of the circuit, one can find inputs to the circuit which excite the fault, and then find other inputs to the circuit which can guarantee that the faulty behavior is seen on a circuit output. By creating tests in such a

way, the number of input vectors required to confirm a circuit has no faults is reduced to the number of faults in the circuit, which is often far fewer than the number of input/state combinations of the circuit. Also, it may be possible to detect more than one fault with a single input combination, which in turn will reduce the number of input vectors to apply even further.

2.3 Test Time Economics

As IC complexity has increased, the time to thoroughly test to an IC, i.e. TAT, has also increased. As more individual transistors are packed onto a single die, more circuit input vectors are required to test every transistor in the circuit. Although some compaction can be achieved by having a single input vector excite multiple transistors, the rate of test compaction is far less than the rate of increasing transistor density. Also, the increasing complexity of microprocessor ICs has made access to transistors more difficult. Further, more difficult transistor access has led to more time to apply a single test, for instance loading test vectors and capturing their responses.

One motivation for the IC tester to decrease TAT is that the time to test an IC can become a bottleneck in the manufacturing process, thereby limiting manufacturing output. For any IC fabricator, throughput demands are being set by the IC designers and consumers, and not meeting these demands can be disastrous. Therefore, the testing process (as well as every other step in the manufacturing process) must happen at a rate equal to or more than the demanded throughput. However, with testing, meeting this throughput demand is becoming more difficult with increasing TAT. Also problematic for meeting throughput is that traditional methods for increasing throughput, i.e. testing multiple devices at once, are costly, since the ATE required to test a device are expensive, high-performance machines, and most fabricators can only afford a few of these machines [16].

Another motivation for decreasing the TAT of an IC is the time-cost relationship for the testing of an IC is unfavorable, even when throughput demands are met. The ATE required to test an IC are not only expensive in their initial purchase cost, but also in their operating and maintenance cost [17]. This is due to their complex electrical and mechanical nature, as well as their use of high-accuracy measurement electronics. Therefore, the more ATE has to operate, the higher the cost to test a device, and for this reason many designers will focus on the reduction of TAT even when

throughput demands are met. This reduction in TAT is so desirable that seemingly insignificant changes in TAT (e.g., 10 ms to 5 ms) are often worth the investment, especially since it is required to repeat the same test multiple times during the manufacturing process (e.g., wafer test, package test, board test, etc.)

2.4 Test Application Time Reduction Techniques

Due to the high cost of TAT for an IC, much existing work has focused on decreasing the TAT of an IC. These efforts have ranged from redesigning microprocessor hardware, modeling defects more efficiently, to developing more efficient tests to detect more faults. This section gives a few such techniques.

Initial work on reducing TAT reduction focused on decreasing the number of test vectors required to test a circuit [18]. As stated in Section 2.3, as more transistors are present in a die, more test vectors are required to test them. Since more test vectors take a longer time to apply (often much longer depending on the test hardware available), great TAT reduction can be had by simply reducing the number of input vectors required to test a device. One method of reducing the number of test vectors is to generate more efficient tests, i.e., tests which detect multiple faults. Such methods use the structure of the circuit to find "equivalent" faults which can be detected by the same vector [19]. Another method is to compact existing test vectors which are "compatible" with each other. Many fault-specific tests can express unused inputs as "don't-cares", which can allow other tests to be applied "at the same time" if these other tests only require other inputs to be applied, or at least need the same values on inputs already in use.

A further approach to reducing TAT is to introduce new hardware for the purpose of applying tests more efficiently. As stated in Section 2.3, accessing transistors for the purpose of applying tests or observing test responses becomes more difficult as circuits become more dense. Without using extra hardware, accessing difficult transistors requires multiple clock cycles to move test inputs and outputs through sequential circuits elements, which can be time-consuming. However, test-specific hardware can either make the movement of test inputs and/or responses easier so as

to save time. Some examples of test hardware (one of which is commonly used for non-test applications) are circuit wrappers and scan-chain flip-flops. To assist in accessing inputs and outputs in embedded "sub-circuits", test-wrappers were developed [20]. These wrappers, as illustrated in Figure 2.3, allow the inputs/outputs of a circuit to be set/observed when such pins are not directly accessible. By inserting these test wrappers, testers no longer need to manually move circut inputs and outputs throughout an IC, but instead can directly set and observe them. Although it takes clock cycles to serially set the inputs and outputs of a test wrapper, the number of clock cycles is often far less compared to manually moving signals through the IC [20]. Another form of test-specific hardware, also shown in Figure 2.3, are scan-chains [21]. Scan-chains are normal flip-flops which are connected serially and can be serially load/set from an outside source. Such scan-chains have become so common that they are frequently used for non-test purposes, most commonly in the form of Joint Test Action Group (JTAG) [22]. While testing, scan-chains allow flip-flops to be set and observed, which makes the applying of tests and the observation of their result simpler. The alternative to using scan chains is to run "initialization sequences" to set memory elements in a circuit [23], which not only can take several clock cycles to set, but also require the inputs of the circuit to be set as well, which in turn can take several clock cycles to accomplish.

Another common test hardware implementation found in many modern ICs is built-in self-test (BIST) [24], which allows for a circuit to be tested without applying any inputs or observing any outputs. BIST hardware is often implemented in the form of a read-only memory to hold test vectors or a shift register to generate vectors, and a compactor to observe the output of the test. The benefit of BIST is that the complexity of test is greatly reduced, as using BIST often requires no more than starting the BIST hardware and observing its final response. However, BIST is not without faults, as BIST requires extra hardware to implement, which may not be available during design, and this hardware is of little (if any) value during normal system operation. Also, the fault coverage of generated BIST vectors is rarely one-hundred percent, which implies ATPG tests vectors are still required to find reaming faults, albeit far less than without BIST.

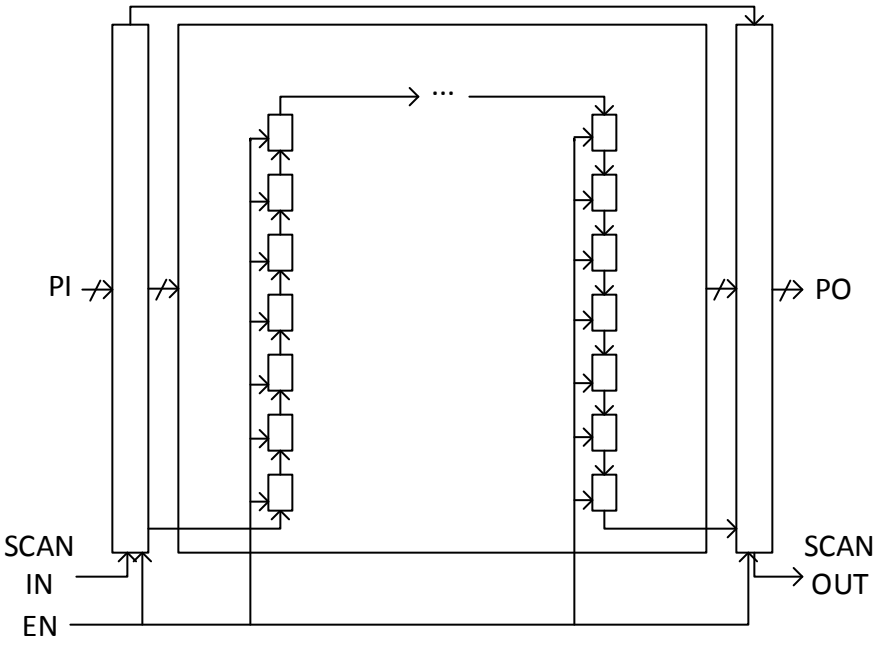

Figure 2.3 An example of test wrapper and scan chain implementation.

2.5 The Increasing Complexity of IC Test

Although before mentioned test reduction methods are useful for reducing the TAT for smaller circuits, the complexity of effectively deploying such strategies increases with larger modern circuits to the point of becoming impractical. The computational complexity of using the above methods is generally proportionally to the number of inputs to a circuit, or more specifically, it is $O(2^I)$ for ATPG algorithms [15]. This computational complexity is impractical as the number of inputs of a circuit grows.

To deal with larger modern IC designs, new design techniques have been developed which have not only assisted hardware designers, but will also assist in the reduction of TAT while still keeping fault coverage high. These new design techniques divide a circuit into smaller components for which traditional TAT reduction strategies are still effective, and then reassembles these components to make a complete, fully tested circuit. More on this process is explained in Chapter 3.

Chapter 3

SoC Design and Test

3.1 SoC Design Model

With the ever-increasing complexity of microprocessor ICs, certain design techniques have become commonplace, most notably dividing large designs into to smaller parts. With the initial introduction of silicon ICs, transistor density was only high enough to fit a few logic gates onto a single die. However, as IC feature sizes decreased, it became possible not just to fit a whole microprocessor onto a die, but also several related components onto the same die. This new style of design became known as a "system-on-chip", as what was once an entire system of interconnected components was now integrated onto a single chip. An example of a modern SoC is shown in Figure 3.1. Although there are no advantages to using an SoC from the consumer's point of view (as opposed to a non-SoC), there are several advantages to the developer and tester. These advantages will be discussed in this Chapter.

The most straightforward advantage in the development of SoCs compared to non-SoCs is a decrease in design complexity. When any design is proposed, from the largest of microprocessors to the smallest of single-gate circuits, the development time is proportional to the complexity of the design. Therefore, when large complicated designs are proposed, it is impractical to develop the entire design as a single entity. Instead, it is natural to divide the larger design into several smaller designs, with each smaller design having a distinct purpose. This way, the smaller designs can be independently developed and verified, and then after smaller parts are designed, they can be interconnected with glue logic. Although in theory such a method can only lead to an inferior

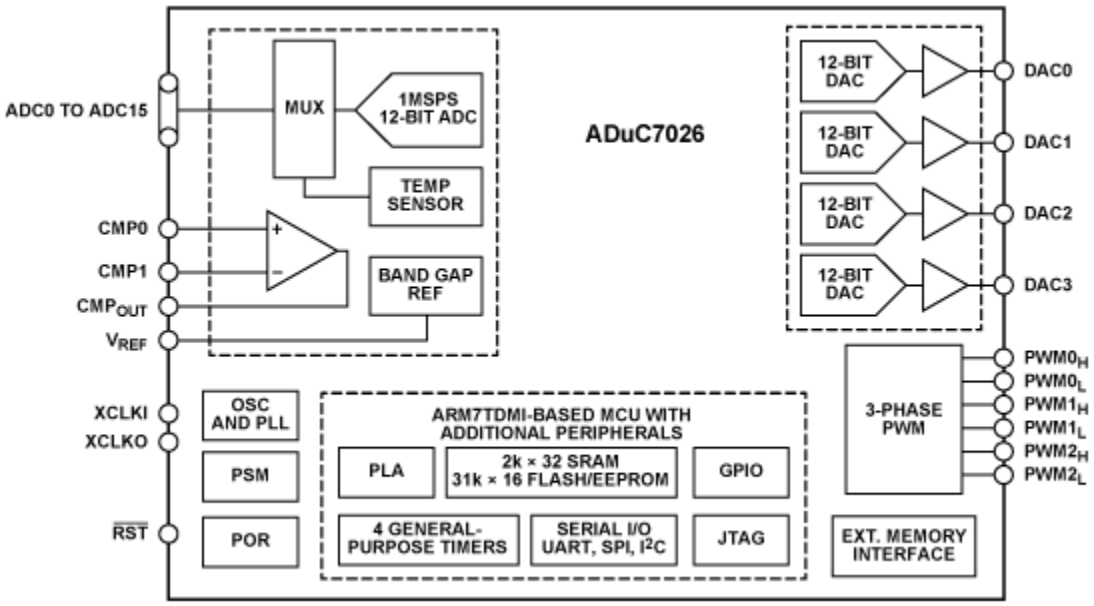

Figure 3.1 Example of a modern SoC (©Analog Devices)

design since the smaller designs cannot optimize their glue logic, in practice this method of design is worthwhile since the time to optimize such a large design can be impractically large.

A unique and common feature of a SoC designs is the integration of reusable "third-party" cores for the sake of faster and more cost-efficient development. With SoC designs becoming more commonplace, it is expected to see similar features recurring across multiple designs, which is a feature that can be exploited to create designs more quickly through core reuse. For instance, if a SoC designer has made a design in the past which shares a component required by a new design, then the new design can use a "copy" of the old component instead of creating it again from scratch. This situation has become so commonplace that entire design firms specialize in the creation of intellectual property (IP) cores for the sole purpose of selling these IP cores to other SoC designers. This design model is a benefit to both parties, since the SoC designer need not design every core, while the IP core designer benefits from reselling their designs to multiple parties.

Although the integration of IP cores has inherent advantages, there exist design and security issues that need to be resolved before a SoC can be brought to market. Although it can be presumed that an IP core will function according to its specification, the SoC designer still needs to

confirm that the glue logic used to implement the core will create a functional SoC. Confirming the SoC design to be correct requires functional simulation, which requires knowing how every core is implemented. However, IP core designers need to be secretive about the implementation of their core, else their business model of reselling cores may not function, since if the implementation of an IP core is leaked, then one can implement the IP core without compensating the IP core designer. Although there are different levels of trust between IP core developers and users (depending on company reputation and legal authority), an IP core designer would desire to release as little information as possible. Although the actual implementation of a core may not be necessary to implement a design (e.g., a functional description may be sufficient for many designs), other design issues, like generating tests for core, require the structure of the circuit. To resolve this, a new SoC testing model was developed, as will be discussed in Section 3.2.

3.2 SoC Testing Model

Since an IP core's implementation is often hidden from the SoC designer, the SoC designer is often incapable of developing tests for the IP core. Normally, a SoC designer would create tests for each core using traditional test generation methods [15], but doing so requires knowing the structure (and faults) of the circuit. Since the SoC designer is not given the structure of the IP core for reasons of security, it is impossible to generate tests for the core and thereby adequately test their SoC. Although the SoC designer can still apply functional tests to the core, such functional tests will rarely thoroughly test the core for reasons explained in Section 2.2.

To cope with the lack of knowledge of core structure, IP core vendors will often provide a series of test inputs (and corresponding non-faulty outputs) which are guaranteed to test for all possible defects and/or faults in the core, thereby revealing the SoC designer from generating their own tests. Although the SoC designer will not have access to the required structure to generate tests, the IP core designer clearly does. Since it is in the IP core designer's best interest to provide the highest quality core, the IP core designer will generate a series of tests for the core with the guarantee that applying all the tests will thoroughly test the core. The IP core designer will also

attempt to reduce the number of test vectors for the core as much as possible, as well as insert test-specific hardware to reduce the TAT of the core using methods described in Section 2.4.

When cores with such proprietary tests are given to a SoC designer, the designer's goal then becomes to apply all such tests in the shortest possible period of time. Whether a design is a traditional or a SoC design, the goal of test is still to test for as many faults as possible in the shortest possible period of time. When predetermined core tests are given to the tester, the task is no longer one of generating the actual tests, but instead to apply the given tests while still reducing test time. Although such a task appears simpler than reducing TAT through traditional methods, such a task can still be met with several constraints, as will be explained in Section 3.3.

Even when the cores in a SoC are not proprietary, the SoC testing model is still useful in reducing test time. As suggested in Section 2.5, applying traditional TAT reduction methods to entire, non-segmented designs can be computationally infeasible. However, it is still feasible to apply traditional TAT reduction methods to smaller parts. By dividing a large design into smaller parts, generating tests for the smaller parts, and then applying the tests using the SoC testing model, reduced TAT can be achieved without using IP cores.

3.3 SoC Test Constraints

When scheduling all tests for a SoC, certain constraints must be met to ensure that the created test schedule is valid. For a SoC, applying all tests for all cores at the same time is often not feasible due to shared hardware or other technology-dependent constraints. A straightforward example of this is if a given core has more than one test, both cannot be applied at the same time. However, other complex constraints arise due to the presence of the glue logic which connects individual cores in the SoC. Also, as IC feature sizes scale downwards, new constraints are being introduced which were not a concern in earlier technologies. This section addresses these constraints.

One common constraint in SoC testing are hardware constraints created from the SoC design. As shown in Figure 3.1, modern SoC designs are composed of many different cores which are interconnected using glue logic. Because these cores interact with each other, applying tests to one core can result in undesired signals being propagated to another core. For example, if a memory is

being tested by writing values to it, then a CPU core which reads values from the memory can read unexpected values durings its test. For this reason, the simultaneous testing of cores which share hardware is often prohibited.

Along with general hardware, test-specific hardware is often limited for testing, and can therefore become another form of hardware constraint. Section 2.4 introduced several different kinds of test-specific hardware used for the purpose of reducing TAT. However, designers will often limit the amount of these hardware resources since these resources take up extra space on the die. Thankfully, unlike regular hardware, the tester often has the ability to allocate testing hardware after all other design decisions have been made, as long as the hardware requests of the tester are within the constraints set by the designer. By making choices in the testing hardware, large changes in TAT can be achieved, so the goal of a tester with a test hardware constraint is to minimize the TAT while not violating the hardware constraint given by the hardware designers.

The most common form of a SoC test-specific hardware constraint is a test access mechanism (TAM) pin constraint [25, 26, 27]. TAM pins are test-specific external pins on a die used to access the test wrappers, which in turn are used to apply tests to individual cores. For cost reasons, hardware designers want to limit the number of external pins a given design has, and therefore allocating TAM pins is often an afterthought when creating their designs. This is unfortunate since an increase in the number of TAM pins can lead to a dramatic decrease in TAT for two reasons. First, applying a test to a core requires at least two test pins to carry data (one in, one out), along with extra pins to control the flow of test data. The more test pins that are available during test, the more cores of an SoC that can be tested in parallel, and therefore the less time is required to test the SoC. Second, when scan-chain flip-flops are required to apply tests to a core (which is often the case), the time it takes to set flip-flops in a circuit is proportional to the longest scan chain [8]. Allowing more TAM pins to access a single core can allow for a single chain to be divided into several smaller scan chains, thereby reducing the longest scan chain in the core, thereby reducing the TAT of the tests of that core.

As the feature sizes of devices under test have become smaller, the increased number of transistors per unit area have introduced new constraints, such as power during test. Although Moore's

Law originally assumed that the power per transistor would stay constant [1], this assumption is not holding to be true. Instead, the power density of ICs during normal operation has increased more than expected. In the past, the power dissipation of ICs has been small enough that power issues could practically be ignored (and in some cases, seen as a benefit when the heat from electrical components was desired), but with increasing power density, power during SoC test has become yet another problem to address.

The reason high power dissipation during test can be problematic is that it can cause damage to the device under test, or it can create voltage sag induced false failure(s) [28], both of which decrease device yield. Damage from high power occurs when so much power is dissipated in a small area that local "hot spots" are created. These local "hot spots" can have high enough temperature to cause silicon drift and other chemical changes which change the function of the device. The other problem that can occur from high power dissipation is a "voltage sag", which occurs when the power demand of a SoC is so high that the power supply (or power distribution network) cannot meet the demand. When this occurs, the device will not be damaged, but instead will temporary not operate correctly since the required voltage is not available to operate. Both of these problems reduce device yield, and therefore increase manufacturing costs by discarding of good devices. The result of all of this is that power consumption during test has become a new constraint during testing.

With power-constrained testing, not only must hardware constraints be met, but also a given maximum power value, set either based on the power supply used or by the properties of the device, must not be exceeded. Like hardware-constrained testing, scheduling tests to meet a power constraint is an NP-hard problem, but the computational complexity grows significantly compared to hardware-constrained test scheduling. Like hardware-constrained testing, several methods have been proposed to solve power-constrained testing, such as graph formulations [29], mixed-integer linear programming (MILP) formulations [28], and various other algorithms [30, 31, 32], each providing a trade-off between test schedule quality and test schedule computation time.

During test, the power density of ICs is further increased not only by the use of test-specific hardware not utilized during normal operation, but also due to the higher switching activity of

circuits from time-saving tests [32, 33]. As discussed previously, there are many hardware components which are test-specific and therefore are not utilized during the normal operation of a circuit. Because these hardware modules are test-specific, the power dissipation during test will be higher than during normal operation due to extra hardware being utilized. Also, using TAT reduction techniques increases the switching activity of a SoC to a level which would normally never occur, thereby increasing power dissipation even more. Not only is it normal to see switching activity at a much higher rate than during normal operation due to test compaction, but it is also possible to see hardware modules operate simultaneously when they normally would not.

As the power during test increases, so does the temperature during test, which like power, has previously been of no consequence. The temperature of an IC is a function of the power density of an IC, with higher power density leading to higher device temperature, and as the power density of an IC has increased, so has the temperature. Also like power, a high temperature can lead to device destruction through the changing of IC chemical properties. However, unlike power, the temperature of a device is also a function of time, with higher temperature "accumulating" with time from high power and low temperature "dissipating" with time from low power. This modeling of temperature from power will lead to scheduling and modeling challenges, as will be seen in Chapter 6.

Temperature during test is especially problematic due to the unique environment of test. As stated before, the power dissipation during test is higher than during normal operation, which in turn leads to higher temperature. However, the physical environment of test is also unique, which can lead to more temperature problems during test. For instance, manufacturers are reluctant to attach a heat-sink to a device that is destined to be faulty and discarded [34], and therefore temperature will increase faster during test than during normal operation.

Future chapters will focus on how to meet these constraints while reducing TAT.

Chapter 4

Utilizing DVFS For Power-Constrained SoC Test Scheduling

4.1 Introduction

As summarized in Chapter 3, the problem of SoCs test scheduling is becoming more complex with decreasing IC feature sizes. SoC test scheduling started as a problem of enforcing set hardware constraints between modules [35, 36, 24]. Although scheduling with hardware constraints is an NP-hard problem, for small number of cores and tests it can be solved efficiently and optimally. However, with decreasing IC feature sizes came new constraints need to be enforced and the size of the problem also grew. Enforcing new power constraints is especially problematic during test due to higher dissipation from test-specific issues [32, 33]. Like hardware-constrained testing, scheduling under a power constraint is an NP-hard problem, but the computational complexity grows significantly compared to hardware-constrained test scheduling. Also like hardware-constrained testing, several methods have been proposed to solve power-constrained testing, such as graph formulations [29], MILP formulations [28], and various other algorithms [30, 37, 31, 32].

A new technology which can provide potentially better schedules under power constraints is DVFS. This technology has been used in recent designs to allow for processors to save power and energy during less demanding intervals of operations by decreasing their operating frequency and voltage supply [38]. This decrease in voltage and frequency effectively slows down the operation of a given core or module while at the same time lowering its power consumption. DVFS technology can be used to scale the power and TAT of individual tests to find better and more compact test schedules. Test scheduling using DVFS has been studied in the recent past[39, 40, 41, 42], but the methods proposed in literature have either ignored the power during test aspect, or the solutions

proposed have been less than optimal by relying on session-based formulations, using heuristic methods, and enforcing other unnecessary constraints. This study address the power issue explicitly, with its main contributions as follows:

- An optimal formulation for DVFS SoC testing under power and hardware constraints is given, for both session-based and sessionless environments.
- It is shown that DVFS scheduling can provide better and more compact test schedules for power constrained testing compared to schedules without utilizing DVFS technology.
- It is shown that sessionless formulations generate superior test schedules compared to alternate session-based solutions without excessive computation overhead.

The remainder of this chapter is organized as follows: Section 4.2 gives a brief history of power-constrained and DVFS testing. Section 4.3 gives a session-based formulation for DVFS, while Section 4.4 gives a sessionless formulation. Section 4.5 gives the experimental setup used to evaluate the effectiveness of the two formulations, while Section 4.6 provides the results for various benchmark SoCs, and the chapter concludes with Section 4.7. The majority of the content of this chapter is a verbatim collection of previously published work from [43].

4.2 Past Work

4.2.1 Session Based & Sessionless Test Scheduling

After the introduction of hardware-constrained test scheduling [35], the first problem to be addressed was the inefficiency of session-based testing while scheduling under hardware constraints [44]. Initial test-scheduling formulations were for session-based testing due to their simplicity [35]. Scheduling tests in a session-based manner can be viewed as a bin selection problem, where each test must be assigned to a bin, but no two tests can be assigned to the same bin if they are incompatible (due to sharing of resources, such as common hardware). After each test has been assigned to a bin, the TAT of each bin is the TAT of the longest test in the bin, and the TAT of the schedule is the sum of the TAT of each bin.

However, a problem with session-based formulations is that they lack the ability to exploit potential overlaps. When tests are assigned to sessions, it is impossible to overlap tests with any other test outside of the session, even if those tests are compatible. This occurs because a session may have many tests and some of these tests may be incompatible with the desired overlapping test(s) in another session. If such tests can be overlapped, then the TAT can be reduced (see Figure 4.1).

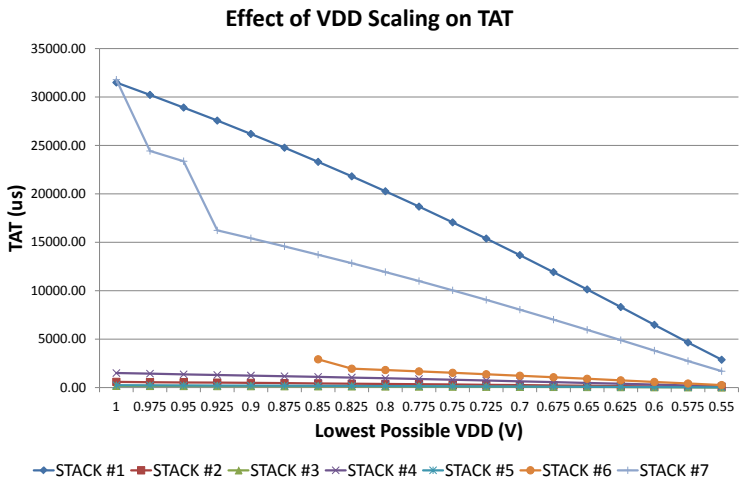

Figure 4.1 Presuming that test 2 and 5 are compatible, test time can be reduced by eliminating session requirements.

Formulations for sessionless test scheduling have attempted to remedy the overlapping restraints of session-based formulations as follows. For any sessionless scheduling method, tests must be able to be scheduled arbitrarily in time. This requirement inherently increases the complexity of sessionless scheduling methods. Solutions for sessionless scheduling often make use of heuristics, and a common heuristic makes use of list-based algorithms [45]. However, there have been instances of using deterministic MILP-style formulations for hardware-constrained test scheduling [46].

Although sessionless formulations are successful in finding shorter test schedules, they do so at the expense of longer computation times or heuristic based results providing no guarantee about the quality of the solution. Since sessionless formulations require tests to be scheduled arbitrarily in time, more variables and constraints need to be enforced to implement such a requirement. Some scheduling methods, such as list-based algorithms, are not deterministic and therefore the quality of a schedule depends on the order of the tests in the list [45]. If a good list is given, a good test schedule will be found, but if a bad list is given, the schedule found may be worse than a good session-based schedule.

4.2.2 Power & DFVS Test Scheduling

One of the first studies into power-constrained test scheduling was done by Chou et al. [29]. In this study, the complexities of power-constrained test scheduling were first observed as an extension to hardware-constrained test scheduling. The scheduling problem was viewed as a hardware-constrained test scheduling problem with a test or a set of tests being designated as incompatible with another test or tests, except a new power constraint was also enforced. This alone implied that power-constrained test scheduling is more complex than hardware-constrained test scheduling. The approach used to solve the test scheduling problem was a graph-based approach, in which all possible "cliques" and their "subcliques" of a graph are found, with each clique or subclique corresponding to a test session. This approach relied on an earlier implementation with hardware-constrained test scheduling [35], where each test in a clique must be hardware compatible. However, if power constraints are enforced then an additional requirement is that the sum of power from each test in a clique or subclique must also be lower than the given power constraint. This graph-based approach has effectively been implemented in MILP formulations in [41, 42].

As power during test has become larger due to smaller IC feature sizes, DVFS has been explored as a new technique to achieve better test schedules without violating power constraints. The original purpose of DVFS was to decrease the operating voltage and frequency of a module or core when the workload on it decreased so as to save power without hindering performance[38]. However, DVFS can also be used to slow down or speed up individual tests during scheduling to

allow for better test compaction under power constraints. If the operating voltage or frequency is lowered for a test, then the test will take longer to complete, but at the same time the test will consume less power. This can allow for two tests that normally could not run in parallel due to their combined power being too high to be made compatible by lower operating voltage or frequency for one or more of the tests. Likewise, a test that must be run individually can increase its operating voltage and frequency to decrease the time needed to apply the test at the expense of increased power consumption.

Initial work on DVFS test scheduling focused on scheduling all possible voltage or frequency combinations of a given core, thereby thoroughly testing a device [39]. Such studies take the form of a hardware-constrained test scheduling problem, since the goal of such studies is to test all possible DVFS combinations for each IC. However, such test models presume tests need to be applied at multiple VDD values, which is not the case for non-voltage dependent faults. Therefore, such studies have not considered that only a single voltage and frequency pair needs be scheduled for a given test, nor have they considered the power dissipated during test as a constraint.

Some more recent work focused on scheduling tests such that a test is scheduled with a single voltage and frequency pair under power constraints [41, 42], with the goal being to schedule each test at least once. Initial work focused on only frequency scaling [41], which was considered to be a simpler problem since power scales linearly with operating frequency. After frequency scaling alone was studied, voltage scaling was added [42]. Both cases showed that allowing for voltage or frequency scaling would result into significantly better test schedules with power constraints. However, these studies not only presumed that scheduling was done in sessions, but they also presumed that for any given session the voltage and frequency pair for every test in the session must be the same, which can lead to inferior results.

4.3 Session Based Testing

The first formulation presented in this chapter is for session-based testing of SoCs. This formulation is presented for the purpose of gaining an understanding of DVFS scheduling, as well as to later gain a perspective on the deficiencies of session-based formulations.

4.3.1 Test Environment and modeling

Before a formulation is given, some assumptions have to be made about the environment in which tests are scheduled. As shown in Section 4.2, different models exist describing how DVFS is used during test. Section 4.7 will discuss what effect other models will have on this study and how the proposed formulations can be extended to fit other models.

The first assumption is that the power dissipated by a test is constant throughout the entire test. This assumption is made so that the power values of a test can be directly used in an MILP-style formulation, for changing power values during test drastically increases the complexity of any formulation. This assumption, that the power during test is constant, has been made by the majority of formulations in the past due its simplicity. For tests that do not have constant power dissipation, the power dissipation can be presumed to be the maximum power dissipation of a test. Although it is possible to achieve better test schedules if power is not assumed to be constant by exploiting more opportunities for overlap, it is presumed that this overlap is small relative to any scheduling benefit that can be achieved through other scheduling methods. In the case where 2 or more cores share DVFS hardware, specific constraints can be added to enforce incompatibility between tests being applied at different voltage and frequency values.

It is also assumed that the V_{DD} and frequency of any test can be independently controlled at any time, regardless of what session it is scheduled in or what other tests it overlaps with. Presuming such will distinguish this formulation from the one presented in [42]. In [42], it was assumed that every test in a session must have the same V_{DD} and frequency applied to it. By eliminating this assumption, better schedules can be obtained by allowing for better compaction of tests within a session.

In this chapter it is assumed that the time overhead required to change the operating V_{DD} and frequency of a core is negligible. If such is not the case, overhead can be modeled by increasing test time corresponding to DVFS switching overhead.

The last aspect of modeling is how operating V_{DD} and frequency is chosen for test scheduling. An initial assumption is that during normal V_{DD} operation (in this study, $1V$), the operating frequency is chosen so as to not violate timing constraints. This means the operational frequency

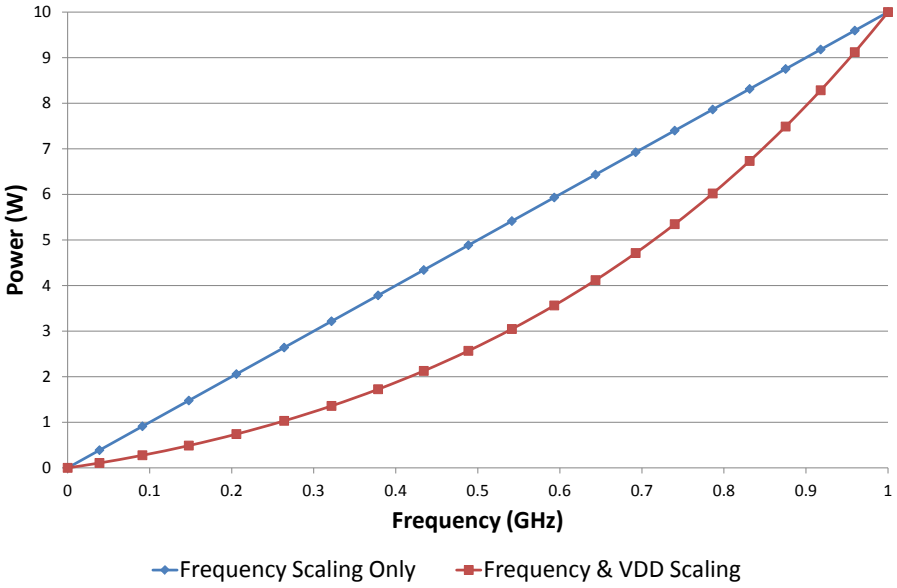

Figure 4.2 By scaling V_{DD} with frequency as opposed to frequency alone, better power reduction can be achieved.

of a test can be increased (therefore decreasing the length of the test) only if the operating voltage is increased. As well, if the frequency of a test is decreased (test length increased), the operating voltage may be decreased since there is no requirement to have a higher voltage drive the lower operating frequency.

Given this assumption, every V_{DD} value should have only the highest possible frequency value assigned to it, since assigning any lower frequency value would only waste power and increase test time or both. The same can be said of assigning V_{DD} values to frequency values, but in this study pairs are chosen based on V_{DD} since it is presumed that frequency can be scaled more finely, whereas V_{DD} can only be scaled by set amounts. From this, for a given V_{DD} value, the frequency can be set according to the alpha power law [6, 38], where f_s is the factor by which frequency is scaled:

$$f_s \propto \frac{(V_{DD} - V_{TH})^{\alpha}}{V_{DD}} \tag{4.1}$$

The motivation for scaling V_{DD} with frequency as opposed to scaling frequency alone is that scaling V_{DD} along with frequency allows for less power to be consumed. If scaling frequency is done alone, dynamic power consumption is scaled by the same factor. However, scaling to a lower

frequency allows for V_{DD} to be scaled lower as well. By scaling V_{DD} to the lowest possible value for a given frequency, the relation between frequency and power is no longer linear, but is instead cubic, since the relation between dynamic power consumption and DVFS is $P \propto F \cdot V_{DD}^2$. This concept is illustrated in Figure 4.2, which presumes the assumptions stated below with a normal operation frequency of $1GHz$ and normal power consumption of $10W$.

In this chapter, it is assumed that modern short-channel MOSFET technology is used, with $V_{DD} = 1V$, $V_{TH} = 0.5V$, and $\alpha = 1.3$.

4.3.2 Formulation

The formulation for both session-based and sessionless scheduling is given in the form of a MILP formulation. This style of formulation was chosen due to its deterministic nature. Below we give a detailed formulation of the session based scheduling problem.

The first constraint of any scheduling formulation is to define the objective, and the objective in this case is to minimize the total TAT of the schedule. Since the constraint must be made linear, this can be done by forcing the overall TAT to be greater than or equal to the finishing time of each session, $t_f(s)$.

$$\text{minimize } t_{finish}$$

$$\text{subject to...}$$

$$\forall s \in S : t_{finish} \geq t_f(s)$$

The start time and finish time of each session, $t_s(s)$ and $t_f(s)$, is dependent on the TAT length of the session, $L(s)$. However, the start time of each session is guaranteed to be the finish time of the previous session, since doing otherwise would only waste time. Note that the maximum possible number of sessions is equal to the number of tests, since the worst-case schedule would have every test assigned to its own session.

$$\forall s \in S : t_f(s) = t_s(s) + L(s)$$

$$\forall s = 2 \ldots |S| : t_s(s) = t_f(s-1)$$

$$t_s(1) = 0;$$

Every test must now be assigned to a session. A new binary variable is created for the purpose of assigning tests (t) to sessions.

$$\Delta(t,s) = \begin{cases} 1 & \text{if the test } t \text{ is assigned to session } s \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t \in T : \sum_{\forall s \in S} \Delta(t,s) = 1$$

If two tests are incompatible due to hardware constraints, then it is impossible for them to be scheduled in the same session.

$$\Gamma(t_1,t_2) = \begin{cases} 1 & \text{if tests } t_1 \text{ and } t_2 \text{ are incompatible} \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t_1 \in T, \forall t_2 \in T, \forall s \in S : \Delta(t_1,s) + \Delta(t_2,s) \leq 2 - \Gamma(t_1,t_2)$$

Every test must be assigned to one and only one V_{DD}/frequency pair, assuming there are N V_{DD}/frequency selections available to choose from for each test.

$$\Psi(t,n) = \begin{cases} 1 & \text{if the test } t \text{ is assigned to} \\ & V_{DD}\text{/frequency pair } n \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t \in T : \sum_{\forall n = 1 \ldots N} \Psi(t,n) = 1$$

The next variable defined, $\Theta(t,s,n)$, is for the purpose of stating the TAT and power of each session in a linear formulation.

$$\Theta(t,s,n) = \begin{cases} 1 & \text{if the test } t \text{ is assigned to session } s \\ & \text{and assigned to } V_{DD}\text{/frequency pair } n \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t \in T, \forall s \in S, \forall n \in N : \Theta(t,s,n) \geq -1 + \Delta(t,s) + \Psi(t,n)$$

$$\forall t \in T, \forall s \in S, \forall n \in N : \Theta(t,s,n) \leq \Delta(t,s)$$

$$\forall t \in T, \forall s \in S, \forall n \in N : \Theta(t,s,n) \leq \Psi(t,n)$$

The TAT of each session is defined as the longest time amongst all tests within the session. Since the number of sessions is set to the number of tests, some sessions may be unused. If such is the case, the TAT of each session must be at least 0. Here, $L_{t,n}$ is the TAT of a test t if the n^{th} V_{DD}/frequency pair is chosen for it.

$$\forall s \in S : L(s) \geq 0$$

$$\forall s \in S, \forall t \in T, \forall n \in N : L(s) \geq L_{t,n}\Theta(t,s,n)$$

To satisfy a given power constraint, the sum of power from all tests in a session must be lower than the given power bound in every session. Here, $P_{t,n}$ is the power of a test t if the n^{th} VDD/frequency pair is chosen for it.

$$\forall s \in S : \sum_{\forall t \in T} \sum_{\forall n \in N} P_{t,n}\Theta(t,s,n) \leq P_{BOUND}$$

4.4 Sessionless Test Scheduling

As was stated in Section 4.2.1, session-based formulations will often fail to find a low test time solution due to their restricted nature. It is easy to see that any possible session-constrained schedule can be converted to a sessionless schedule in the sense that no resource and power constraints are violated, but not every sessionless schedule can be stated as a session-based schedule. Therefore, if there is any test set which has a better sessionless schedule than an optimal session-based formulation, then sessionless formulations are guaranteed to find a schedule no worse (but often better) than a session-based formulation. Such a schedule has already been presented in Figure 4.1.

Unlike with session-based test scheduling, tests can be scheduled arbitrarily in time as long as no two overlapping tests have a hardware constraint. Because of this, timing constraints need to be redefined. First, test application time is no longer based on sessions, but based on individual tests.

$$\text{minimize } t_{finish} \tag{4.2}$$

subject to...

$$\forall t \in T : t_{finish} \geq t_f(t) \tag{4.3}$$

$$\forall t \in T : t_f(t) = t_s(t) + L(t) \tag{4.4}$$

$$\forall t \in T : t_s(t) \geq 0 \tag{4.5}$$

The TAT of each test, $L(t)$, is dependent on the DVFS pair chosen for that test. The variable $\Psi(t, n)$ can be kept in use for this purpose.

$$\forall t \in T : L(t) = \sum_{\forall n \in N} L_{t,n} \Psi(t, n) \tag{4.6}$$

To keep tests from overlapping that have a hardware incompatibility between them, the constant $\Gamma(t_1, t_2)$ can be reused. However, the fashion in which the constant is used must be changed, since tests are no longer assigned to sessions, but instead can overlap arbitrarily. To define how tests can overlap, the method from [46] is borrowed to accomplish this. Here, λ is defined as the longest possible test schedule, which is the sum of run times of all tests at their slowest frequency. Setting λ to such or higher value allows for the arbitrary scheduling of the start time of any test from time 0 to $\lambda - L(t)$ while defining overlapping tests.

$$\eta(t_1, t_2) = \begin{cases} 1 & \text{if the test } t_1 \text{ finishes} \\ & \text{before the test } t_2 \text{ begins} \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

$$\forall t_1 \in T, \forall t_2 \in T : t_f(t_1) \leq t_s(t_2) + (1 - \eta(t_1, t_2)) \cdot \lambda \tag{4.8}$$

$$\forall t_1 \in T, \forall t_2 \in T : t_s(t_s) \leq t_f(t_1) + \eta(t_1, t_2) \cdot \lambda \tag{4.9}$$

Two tests t_1 and t_2 are overlapping if both $\eta(t_1, t_2)$ and $\eta(t_2, t_1) = 0$. Using this, hardware compatibility can be enforced.

$$\forall t_1 \in T, \forall t_2 \in T : \eta(t_1, t_2) + \eta(t_2, t_1) \leq 1 \tag{4.10}$$

$$\forall t_1 \in T, \forall t_2 \in T : \eta(t_1, t_2) + \eta(t_2, t_1) \geq \Gamma(t_1, t_2) \tag{4.11}$$

To enforce a power constraint in a sessionless formulation, the sum of all power values for any combination of overlapping tests must not exceed a given bound.

$$\Omega(t_1, t_2) = \begin{cases} 1 & \text{if test } t_1 \text{ overlaps with test } t_2 \\ 0 & \text{otherwise} \end{cases} \tag{4.12}$$

$$\forall t_1 \in T, \forall t_2 \in T : \Omega(t_1, t_2) = 1 - (\eta(t_1, t_2) + \eta(t_2, t_1)) \tag{4.13}$$

The variable $\Theta(t_1, t_2, n)$ is redefined to allow the power constraint to be stated linearly. Its formulation is nearly identical to the $\Theta(t, s, n)$ formulation in Section 4.3.2, except $\Psi(t, n)$ and $\Omega(t_1, t_2)$ are used instead of $\Psi(t, n)$ and $\Delta(t, s)$.

$$\Theta(t_1, t_2, n) = \begin{cases} 1 & \text{if test } t_1 \text{ overlaps with test } t_2 \\ & \text{\& the } n^{th} \text{ VDD/frequency pair for} \\ & t_1 \text{ is chosen} \\ 0 & \text{otherwise} \end{cases} \tag{4.14}$$

$$\forall t_1 \in T : \sum_{\forall t_2 \in T} \sum_{\forall n \in N} \Theta(t_1, t_2, n) \cdot P_{t_3,n} \leq P_{BOUND} \tag{4.15}$$

4.5 Experimental Setup

4.5.1 Benchmarks

Benchmarks used in this study are based off the ITC'02 benchmarks [47]. These benchmarks represent SoC and provide test length information. Although they also provide module information

for the purpose of generating hardware compatibility, this study generates hardware compatibility using a randomized graph-based approach, since hardware compatibility is not the focus of this study [45]. Since the original benchmarks do not contain significant power information, power traces for individual modules and their tests are generated by simulating ITC'99 circuits [48] and assigning those power traces to modules by matching the size and complexity of the ITC'02 modules. Leakage power is not addressed in this study. However, if leakage power is modeled as constant power dissipation proportional to core area than this has the equivalent effect of increased dynamic power consumption by a constant factor proportional to V_{DD} which in turn can be incorporated in DVFS pairs. Specifics on the benchmarks can be found at [43]. These benchmarks represent 3D-stacked ICs, which in this study has the effect of increasing the number of modules for a given benchmark.

4.5.2 Implementation

The first experiment is done to evaluate the effectiveness of session-based scheduling as opposed to sessionless scheduling across a series of benchmarks. Information on the sixteen benchmarks used to evaluate the scheduling methods is provided in Table 4.1, which includes the number of tests (and modules, since ITC'02 benchmarks used have one test per module) in the benchmark and the maximum power consumed by any given test in the benchmark. The two results recorded for each scheduling method on each benchmark are the optimal scheduled TAT and the computation time required to obtain the optimal schedule.

The second experiment is done to evaluate the effectiveness of DVFS scheduling as more DVFS pairs become available to schedule. Several benchmarks are taken from the original set of sixteen benchmarks and are run with a varying number of DVFS pairs available for scheduling. This is done to observe the effect DVFS has on TAT.

All MILP formulations are implemented using IBM ILOG CPLEX, and they are run to completion to find an optimal solution for a given formulation. For all experiments, the maximum power bound (P_{BOUND}) is set to 20 Watts, which is purposely set lower than the max power of some benchmarks to show the requirement of DVFS in test scheduling in such benchmarks.

Table 4.1 Benchmark Information

Bench #	# Tests	Max Power (W)	Bench #	# Tests	Max Power (W)
1	9	10.9	9	29	11.6
2	15	11.6	10	25	11.6
3	10	4.2	11	24	5.4
4	8	14.2	12	12	34.5
5	14	5.4	13	39	11.6
6	4	34.5	14	33	34.5
7	4	13.1	15	22	34.5
8	7	13.4	16	47	14.2

4.6 Results

Table 4.2 shows the results of session-based and sessionless DVFS scheduling on sixteen benchmarks, presuming a normal operating V_{DD} value of $1V$ and a normal operating frequency of $120MHz$. For these benchmarks, four V_{DD} values were available for each test to choose from (1, 0.9, 0.8, and 0.7 volts). The frequency values corresponding to these V_{DD} values are obtained using the method described in Section 4.3.1.

The results in Table 4.2 clearly show that for any benchmark the schedule obtained by sessionless scheduling is always equal to or better than that achieved by session-based scheduling. The reason for this, as explained in Section 4.2.1, is that any session-based schedule is possible with sessionless scheduling, but certain sessionless schedules cannot be expressed as session-based schedules. However, there are cases when both scheduling methods obtain the same or nearly same total TAT. These instances appear more often for benchmarks with a lower number of tests to schedule. The more tests there are to schedule, the more possible schedules exist, which in turn means there are more schedules that sessionless scheduling can take advantage of that session-based scheduling cannot.

Although sessionless scheduling can clearly give better TAT results, the primary motivation behind session-based scheduling is the simplicity of implementation, and thus reduced design effort and hardware complexity. Also, it is intuitively expected that computation effort to find solution to session-based scheduling is lower than that for sessionless schedules. However, the results in Table

Table 4.2 Benchmark Results

Bench #	TAT (us)		Scheduling Runtime (s)	
	Sessionless	Session-based	Sessionless	Session-based
1	13672.39	13672.39	0.29	13.62
2	250.32	264.79	1.1	30.5
3	75.24	75.24	0.27	15.72
4	645.34	645.34	0.18	1.85
5	102.73	102.73	2.15	779.43
6	1219.79	1219.79	0.03	0.03
7	8048.03	8048.03	0.03	0.1
8	55143.92	60166.34	0.95	2.09
9	238.74	259.00	68.76	22689.2
10	238.74	263.34	47.71	2686.36
11	82.47	82.47	71.45	1031.55
12	1219.79	1234.26	0.93	3.47
13	238.74	260.45	216.81	31305.73
14	1222.68	1240.05	18553.6	2081.72
15	8048.03	8076.97	172.18	5843.37
16	436.98	455.79	2816.44	5387.26

4.2 show a contrary result. The computation time for all benchmarks except for one (Bench 14) favors sessionless scheduling. The reason for this anomaly is that both methods require the same number of variables to solve, even though the session based scheduling problem is apparently simpler. In sessionless scheduling, the primary variable in question is the start time of each test ($t_s(c)$), while in session-based scheduling the primary variable is which session a test belongs to (in this study, implemented using $\Delta(c, s)$). From the point of view of a MILP solver, session-based scheduling is actually more complex since the "primary variable" is implemented using several binary variables as opposed to a single integer variable. It may be possible to make the session-based formulation more efficient, but the point remains clear that a sessionless formulation is by no means overly complex to implement or time-consuming.

Figure 4.3 gives the TAT of sessionless formulations for selected benchmarks (selected for feasible computation time) as the allowed range of V_{DD} values increases. The value at the X-axis represents the lowest possible V_{DD} value allowed, with the all V_{DD} values allowed between that

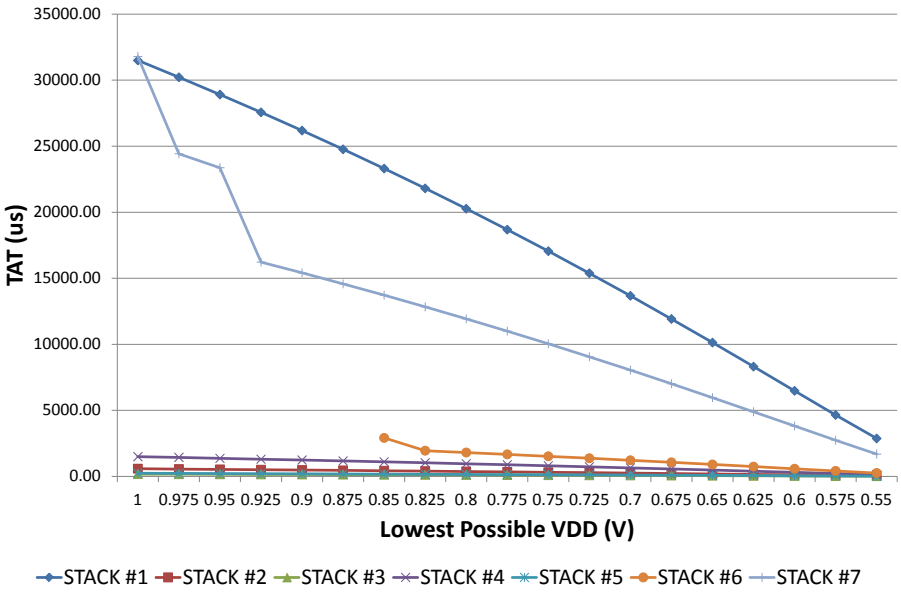

Figure 4.3 As more V_{DD} options become available, better schedules can be obtained.

value and $1V$ with a $0.025V$ granularity. For instance, for the $0.9V$ data point, the V_{DD} values available to schedule for each test is 1, 0.975, 0.95, 0.925, and $0.9V$.

One observation based on Figure 4.3 is that as lower V_{DD} values are allowed for scheduling, the TAT is greatly reduced. It is true that lowering V_{DD} values can allow for the overlapping of tests that was not possible before due to power constraints, but at the same time lowering V_{DD} values (and in turn lowering operating frequency) increases individual test lengths. This leads to the conclusion that the effect of reducing test power outweighs the effect of increasing test time due to greater test overlap potential.

A second observation from Figure 4.3 is that DVFS may make test scheduling possible for circuits that were not testable without DVFS. This is shown well with Stack #6, which is not schedulable until $0.85V$ V_{DD} is allowed. Although it is not wise to create a design with test power higher than a given limit, this may be the case with high-power designs, since test power is often higher than normal operating power [32].

Figure 4.4 gives the TAT of both session-based and sessionless formulations for Benchmark #8 as the allowed range of V_{DD} values increases. An observation based on Figure 4.4 is for any

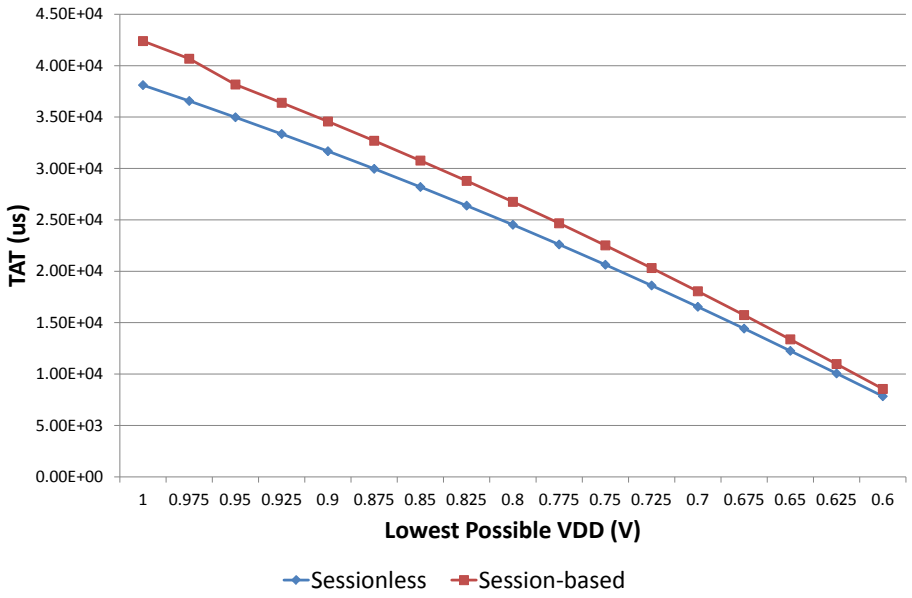

Figure 4.4 As more V_{DD} options become available, sessionless scheduling will still give better schedules than session-based schedules.

DVFS values available for scheduling, sessionless schedules always performs better than session-based scheduling. This is a trend observed in all stacks. This is confirmation of the statement made in Section 4.2.1, since adding DVFS generates more scheduling possibilities and sessionless scheduling can take more advantage of these scheduling options than session-based scheduling can.

4.7 Conclusions

Scheduling tests in a DVFS environment shows the potential for greatly reducing TAT for multi-core SoCs under power constraints. As voltages are scaled lower, individual test times are sacrificed for reduced power consumption. This trade off allows for greater test time compaction under power constraints, which in turn leads to better test economy.

The benefit of the TAT reduction achieved by the proposed formulations can be further applied to different DVFS test scheduling models. For models that presume tests must be scheduled for every DVFS voltage, each test is duplicated to represent running a test at a particular voltage

level and the variable determining operating V_{DD} will determine the operating frequency instead. To model the existence of voltage islands (multiple modules sharing the same voltage/frequency source), a new MILP constraint can be added to force the Ψ values of two tests to be equal if they are scheduled simultaneously and share the same voltage/frequency source. In both cases, the proposed formulations will achieve lower TAT compared to previous formulations, and sessionless scheduling will allow for lower TAT than session-based scheduling.

To take maximum advantage of DVFS scaling, the session-based test scheduling barrier must be broken. By breaking this barrier, better test schedules can be found, thereby reducing TAT and leading to better test economy. Although session-based scheduling has been previously implemented to reduce schedule computation time through simpler formulations, this study has shown that the computation time of sessionless formulations is by no means a hindrance to their implementation.

Chapter 5

Optimal Scheduling of Tests Under Various Constraints

5.1 Introduction

Although the study of hardware-constrained SoC test scheduling is in many ways "complete", new hardware technologies, like 3D-stacked integrated circuits (3D-ICs), are creating new testing constraints. In a 3D-IC, multiple silicon dies are stacked before being packaged into a complete design. This design method has several benefits, namely allowing a device to have a smaller footprint (a must in many different applications such as in mobile computing) and decreased die interconnect lengths, which can give faster device operating speeds. Although such designs are useful from a designer's point of view, they create difficulties for SoC testers, as they add new hardware constraints to the test scheduling process [49].

Although many past studies have addressed individual issues of SoC test schedules, few have incorporated multiple issues at once. Most previous SoC test scheduling studies have chosen to select only a single issue to remedy at a given time, only occasionally choosing to combine more than one issue. For instance, several studies address 3D-IC testing [17, 50, 49] and several address DVFS in test scheduling [41, 43, 42], but none (to the authors' knowledge) address both. Also, many previous test scheduling studies make unnecessary presumptions on the test scheduling environment, thus limiting their ability to find truly optimal test schedules.

By failing to incorporate several issues at once or by making unnecessary assumptions on the test scheduling process, all testing constraints will not be enforced, and not all methods of reducing TAT are being utilized. As the goal of SoC test scheduling is always to reduce TAT to its minimum in order to reduce testing costs while not providing an invalid test schedule, a test

scheduling method must be developed to do so. This chapter presents an optimal test scheduling formulation which allows for static and dynamic hardware constraints, power constraints, 3D-IC hardware constraints, and the use of DVFS during test.

The remainder of this chapter is organized as follows. Section 5.2 provides past work focusing on the constraints incorporated into SoC test scheduling. Section 5.3 provides an optimal SoC test scheduling formulation that addresses multiple scheduling constraints. Section 5.4 describes a series of experiments to evaluate the provided formulation, and Section 5.5 gives the results of these experiments. Section 5.6 gives a discussion about the incorporation of various other constraints in the formulation, and the chapter concludes with Section 5.7. The majority of the content of this chapter is a verbatim collection of previously published work from [51] AND [52].

5.2 Related Works

Although general hardware constraints were explored and enforced in Chapter 4, the enforcement of test-specific hardware constraints (and the control of such constraints by the tester) have yet to be explored. Studies in hardware-constrained SoC testing introduced the concept of generating hardware constraints during the testing process by giving control of test hardware allocation to the scheduler in the form of TAM pin allocations [20, 27]. Since resources like BIST or TAM pins are test-specific, it may be possible for the test scheduler to allocate these resources for achieving better TATs in SoC test schedules, as opposed to static hardware constraints which cannot be changed or modified. A SoC test scheduler can be given a constraint in the form of the total number of TAM pins allowed in the design, and the scheduler can then schedule tests while enforcing this constraint. In general, having more TAM pins available to the scheduler can greatly decrease the TAT [27], but there is often a constraint on the number of TAM pins allowed since allocating space for test-specific pins is expensive.

As new design methods were introduced, like 3D-ICs, hardware-constrained test scheduling formulations were expanded for these new environments. The act of testing a single die in a 3D-IC is practically the same as testing a non-3D-IC, as the pins needed to test the die are more or less available when the die is not stacked, but the testing of an entire stack must still be done to confirm

no faults are created during the stack process and to reduce overall testing costs [17]. In order to test entire stacks, test schedulers have implemented more complex TAM constraints based on the new hardware required to access TAM pins on stacked dies [53]. An example of scheduling tests using such new hardware is shown in [49].

Due to the complexity (NP-hard) of finding optimal test schedules, many SoC test scheduling methods are driven by heuristics. The main motivation behind using a heuristic, as opposed to an optimal solver, is the reduction in time to find a solution. Most SoC test scheduling heuristics take the form of a list-based algorithm, where the final schedule depends on the order of the input test list.

The scheduling method presented in this chapter will not only incorporate all the previous SoC constraints into a single optimal scheduling formulation, but will also overcome the sessions-based scheduling barrier to achieve higher quality schedules. Although many of the scheduling issues described here have been addressed individually, only some of these issues have been addresses together. Testing 3D-IC with power constraints has been addressed [50], testing under power constraints with TAM pin constraints has been addressed [37], but never have all of these issues been addressed at once, especially without making sub-optimal assumptions involving test sessions and heuristics. The formulation given in Section 5.3 will allow for optimal scheduling under all of these constraints.

5.3 Test Scheduling Formulation

The formulation presented in the section is a modification of the method presented in Section 4.4. More specifically, the formulation is modified for two reasons: 1) in order to correct flaws in its scheduling process, and 2) to incorporate new constraints.

Like the previous formulation, the goal of this formulation is to minimize the TAT of the schedule. For this reason, Equation (4.2) remains the goal. Also, Equations (4.3), (4.4), and (4.5) are kept in order to define the start and finish time of each test using its runtime.

Unlike the previous formulation, the runtime of each test is a function of two variables instead of one: what voltage/frequency a test is run at, and the number of TAM pins available to the test.

This is noted by the variable $\Psi(t, n, c)$, which replaces $\Psi(t, n)$ from Section 4.4. This formulation presumes that each test t has a given number of choices available as to what voltage/frequency it is executed at ($n \in N_t$) and how many pins can be assigned to the core ($c \in C_t$). Here, n and c are not the specific voltage/frequency and number of pins, but only an index (e.g., for a 6 voltage/frequency choices and 4 TAM pin amount choices, $n = 1 \ldots 6 = |N_t|$ and $c = 1 \ldots 4 = |C_t|$). The actual voltage/frequency corresponding to each n is irrelevant for scheduling, and the number of TAM pins associated with each pin count c is discussed later. Below, the constant $L_{t,n,c}$ denotes the runtime of test t when executed at DVFS index n and pin index c.

$$\Psi(t, n, c) = \begin{cases} 1 & \text{if the test } t \text{ is executed at} \\ & \text{frequency selection } n \text{ with pin} \\ & \text{selection index } c \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t \in T : \sum_{\forall n \in N_t} \sum_{\forall c \in C_t} \Psi(t, n, c) = 1 \tag{5.1}$$

$$\forall t \in T : L(t) = \sum_{\forall n \in N_t} \sum_{\forall c \in C_t} \Psi(t, n, c) \cdot L_{t,n,c}$$

To determine if two or more tests are overlapping, Equations (4.7), (4.8), (4.9), (4.10), (4.12), and (4.13) are borrowed from Section 4.4. These equations are used to create $\Omega(t_1, t_2)$.

Although the formulation presented in Section 4.4 will not create a session-based schedule, the scheduling formulation was not truly sessionless. An example of where the formulation from Section 4.4 will give a flawed result is shown in Figure 5.1. In this example, it is clear that the maximum power of this schedule is 15 W. However, the formulation from Section 4.4 defines the maximum power as "the sum of all power from all tests which overlap with any given test". Since there are 4 tests which overlap with the bottom-left test, the total power will be determined as 25 W, which implies that the given schedule is invalid under a 15 W power bound. The next equations presented here will remedy this.

To enforce power and TAM pin constraints, it is useful to note that the enforcement of such constraints is only necessary when a test begins. Since the number of TAM pins cannot increase

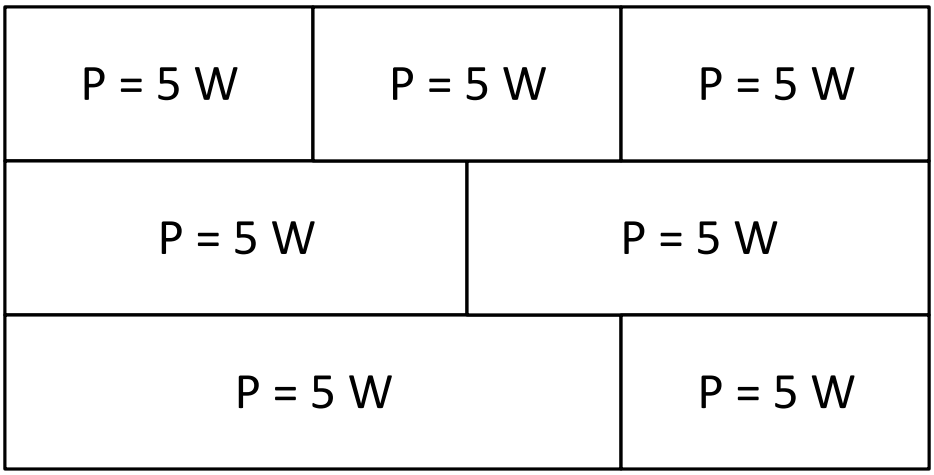

Figure 5.1 A true maximum power of this schedule is 15 W, but Section 4.4 will record it as 25 W.

when a test is in execution, and neither can the power consumed by a test (presuming a constant power model or maximum power model), checking a power or TAM pin constraint during test execution is unnecessary since this constraint cannot change. The checking of these constraints when a test finishes is also not necessary, since the power usage and TAM pin usage can only decrease under the same presumptions. To assist in the checking of constraints only when tests begin, a new binary variable "$\pi(t_1, t_2)$" is introduced.

$$\pi(t_1, t_2) = \begin{cases} 1 & \text{if the test } t_1 \text{ begins before} \\ & \text{or when the test } t_2 \text{ begins} \\ 0 & \text{otherwise} \end{cases}$$

$$\forall t_1, t_2 \in T : S(t_1) \leq S(t_2) + (1 - \pi(t_1, t_2)) \cdot \lambda_L$$

$$\forall t_1, t_2 \in T : S(t_1) > S(t_2) - \pi(t_1, t_2) \cdot \lambda_L$$

$$\forall t \in T : \pi(t, t) = 1$$

Using $\pi(t_1, t_2)$, constraints can be enforced when any test starts its execution by way of the following. A new variable "$P(t_1, t_2)$" is introduced, which is equal to the power of the test t_2 when test t_1 starts its execution. If the tests t_1 and t_2 do not overlap, $P(t_1, t_2)$ must be zero. The same is true if t_2 starts its execution after t_1 starts. However, if both of these conditions are not

the case, then $P(t_1, t_2)$ is equal to the power of t_2. Below, the constant $P_{t,n}$ denotes the power of test t when it executes at DVFS index n. Also, λ_P is the highest possible power achievable in any test schedule (i.e., the power of every test executed simultaneously at the highest possible voltage/frequency).

$$\forall t_1 \in T : \sum_{\forall t_2 \in T} P(t_1, t_2) \leq P_{BOUND}$$

$$\forall t_1, t_2 \in T : P(t_1, t_2) \leq \Omega(t_1, t_2) \cdot \lambda_p$$

$$\forall t_1, t_2 \in T : P(t_1, t_2) \leq \pi(t_2, t_1) \cdot \lambda_p$$

$$\forall t_1, t_2 \in T : P(t_1, t_2) \geq \sum_{\forall n \in N_{t_2}} \sum_{\forall c \in C_{t_2}} \Psi(t_2, n, c) \cdot P_{t_2,n} - (1 - \Omega(t_1, t_2)) \cdot \lambda_p - (1 - \pi(t_2, t_1)) \cdot \lambda_p$$

A TAM pin constraint can be enforced using the same method which enforces the power constraint. Below, $C(t_1, t_2)$ is the number of pins used by test t_2 at the time when test t_1 begins, and the constant $C_{t,c}$ is the number of pins used by test t at TAM pin index c. Below, λ_c is the maximum possible number of pins that can be utilized in any test schedule (i.e., the sum of all tests using their maximum number of pins).

$$\forall t_1 \in T : \sum_{\forall t_2 \in T} C(t_1, t_2) \leq C_{BOUND}$$

$$\forall t_1, t_2 \in T : C(t_1, t_2) \leq \Omega(t_1, t_2) \cdot \lambda_c$$

$$\forall t_1, t_2 \in T : C(t_1, t_2) \leq \pi(t_2, t_1) \cdot \lambda_c$$

$$\forall t_1, t_2 \in T : C(t_1, t_2) \geq \sum_{\forall n \in N_{t_2}} \sum_{\forall c \in C_{t_2}} \Psi(t_2, n, c) \cdot C_{t_2,c} - (1 - \Omega(t_1, t_2)) \cdot \lambda_c - (1 - \pi(t_2, t_1)) \cdot \lambda_c$$

In order to test stacked dies in a 3D-IC, through-silicon vias (TSVs) are required to transport test data between dies. Testing a stacked die requires the availability of TAM pins on the lowest die as well as sufficiently many TSVs beneath lower dies to transport data upwards. Below, $V(d)$

denotes the number of TSVs (available for testing purposes) under each die $d \in D$. Since the first (bottom) die requires TSVs to access it, its value is pre-determined ($V(1) = 0$). Every die above the first (bottom) die has the requirement that enough TSVs must exist to test that die and every die above at any time in the scheduling process. For example, if there is a time in the schedule when 8 pins and 7 pins are required to test the second and third dies receptively, then the second die requires 15 TSVs beneath it and the third die requires 7 TSVs beneath it. Therefore, $V(d)$ in every die (except the bottom die) must have the property that at any given time it must be larger than the total number of TAM pins currently being utilized by all dies above it. Since the number of pins used by any given test can only increase when a test starts, $C(t_1, t_2)$ can be re-used to check the number of pins in use at any given time by checking it for every test t_1. Below, T_d denotes all tests in die d and dies above it.

$$\forall d \neq 1 \in D, t_1 \in T_d : V(d) \geq \sum_{\forall t_2 \in T_d} C(t_1, t_2)$$

$$\sum_{\forall d \in D} V(d) \leq TSV_{BOUND}$$

5.4 Experiments

The purpose of the first experiment is to evaluate the usefulness of the formulation when constraints are added or removed. Since there are no other scheduling formulations which incorporate all of the constraints incorporated in this formulation, a fair comparison of this formulation against any other is not possible. However, the effect on the final schedule result can be observed when constraints are removed and/or added, which will show the effectiveness of the scheduling method in selected environments. The first experiment will show the effect of the scheduling method when no power constraint is enforced, when no DVFS is allowed during test (i.e., only 1 V and its corresponding frequency is allowed during scheduling), and when tests are required to be in sessions. In the case when no power constraint is enforced, power violations in the schedule will also be recorded.

Table 5.1 Benchmark Information

Bench	ITC'02 Benchmarks	# Tests	Max Pow	P_{BOUND}
1	u226	9	0.78	1.02
2	g1023	14	3.26	11.40
3	f2126	4	4.26	3.84
4	q12710	4	34.50	41.86
5	a586710	7	3.90	11.30
6	u226, g1023	23	3.26	12.43
7	u226, f2126	13	4.26	4.87
8	u226, q12710	13	34.50	42.88
9	u226, a586710	16	3.90	12.32
10	g1023, q12710, u226	27	34.50	54.29
11	f2126, q12710, u226	17	34.50	46.73
12	q12710, a586710, u226	20	34.50	54.18
13	g1023, q12710, a586710	25	34.50	64.56
14	g1023, f2126, a586710	25	4.26	26.55
15	g1023, f2126, q12710	22	34.50	57.11
16	u226, q12710, a586710	20	34.50	54.18

The second set of experiments will observe the effect the TAM pin and TSV constraints have on the overall schedule quality. It is clear that allowing for more TAM pins and more TSVs during the scheduling process can lead to shorter test schedules, but given the high cost of TSVs and TAM pins during the design process, adding such hardware may not be worth a marginal increase in schedule quality. The second experiment will show specifically how allowing such hardware during the scheduling process will affect the test schedule quality.

The benchmarks used in this study are generated using the same methods described in Section 4.5.1, with some minor modifications. The first modification is a different set of ITC'02 benchmarks are used to create 3D-ICs. This is accomplished by "stacking" individual ITC'02 benchmarks on each other. Different ITC'02 benchmarks are also chosen to reduce runtime complexity. The second modification is more test information is added, namely different test TATs corresponding to different TAM pin availabilities. Details on the benchmarks are given in Table 5.1.

Table 5.2 Schedule Results

Bench	Proposed TAT (ms)	No DVFS TAT (ms)	No Pow TAT (ms)	Slack (W)
1	1365.462	1364.605	1365.462	-0.02159
2	77.516	78.285	75.573	-0.530
3	644.236	644.236	644.236	0.07999
4	4739.51	5576.448	3940.106	-31.184
5	158898.107	165546.541	158898.107	-3.584
6	1365.667	1364.703	1365.667	0.032
7	1794.498	1794.498	1794.498	0.12
8	5363.602	6405.573	4908.438	-9.2422
9	158898.107	171561.368	158898.107	-1.304
10	7204.977	7209.633	6536.602	-11.984
11	7204.977	7209.633	6536.602	-8.944
12	177631.934	177631.934	177631.934	-9.226
13	158898.107	158898.107	158898.107	2.968
14	158898.107	167038.047	158898.107	-5.264
15	4739.51	6121.938	4100.844	-16.060
16	158898.107	158898.107	158898.107	2.968

Other scheduling constraints are as follows. It is presumed that when scheduling tests under DVFS, voltages of 1, 0.9, 0.8 and 0.7 V are available to the scheduler for each core (i.e., $N_t = 4$ for every test t), and the standard testing frequency is 120 MHz. The frequencies corresponding to these voltages (required to find the constants $P_{t,n}$ and $L_{t,n,c}$) are found using the method given in [43], which is the highest possible operating frequency for a given voltage. To find the number of relevant pin choices available for individual tests and the associated TAT (at 1 V and the corresponding highest frequency) corresponding to these pin choices, the method from [27] is used. MILP formulations are solved used IBM/ILOG CPLEX.

5.5 Results

Table 5.2 gives the results of the first experiment described in Section 5.4. These results have a constraint of 5 data-carrying TAM pins per die and 5 data-carrying test TSVs per die (the number of pins and TSVs which control test execution are not incorporated as the number need not change depending on the schedule [49]). The table gives the TAT (in milliseconds) of each benchmark of

the proposed formulation, the proposed formulation without DVFS, and when no power constraint is enforced. The table also gives the power slack (i.e., the power bound minus the peak test power, in Watts) when no power bound is enforced, with a negative slack implying that the schedule violates the power bound.

Table 5.2 shows interesting trends with regards to the impact DVFS has on the schedule quality, and with regards to the effect a power bound has on the TAT. When comparing the schedule quality with and without allowing DVFS, it is clear that DVFS can have a benefit on TAT. Although DVFS can make individual TATs longer for each test (since 1 V allows for the fastest execution of any single test), it appears that DVFS creates enough opportunities for overlap under power constraints to reduce the TAT substantially. Of course, when the power bound is removed completely, the TAT is at its lowest. However, doing so appears to create an invalid schedule most of the time, even though a schedule with the same TAT can be achieved without violating the power bound, which implies enforcing a power bound will not always require increasing TAT. The normalized TAT results of Table 5.2 are also shown in Figure 5.2 to more clearly show the impact of the different scheduling methods, with violating no power bound schedules also being indicated.

Figure 5.3 gives the TAT result of 2 f2126 benchmarks stacked together (with combined power bound) as the number of data-carrying TAM pins and TSVs are increased. The figure gives the number of TAM pins allowed on the x-axis, while the number of TSVs allowed is given on separate plots. The figure shows that the decrease in TAT becomes smaller as more TAM pins are allowed for scheduling. This trend has been observed for single tests in [27], but to the author's knowledge this has never been observed for an entire SoC. However, the graph also shows an interesting relationship between the TAM pin bound and the TSV bound, most notably that they limit each other on the quality of the schedule. For instance, when only a single TSV is allowed for scheduling, allowing for more TAM pins does not improve the schedule quality. The converse is also true, since allowing extra TSVs gives no decrease in TAT when few TAM pins are available. One can conclude two things from this plot: first, although increasing the available TAM pins and TSVs can give a better schedule, the investment is worth much more for fewer pins and TSVs, and

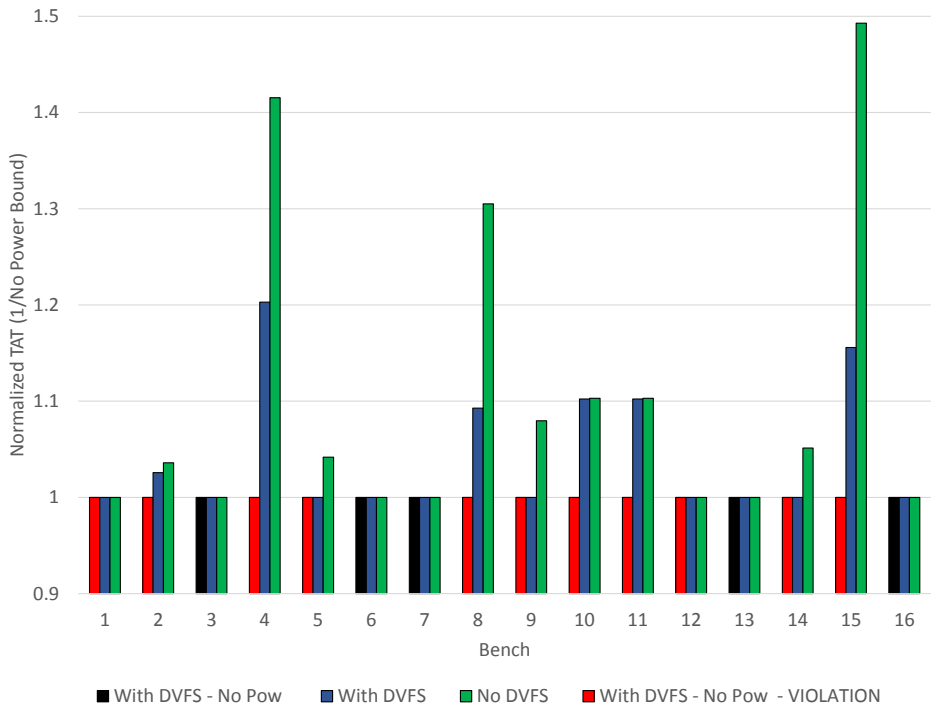

Figure 5.2 Normalized TAT results with invalid schedules marked.

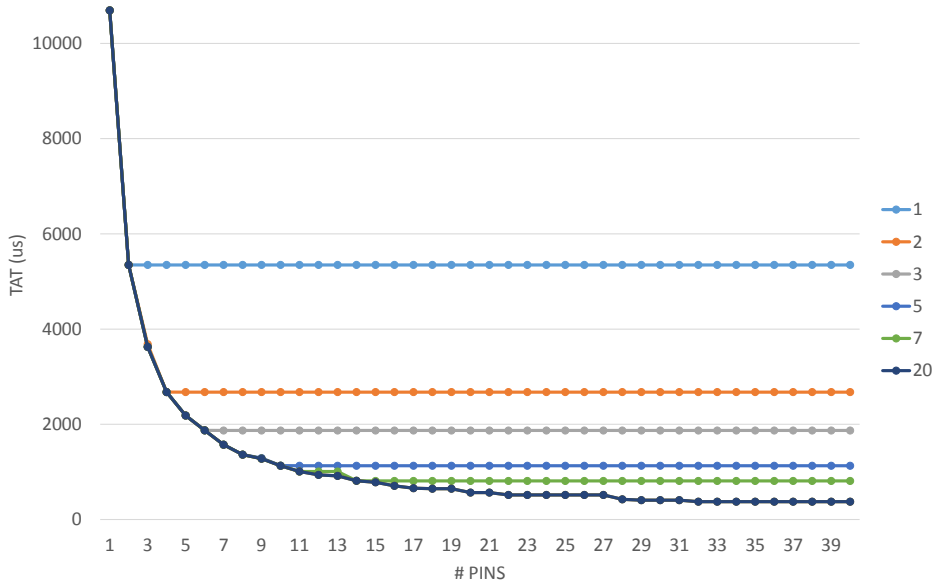

Figure 5.3 TAT result of two stacked f2126 benchmarks as more pins and TSVs are allowed for scheduling.

second, when scheduling tests for 3D-ICs, one must balance the available TAM pins and TSVs, as these hardware resources are costly and should not be allowed to go to waste.

5.6 Extensions for Other Constraints

Although the formulation presented here has given a method for scheduling under several different constraints, many other constraints may still be desired depending on the scheduling environment. Some scheduling environments can have different constraints due to the hardware available for test scheduling or due to the nature of the tests to be applied. For this reason, a number of extensions to the formulation are given below to cope with other testing environments.

It is not uncommon for pre-defined hardware constraints to be given to the test scheduler, which can come from either test-specific hardware (e.g., BIST) or non-test-specific hardware (e.g., memory). If such constraints are required to be enforced, Equation (4.11) can be enforced with the given hardware constraint constants.

Some schedules may require individual tests to be executed at one or more specific voltage/frequency values. The reason for this is that some faults are voltage/frequency dependent and can only be detected at such voltage/frequency values. Forcing the execution of a test t at a specific voltage/frequency value n can be achieved by setting $\sum_{\forall c \in C_t} \Psi(t, n, c) = 1$ for the required t and n. This modification may also require a change of the right-hand side of Equation (5.1) to reflect how many times a test must be executed.

Many designs have voltage/frequency hardware shared between tests, which implies that if two tests that share such hardware are executing at the same time they must also be executing at the same voltage/frequency. If this is the case for two tests t_1 and t_2, a new constraint given below can be added.

$$\exists t_1, t_2 \in T, \forall n \in N \sum_{\forall c \in C_t} \Psi(t_1, n, c) \geq \sum_{\forall c \in C_t} \Psi(t_2, n, c) - (1 - \Omega(t_1, t_2))$$

In some scheduling problems, the issue at hand is not necessarily to minimize the TAT of a device, but instead to reduce the overall cost of testing a device, since TAM pins and testing TSVs can be a large cost burden on a designer. Since the total TSVs ($Total_{TSV} = \sum_{\forall d \in D} V(d)$) and total

number of TAM pins ($\forall t_1 \in T : Total_{TAM} = \sum_{\forall t_2 \in T} C(t_1, t_2)$) are known, the cost to minimize can be easily formulated as follows, provided the cost of TAT (α), TAM pins (β), and testing TSVs (γ) is available to the scheduler.

$$cost = \alpha * t_{finish} + \beta * Total_{TAM} + \gamma * Total_{TSV} \tag{5.2}$$

5.7 Conclusion

This chapter has presented an SoC test scheduling formulation which incorporates several test scheduling constraints concurrently. By doing so, the formulation allows for the scheduling of tests under environments which previously required assumptions that lead to less than optimal test schedules. Incorporating several constraints at once will not only lead to smaller testing costs by producing shorter test schedules, but will also allow for more valid test schedules under modern design constraints.

Although the presented formulation is an advancement in its ability to incorporate modern testing constraints, future work can focus on further reducing the TAT of schedules by more accurately modeling testing constraints. For instance, allowing voltage and frequency to be scaled independently and continuously, as opposed to forcing voltage and frequency to be discrete values, has the opportunity to allow for more overlaps of tests under power constraints. Also, it is desirable to more accurately model the power of tests, as opposed to presuming the power of a test be constant throughout its execution. By removing these and other presumptions from the test scheduling process, the TAT of test schedules can be further reduced, but this is left for future work.

Chapter 6

Optimal Scheduling of Tests Under Temperature Constraints

6.1 Introduction

One new testing issue that has become important is the temperature during test. While older generations of integrated circuits did not have to cope with temperature issues, the power density of modern integrated circuits has risen to the point where the operating temperatures of devices have become a design issue which is often resolved through external cooling. The importance of this stems from the fact that if a device in question becomes too hot it may fail permanently. Testing devices in as little time as possible, in order to provide better test economy, can directly conflict with this goal. This is even more important if test compaction methods are used to reduce test data volume, since compacting individual tests can cause the power density of a device to increase to the point that may result in temperature causing permanent damage to the design under test (DUT). Therefore, it is often the goal to test a DUT not only as fast as possible, but also to stay within a given temperature bound [34]. Such constraints add to the complexity of an already NP-hard testing problem.

There have been several studies on scheduling tests with power constraints which use either generic heuristic-based algorithms or MILP-like formulations. However, using power alone to enforce a temperature constraint is inadequate. Although power and temperature are directly correlated, it is easy to show that a test profile and corresponding schedule, which has an adequate power constraint, will violate its temperature constraint. This is an important distinction since computation time for generation of accurate temperature profiles can be very large, which has resulted in more simplified temperature models being used during test generation and compaction. Bild et.

al. [46] presented a temperature-constrained MILP compaction formulation based on a steady-state temperature model which can be implemented from power traces. Rosinger et. al. [54] proposed a heuristic-based algorithm using a steady-state model, as well as a computation-intensive optimal algorithm.

This chapter addresses the problem of test scheduling for 3D-ICs while satisfying resource, power and thermal constraints. The contributions of this chapter include:

- Several MILP formulations for thermally-constrained test time reduction are presented, providing near optimal and pessimistic but practical solutions for 3D-ICs.
- Comparing our formulations and solutions against steady-state model based formulations extended to function for 3D-ICs to demonstrate the inherent inefficiencies of steady state models.
- Exploring the complexities of expanding test scheduling formulations in three dimensions.

The remainder of the chapter is organized as follows: Section 6.2 introduces several techniques for temperature modeling used in past studies and previous 3D-IC scheduling studies. Section 6.3 discusses the methodology used to generate the needed information for tests, as well as the use of the superposition principle. Section 6.4 presents a basic MILP formulation using the superposition principle and its reasoning, while Section 6.5 expands it to obtain a pessimistic and always valid solution. Section 6.6 discusses the results of using such methods against existing steady-state formulations, and the paper concludes with Section 6.7. The majority of the content of this chapter is a verbatim collection of previously published work from [55].

6.2 Past Work

The most basic technique to schedule tests is to rely on power-constrained test scheduling. The rational behind such a technique is that the temperature of a device is a direct result of the power dissipated by a device. Therefore, if the power dissipated during a test is kept below a given

power bound, then the temperature will stay below a proportional bound. There have been examples of generic power-constrained test scheduling such as in [29], and there have been studies in power-constrained test scheduling for the purposes of limiting device temperatures [56]. Although power and temperature are correlated, using power values alone is not adequate for determining temperature values. The actual temperature of a DUT is not just dependent on the present power being dissipated by the DUT, but also on the past power dissipation and the physical properties of the DUT. It is easy to show that a relatively low power value can raise the temperature of a well-insulated device to the point of failure. This problem is compounded by the non-linear relation between power and temperature. Although power and temperature values are related, they are not directly proportional because of the thermal capacitance of a device, which can lead to under and over-compaction by presuming that the temperature of a DUT is at all times proportional to the power value.

Another technique is to directly use power-generated temperature traces while scheduling. Such techniques use power traces to generate temperature traces using a simulator, most often based on the thermal RC model, for a given potential test schedule. Such a technique was proposed in [54], where an algorithm generates a potential test schedule, which is then simulated to check for temperature constraint violations. This technique has the benefit of being extremely accurate, and therefore can potentially provide an optimal schedule as suggested above. However, such temperature simulations have a drawback of having a very large runtime that cannot be ignored. Doing a single accurate temperature simulation is extremely computationally intensive, which in turn implies that simulating every possible test schedule is practically impossible, especially for DUT with numerous tests. This was also addressed in [54], which sacrificed the optimal algorithm for a heuristic-based algorithm for the sake of algorithm runtime.

An alternative to the two extremes described above has been to use steady-state values of temperatures of devices during scheduling. While some techniques attempt to model the actual cycle-accurate temperature of a test, steady-state models use the maximum possible temperature achievable by a test. Such a technique was effectively used in a formulation proposed in [46]. The advantage of steady-state formulations is the simplicity of the thermal RC model in which

thermal capacitances are deleted. However, removing the capacitance from the RC temperature model is also its downfall. If thermal capacitance is removed, the actual temperature of a test will always be lower then the presumed steady-state temperature unless the test runs for a long period of time, which can be a great hindrance when scheduling short, power-intensive tests. Another drawback of the steady-state model is tests which result in a very high steady-state temperature may be impossible to schedule. One last drawback of the steady-state model is that it is not cycle-accurate, which means individual tests cannot be partially overlapped if their combined steady-state temperatures violate the temperature bound.

An alternative to all these strategies and the one used in this study is the superposition principle proposed by Yao et al. [45]. The superposition principle states that the temperature offset of two or more tests running in parallel is the sum of temperature offsets generated by running those tests by themselves. Validity of this can be checked either by simulation or by mathematically manipulating the thermal RC model. The advantage of such a model is that it is cycle-accurate and its computational complexity is significantly less than other approaches with the same accuracy. The accuracy aspect is self-explanatory, since actual temperature values are being used instead of estimates, and the low computation-complexity is due to the summation of temperature offsets being a simple addition operation. Also, the superposition principle can be used in a MILP formulation since it is a linear formulation. What is different from previously discussed simulations is the temperature profile only needs to be simulated once per each test instead of once per proposed solution.

6.3 Methodology

Thermal simulation in this study is done through a modified version of the open-source thermal simulator, Hotspot [57]. Hotspot uses an RC thermal model to generate the temperature profile of a given IC with a given power trace. Modifications had to be done to Hotspot in order to simulate thermal through-silicon via (TTSV) behavior, which is not natively supported by Hotspot.

Power traces for a given test are generated using a pseudo-random Markov Chain technique [58]. Ideally, tests would be given in the form of inputs and outputs to actual ICs. However, due

to the proprietary nature of IC manufacturing, such information is unavailable. It is safe to assume that the maximum power of the IC is limited by some constant factor which is reflective of current technology scaling [59]. We also model the conditions where the power consumption of a test is known, such as high during scan-in and scan-out stages or low during idle stages. Although it may be sufficient to claim that the power dissipation is constant throughout a test for the purposes of scheduling, the Markov model provides a more accurate and realistic behavior of a test.

The modeling of test compatibility is a more complicated issue given the nature of 3D-ICs. Compatibility within dies is modeled based on the locality of the modules under test, with modules that are close to each other being more likely to be incompatible than if they were farther away from each other. This assumption is based on the observation that TAMs used between adjacent modules are more likely to be shared between each other, thus making them incompatible. Compatibility between dies is a more complicated issue that resolves itself to a simple solution for the sake of this study. In this study, dies are considered to be universally compatible. Although this is not a realistic assumption to make, it will not take away from the validity of this study since the purpose is to evaluate how different scheduling methods perform handling temperature conflicts and not hardware conflicts.

As with compatibility generation, the placement of TTSVs must be done somewhat randomly for lack of information. Although it is tempting to randomly place TTSVs across all dies, it is much more reasonable to follow some guidelines found in other research. For one, instead of placing several small TTSVs randomly, fewer larger TTSVs are used as suggested in [60]. It is also sensible not to "stagger" TTSVs placements between layers, instead having TTSVs run continuously throughout the stack as suggested in [61]. Although there have been studies into how TTSVs should be placed [61, 62] or how modules should be placed around TTSVs [63], for the purposes of testing it is reasonable to assume that the actual placement of TTSVs was left to the designer's discretion, whether they be good choices or not. From this assumption, in this study general TTSV locations are placed randomly, with TTSVs penetrating entirely though the stack with a randomly-generated clustering factor.

6.4 Optimistic MILP Formulation

6.4.1 Previously Used Constraints

Much like the formulations in Sections 4.4 and 5.3, the formulation presented here is a session-less formulation which allows tests to be arbitrarily scheduled in time. Therefore, many variables and constraints are borrowed from the previous chapters.

From Section 4.4, Equations (4.2), (4.3), (4.4), and (4.5) are reused to define the goal, start time, and finish times of tests. However, since DVFS and TAM pin constraints are not implemented in this study, $L(t)$ is no longer a variable, but instead is a constant.

Equations (4.7), (4.8), (4.9), (4.10), (4.12), and (4.13) are also borrowed to determine test overlap. Equation (4.11) is also borrowed to enforce non-test hardware constraints. Non-test hardware constraints will also become significant latter on when new "tests" are introduced, even if there are no hardware constraints.

6.4.2 Temperature Constraint

Because of the superposition principle, the enforcement of temperature constraint can be done in a manner similar to the power constraint in Sections 4.4 and 5.3. Unlike the formulation in [46] which computed the temperature in all cores in the formulation itself, this formulation will find the temperature by simply "adding" the temperature in individual cores. Thermal simulation is done for the power traces described in Section 6.3 to find the maximum temperature of every test in every core. By presuming that the temperature impact of a test is zero before a test starts and after a test finishes, the temperature of every test in every core is equal to the sum of the temperature impact of every overlapping test. In essence, the temperature constraint can be expressed very similarly to Equation (4.15), with the exception that DVFS has no impact, temperature is being summed (not power), and the constraint must be enforced in every core. The generated constraint is given below, where $T_M(t, c)$ is the maximum temperature offset from ambient temperature achieved by test t in the core c.

$$\forall t_1 \in T, \forall c \in C : \sum_{\forall t_2 \in T} \Omega(t_1, t_2) T_M(t_2, c) \leq T_{BOUND} - T_{amb}$$

An advantage to such a formulation is its simplicity. A steady-state counterpart such as proposed in [46] requires another binary variable for each pair of tests to represent power dissipation during test, as well as variables and constraints to represent the temperature during test. With this superposition-pasted formulation, such variables and constraints are not needed. It is true that a time-consuming temperature simulation is needed for every single test, but this simulation need only be performed once for each test before computing the formulation. Another advantage of this model over steady-state models is that it can cope with short tests and power-intensive tests that steady-state formulations can not. In a steady-state formulation equivalent to the one above, $T_M(t, c))$ would not be the maximum temperature achieved by the test, but the maximum possible temperature achieved given that the test length is relatively long. This means power-intensive tests that run for short periods of time can create a formulation which may indicate non-existence of a schedule even though one exists, since the steady-state temperature may be higher than the maximum temperature bound. Also, short tests who not do reach their steady-state temperature will still have their maximum temperature overestimated.

There are disadvantages of this superposition-based formulation that are shared with steady-state formulations: both formulations are pessimistic since they both over-estimate temperatures for the test, especially during the beginning of the test. This leads to under-exploitation of tests that can be partially overlapped without violating temperature constraints. This superposition-based optimistic formulation has another major shortcoming. It is a fact that the temperature change caused by a test when it finishes is not zero, but instead it takes time for the temperature offset to cool back to zero, an effect that is not modeled by this formulation. This can lead to over-compaction by scheduling too many "hot tests" back to back. This error is avoided by a steady-state formulation, since the maximum temperature that can be achieved during a test is the steady-state temperature, no mater what the temperature is at the start of the test. The given temperature bound may be exceeded if several power-intensive tests are scheduled consecutively in sequence rather then separated by power-conservative tests. In our approach after the schedule is found, it is verified for temperature violation by thermal simulation and the results in Section 6.6 show that this violation is rarely encountered.

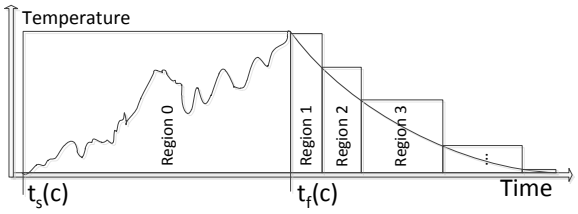

Figure 6.1 An example of partitioning a single test into regions.

6.5 Pessimistic MILP Formulation

The goal of this extended formulation is to account for the discrepancies of the previous optimistic formulation. As stated before, a flaw in the previous formulation is that it does not take into account temperature differentials caused after a test is completed. Although these "trailing temperatures" are not accounted for in a steady-state model either, the maximum temperature achievable is the steady-state temperature regardless of what tests happen immediately before it, but this is not the case with superposition-based formulations.

Before modeling these trailing temperatures, it is important to understand their nature. It is understandable that the temperature offsets of a test after the test is complete will fall back to zero (ambient temperature), but it is not instantaneous as it takes time to fall back to zero at an exponential rate. The challenge of modeling this decay is creating a formulation that models it efficiently, and with a reasonably high degree of accuracy. This exponential decay means that the trailing temperature offset will never be zero, but at the same time one can argue that in time it will be close enough to zero resulting in a negligibly small error. Our approach to modeling this effect efficiently is to assume the duration of the decay time to be significantly long but not unduly long. A very long duration will provide an accurate model but will take much longer to find a solution, whereas a very short duration will have the same disadvantage of optimistic formulation. In other words, the temperature offset of a test at the finish time of the last region should be virtually zero. Therefore as explained below we divide the temperature decay into regions and model each region with a constant temperature.

Although there are many ways to model these "trailing temperatures", it would be beneficial to model these trailing temperatures using already-existing variables and constraints. Because of the

nature of exponential decay, the time of the first region will be smaller than the second, which will be smaller than the third, and so on with each region having its own max temperature (see Figure 6.1). To model these regions in the previous formulations, they are modeled as tests themselves. Unlike actual tests which can occur at any time, these regions must explicitly occur after the test they represent. This conditions is embedded in the following:

$$\forall t \in T : t_{finish} \geq t_f(t, 0)$$

$$\forall t \in T, \forall r = 1 \ldots R : t_s(t, r) = t_f(t, r - 1)$$

$$\forall t \in T, \forall r = 0 \ldots R : t_{finish} = t_s(t, r) + L(t, r)$$

Here, R is the number of extra regions modeling the trailing region of each test. Essentially, each test expands into R more regions, with each region specifically starting when the previous region finishes, or in the case of the first region, when the original test finishes (here, region zero). Note that the overall finishing time only considers the actual test region (region zero), since the trailing regions are not actually part of the test.

The other constraint that is explicitly different for this formulation is compatibility. The compatibility for all trailing regions is not tied to compatibility of the original test or any other test. Instead, all trailing regions are explicitly compatible with all other regions (trailing and original) over every other test. This is important, for it allows trailing regions to overlap with any other region as long as this overlapping does not violate any temperature constraints. Hence:

$$\forall t_1, t_2 \in C, \forall r_1 = 0 \ldots R, \forall r_2 = 1 \ldots R : \Gamma(t_1, r_1, t_2, r_2) = 0$$

Since all other variables and constraints are expanded solely based on adding regions to each test, they will not be explicitly stated here.

The inherent advantage of such a formulation is its correctness over the more basic formulation and its compacting ability over steady-state formulations. The ability to compact more than a steady-state formulation comes from modeling maximum temperatures as actual maximum temperatures, not steady-state temperatures which can be higher than maximum temperatures, and

unlike the previously presented optimistic formulation, this formulation will not violate temperature constraints for trailing temperatures.

However, this extended formulation has a weakness in its trade-off between correctness and complexity. As more trailing regions are added, the more room there is for overlapping tests since temperatures will be modeled more accurately. This is true up to the point where the number of regions for each tests is equal to the number of time steps in the trailing region, at which point the optimal overlapping of trailing regions can be found. However, adding a single extra region is equivalent to doubling the number of tests to schedule, adding another is equivalent to tripling, and so forth, thus causing an exponential growth in complexity with the growth in the number of regions. On the other side of the spectrum, adding only a single new region will generate greatly pessimistic schedules, since it will be presumed that the temperature after testing will be the same as the temperature during testing for the length of the original test. Hence, it is in the scheduler's best interest to find a balance point between the two extremes.

If there is one extension to this formulation that would be most beneficial, it would be to split the original test into multiple regions as well. This would allow for overlaps that would not be available under steady-state or with the current pessimistic or optimistic superposition formulations.

6.6 Results

Like Sections 4.5 and 5.4, this study generates a set of original benchmarks using the ITC02 benchmarks. Also like in Section 5.4, the specific ITC02 benchmarks used are re-selected for the purpose of computation time. The makeup and information of the benchmarks used in this study are given in Table 6.1. The methodology used to generate specific power, temperature, and hardware compatibility for these benchmarks is described in Section 6.3.

Both the optimistic and pessimistic formulations are implemented using IBM's CPLEX. The steady-state formulation from [46] is implemented for comparison as well, which is extended to work for 3D-ICs. Each benchmark stack is evaluated using each formulation with its total runtime in seconds, total schedule time in milliseconds, and temperature slack in degrees Celsius with a

Table 6.1 Benchmarks Composing Different Stacks

Stack	ITC02 Benchmark	Cores	Longest Test	Max Temp
stack1	d281	8	9.63	93.42
stack2	f2126	4	4.16	69.91
stack3	d695	10	5.13	84.28
stack4	p22810	28	1.67	123.56
stack5	h953	8	9.49	82.10
stack6	p34392	19	8.99	126.26
stack7	g1023	14	8.74	95.61
stack8	d281, g1023	22	9.63	109.87
stack9	f2126, h953	12	9.47	70.19
stack10	d695, p34392	29	8.99	118.11
stack11	d281, f2126	12	9.63	105.31
stack12	h953, p22810	36	9.47	92.40
stack13	g1034, p34392	33	8.99	170.90
stack14	g1023, d281	22	9.63	105.96
stack15	d281, f2126, d695	22	9.63	135.19
stack16	h953, p34392, g1023	41	9.47	183.90

temperature bound of 145 ^{o}C. General results for each of these stacks are shown in Table 6.2, which includes the runtime in seconds and the temperature slack in degrees Celsius. The scheduled times of each stack for each formulation are shown in Figure 6.2.

A firsthand evaluation of the optimistic formulation shows that although it gives the best scheduled time in the smallest amount of computational time in all cases, it gives results that can violate the temperature constraint such as in stack4, which is marked an invalid schedule (IS) in the table as it has a slack of -4.56. The pessimistic formulation, on the other hand, shows promising results especially when compared to the steady-state formulation of [46]. Despite the pessimistic nature of the formulation, it achieves better compaction results in less time. However, in fairness this time does not account for the time of temperature simulation, which in this study can take between thirty minutes to three hours if each temperature trace for every test is generated in parallel. However, this is a constant overhead time for each scheduling problem which gives a large improvement in the final result. It is also worthwhile to note that although the optimistic formulation can produce invalid schedules, in most cases it gives better results then the more pessimistic formulation in

Table 6.2 Benchmark Results Under Different Formulations

	Formulation from [46]		Optimistic Formulation		Pessimistic Formulation	
Stack	Runtime	Slack	Runtime	Slack	Runtime	Slack
stack1	0.81	26.16	0.76	10.26	0.56	10.26
stack2	0.05	50.87	0.01	49.51	0.07	49.51
stack3	6.02	38.21	0.43	32.86	0.66	34.26
stack4	X	X	12.29	**IS**	631.12	4.51
stack5	0.42	31.35	0.04	31.35	0.17	31.35
stack6	1046	20.26	1.34	13.37	683.55	20.26
stack7	10.79	23.68	0.2	19.26	9.44	19.26
stack8	612.02	15.84	0.62	9.62	205.55	15.84
stack9	6.39	49.68	0.06	49.68	0.54	49.68
stack10	902.52	1.12	2.17	0.53	600.21	1.12
stack11	3.28	14.68	0.11	11.22	1.19	14.68
stack12	X	X	7.24	25.54	600.27	31.26
stack13	X	X	X	X	X	X
stack14	X	X	1.99	16.65	124.45	16.65
stack15	1282.46	4.54	1.45	3.84	160	4.02
stack16	X	X	X	X	X	X

IS indicates an invalid solution due to negative slack.

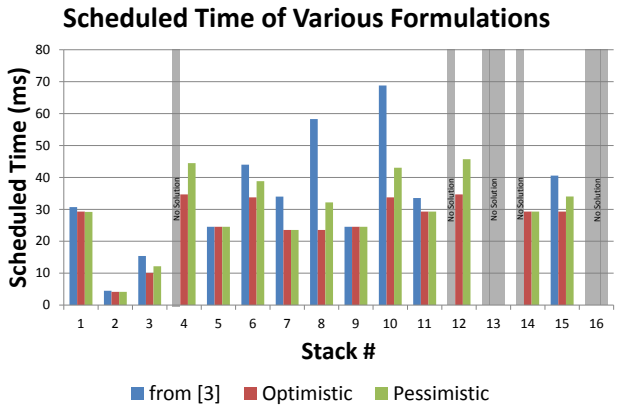

Figure 6.2 Scheduled times of various formulations.

shorter computation time. It is possible to still use this formulation with success given that the result is properly checked.

It must also be pointed out that the steady-state formulation used in [46] failed to generate a schedule in many cases. As stated in earlier sections, steady-state formulation cannot schedule any test that has a steady-state temperature higher than the given temperature bound. Since stacking dies on each other greatly increases the thermal resistance to ambient of many modules, it is not uncommon to see the steady-state temperatures rise greatly, even with high TTSV coverage. This leads to a high rate of failure for steady-state formulations, especially with power-intensive tests, no matter how long the tests are.

A final point to make is that two benchmark stacks (stack13 and stack16) failed to achieve a schedule under any formulation. These stacks are included to demonstrate that some tests, even under realistic conditions, may be impossible to schedule under a given temperature constraint. This situation is especially true for 3D-ICs. Although outside the scope of this study, the only way to resolve this issue is to change the original test itself. Though the scheduler may not be able to change the individual vectors of a test, the scheduler may have the ability to divide a large test into smaller tests. This can allow any hot test to effectively have its maximum temperature reduced to the maximum temperature of the smaller tests. However, this approach will not be effective using a steady-state model, since the power values of these tests will remain the same and therefore have the same steady-state temperatures as before. Given how stacking dies drastically increases thermal resistance, this approach of dividing large tests into smaller ones appears inevitable and remains to be a problem for our future research.

6.7 Conclusions

MILP formulations based on the superposition principle have been shown to provide superior schedules compared to previous formulations, as well as having faster computation times. This decrease in overall test time comes from the ability to use actual temperature traces instead of approximations like steady-state models, and the faster computation times comes form the linear nature of the superposition principle.

However, it was discovered in this chapter that in some cases any formulation discussed in the paper will fail, especially with 3D-ICs. This fact, as pointed out in previous sections, is due to

the condition that single test's temperature profile has a maximum temperature that will violate the given constraint by itself, making it impossible to schedule that test without partitioning it into several smaller tests. This observation is even more important because it implies that the steady-state model will not be able to address this issue, since dividing a test into smaller tests does not decrease steady-state temperatures. This problem becomes magnified with 3D-ICs, since when individual dies are stacked on top of each other the thermal characteristics become more difficult to deal with. When dies are stacked, the power during test must be reduced from the single-die case. Since it is often undesirable to change the tests (it may not be possible provide the desired fault coverage), dynamically partitioning the test during scheduling is a more practical solution. Although outside of the scope of this study, this subject is primed for future work.

Chapter 7

Temperature-Aware Test Partitioning

7.1 Introduction

So far in this study, several different methods of reducing TAT have been introduced. These methods have ranged from generating compact tests, using test-specific hardware, developing new tests models, to scheduling tests for a given test model. All of these methods have built on the complexity of previous methods, with each new study introducing a more complex testing environment. With the ever-increasing complexity of IC test, simpler methods must be utilized to reduce the complexity of test while not sacrificing TAT.

One possible remedy for reducing the TAT without unduly increasing the complexity of test is test partitioning. The partitioning of tests is the practice of dividing a given test into smaller "subtests" (which can often be done since in most cases test vectors can be applied in any order) in order to reduce TAT [44]. Partitioning to reduce TAT makes use of overlaps that can be created by diving large tests into smaller ones which may otherwise not exist. Another benefit of partitioning, relative to other test compaction methods, is that no extra hardware is needed to implement it and it requires no or minimal changes in the design flow since partitioning is done on already-existing tests. The effect partitioning has on reducing TAT was observed early in hardware-constrained scheduling [44]. It has also been observed in power and temperature-constrained scheduling [64, 30]. Partitioning methods investigated in the past have either been simplistic (equal sized-partitions, partition only when required by constraints) or ad hoc. However, to the best of our knowledge the effectiveness of different partitioning methods or what makes a good partition has never be explored.

The contributions of this chapter are as follows:

- To show the significant impact temperature has on 3D-ICs as opposed to non-stacked dies.
- To show that partitioning not only has a significant impact on the quality of schedules but in some cases it is required to obtain temperature constraint schedules.
- To develop a partitioning scheme and to show that it has a significant impact on the quality of SoC test schedules compared to ad hoc partitioning methods.

The rest of this chapter is organized as follows: Section 7.2 gives a brief history of test partitioning for SoC test scheduling. Section 7.3 provides a partitioning method with the goal of developing superior test schedules compared to other partitioning methods. Section 7.4 gives a design of an experiment to evaluate the effect stacking of dies has on temperature-constrained test scheduling, the effect partitioning has on schedule quality, and the effectiveness of the proposed partitioning method. Section 7.5 gives the results of the experiment and a discussion of the results, and the paper concludes with Section 7.6. The majority of the content of this chapter is a verbatim collection of previously published work from [65].

7.2 Test Partitioning

The concept of test partitioning is based on the presumption that an SoC test consists of several independent input vectors that can be applied in any order. Tests for SoCs often consist of a series of input vectors and states, with the output to these vectors and states determining if the DUT is faulty. If the state of the DUT can be manually set (e.g., using scan chains), the the order in which tests are applied and outputs observed is irrelevant. Therefore, re-arranging and grouping a "test" consisting of several vectors into several "sub-tests" can be done without changing which defects will be detected.

Partitioning tests to reduce TAT has been studied under hardware and other constraints, but all previous partitioning methods have been very basic (trivial) and the effectiveness of a partitioning method has never been evaluated compared to other methods. In previous studies, partitioning has

Figure 7.1 By partitioning, greater overlap between tests t_1 and t_2 can be achieved.

been clearly observed to reduce TAT. However, in all previous studies the method of partitioning was either simplistic or ad hoc.

7.3 Proposed Partitioning Method

7.3.1 Partition Quality

Before any partitioning method can be proposed, it must first be determined what qualities define a good as opposed to a bad partition.

Since the goal of test scheduling can be stated as creating as much parallelism in applying tests as possible so that TAT can be minimized, the objective of partitioning should be to allow for new parallelism amongst tests. The worst possible test schedule executes only one test at a time, while the best possible test schedule executes every test at $t = 0$. However, the latter often cannot be achieved due to various constraints. Therefore, the goal of partitioning should be to allow for constraints between two tests to be relaxed. In the case of hardware constraints, this cannot be done between two tests (although it is possible with 3 or more [44]) since new partitions will have the same hardware compatibility as their parent tests. It can be said that hardware constraints are "constant" throughout test, but this is not the case with temperature since temperature fluctuates during test. If parts of one test may overlap with parts of another provided those parts occurred in a certain order, than partitioning can give that order to the scheduler (see Figure 7.1 for an example).

For temperature-constrained test scheduling, two tests cannot be run in parallel if their combined temperature impact is greater than the given temperature bound, therefore the goal of partitioning should be to "lower" the temperature of a test to allow it to run in parallel with many more other tests. In reality, partitioning will not lower the temperature impact of the original test since

partitioning will not change the power of the test (with the exception of partitioning overhead). Clearly, each partition of a given test can have no more temperature impact than the original test if the impact of the overhead of partitioning is ignored. Of course, if partitions of a given test are scheduled one after another, the temperature impact of a test will not change at all. However, if the partitions of a tests are instead viewed as new tests whose combined run time are equal to the original test, these new tests may have lower temperatures than the original test and therefore can be scheduled in parallel with other tests that the original test could not be scheduled in parallel with.

Because the peak temperature of a test is what determines the overlapping potential of two tests, another important objective of a temperature-constrained partitioning method should be to partition tests such that their temperature is constant throughout the test. The example in Figure 7.1 showed the peak temperatures of test t_1 not allowing for any overlap with t_2 with the exception of insignificant proportions of the start and finish of the test. It can be said that if a peak temperature value occurs in a test, then another test cannot overlap "beyond" when that value occurs if the combined temperature at this point violates a given bound. At the same time, if the temperature of a test is constant, then it can be assumed that any partitions of that test will have the same temperature as the original test (this presumption will be addressed later). This implies if two tests with constant temperature are incompatible, then no partitioning can make them compatible. Therefore, it should be the goal of a partitioner is to divide tests in such a way that makes the temperature of partitions as constant as possible, since partitioning any more will give no better result.

Now that it is known that the temperature of test partitions must be as constant as possible, the goal is to create such partitions in a practical manner, which would normally be difficult due to the simulation-intensive nature of temperature profiles. As suggested in Section 7.2, generating a temperature profile from a given power profile requires a time-intensive thermal simulation. Since partitioning a test creates a new power profile for each partition, simulating the temperature of all relevant partitions whenever a partition is made is impractical.

However, the observation that the temperature of a partition will never be greater than the original test can allow for a reasonable estimation of partition temperatures based on the temperature of the original test. Partitions of a test share the same power profile of the original test (plus scan overhead) and are executed in the same environment as the original test. Therefore, the temperature profile of a partition is guaranteed to be equal to or less than that of the original test. This is because temperature is a function of current and past power dissipation, and partitioning can only remove past power values. Additionally, the longer a partition is, the more likely the temperature profile of the partition will match the original test. This is because in the longer partition, past power values will more closely resemble that of the original test.

Based on the observations that the temperature of a partition can be reasonably estimated from the original test and the temperature of each portion should be reasonably constant, a partitioning method is now proposed to achieve these goals.

7.3.2 Partitioning Method

The goal of the proposed partitioning method is to split each test into high and low-temperature partitions based on the temperature of the original test, thereby making the temperatures of each partition constantly high or constantly low. By doing this, high-temperature regions of tests that are normally incompatible with every other test due to temperature constraints will be isolated, while low-temperature regions are (temperature) compatible with every other test. This separation will allow for the maximum overlap without violating temperature constraints.

The first step of the partitioning method is to choose a temperature that defines the difference between high and low temperature. This temperature is referred to as the "partitioning temperature". Based on the assertion made in Section 7.3.1, the temperature of these partitions will be lower than the original test or will be a close match to the original test if the partitions are long enough. Note that if every test is a low temperature test, than no test will be partitioned. This is also not undesirable, since temperature will not play an important role in the test and therefore will not have a significant role in scheduling. When a partitioning temperature is chosen, each test

Figure 7.2 By partitioning before T_{PART} is met, the temperature of all partitions is reduced.

is partitioned when the temperature of the test crosses the specified temperature as long as other conditions are met.

The first condition to partitioning is that any partition generated is not exceptionally small (in this study, less than $30\mu s$ long), the reason for this being that small partitions will always have low temperature, even if the temperature of the original test is high. Also, if partitioning overhead is a factor, small partitions will have relatively large overhead of partitioning which defeats the purpose of partitioning. In the case of temperature rising above the partitioning temperature and then falling below less than $30\mu s$ later, a single partition will be made thereby eliminating this "temperature peak".

The second condition is that tests are partitioned not at the point where the temperature passes the partitioning temperature, but instead shortly before that point (in this study, $30\mu s$). As stated before, the temperature of a partition is dependent on the past values as well as the current value. If partitioning a test creates two partitions, "partition one" and "partition two", then this will decrease the maximum temperature of partition one while keeping the maximum temperature of partition two the same as long as the length of partition two is not exceptionally small. This is because "past values" are removed from partition one and added to partition two. An example of this is illustrated in Figure 7.2.

7.4 Experiment

To judge the effectiveness of the proposed partitioning method in 3D-ICs, it must be compared against other partitioning methods. Although partitioning has been done in other studies, other studies used simple methods such as partitioning arbitrarily, partitioning for equal-sized partitions,

or partitioning only when absolutely required (running of a test by itself violates the given temperature bound). This study implements both of these methods for comparison, as well as scheduling without partitioning.

To fairly compare the proposed partitioning method and partitioning with an equal number of equal-sized partitions, the number of partitions created with equal-sized partitions is set equal to the number of partitions created by the proposed method. This is done because the number of partitions can effect the quality of the schedule, and it would be an unfair comparison if one partitioning method creates more partitions than the other.

To observe the effect the number of partitions has on the effectiveness of the the different partitioning methods, partitioning for each benchmark is done twice: once with a fewer number of partitions, and once with a greater number of partitions. As stated earlier, when temperature-based partitioning is done, the number of partitions created for each test is recorded and equal-sized partitioning is done with the number of partitions specified. In this study, it is done twice by partitioning with partitioning temperature equal to 40° C (fewer partitions) and 30° C (larger number of partitions). These two temperatures were chosen for T_{PART} since as T_{PART} increases, the number of partitions created decreases. This is illustrated in Figure 7.3, which shows the number of partitions created in the selected benchmarks described later.

Much like in Sections 4.5.1, 5.4, and 6.6, the study generates benchmarks from the ITC02 benchmarks. Also like in previous studies, specific benchmarks were chosen for the sake of run-time constraints. Specific details of the benchmarks used are given in Table 7.1. Also, the specific benchmark files are available through [43].

The MILP-based formulation proposed in Chapter 6 is used to schedule tests. This scheduler was chosen because of its deterministic nature (the same result will always be achieved) and because of its performance. The scheduler is an optimistic scheduler, meaning it may produce a schedule that may violate temperature constraints. Because of this, schedules are re-simulated after they are produced to check validity. T_{BOUND} for scheduling is set to 55° C, which is intentionally set below the temperature bound for some benchmarks. This is done to show the requirement of partitioning for scheduling tests for 3D-ICs. The CPU computation time is not measured in the

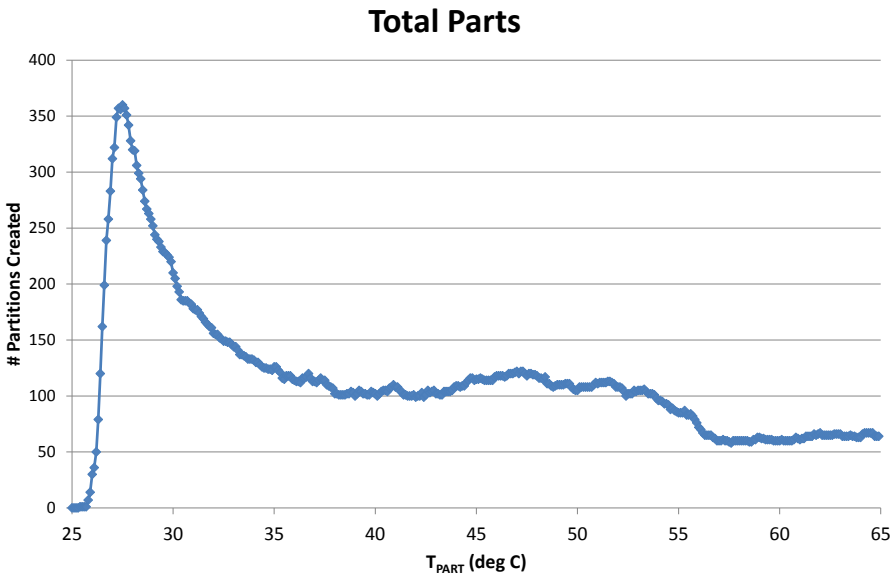

Figure 7.3 Effect of T_{PART} on the number of partitions.

experiment since the scheduling time is the same across all partitioning methods and the partitioning time is trivial for every proposed partitioning method (except minimal partitioning at T_{BOUND} only, which requires successive temperature simulations).

7.5 Results

Table 7.2 gives the results of partitioning with partitioning temperature equal to 40° C. The table gives the schedule length, i.e. TAT (in ms), provided by the scheduler when no partitioning is done, when minimal partitioning (partition only when T_{BOUND} is violated) is done, when equal partitioning is done, and when temperature-based partitioning is done.

The results from Table 7.2 show several trends that show the effect stacking dies has on temperature-constrained test scheduling, and also show the proposed partitioning method performing well in conditions where temperature has a significant impact on test scheduling. For every benchmark, the result for the proposed partitioning method is always better than or equal to without partitioning. This is predictable, since the effect partitioning has on possible test schedules has been explored. Also, some benchmarks fail to find a solution without partitioning. This is

Table 7.1 Benchmark Details

Bench #	3D-IC Bench	# Dies	# Tests	Highest Temperature
1	1-1-LLL	1	9	38.07
2	1-6-LML	1	4	39.25
3	1-7-LHL	1	4	48.66
4	1-9-MML	1	19	42.09
5	2-4-LML	2	12	35.21
6	2-6-MMM	2	23	42.30
7	2-8-MHM	2	35	66.76
8	2-9-HHM	2	63	62.04
9	2-10-MHH	2	38	68.98
10	3-2-MMM	3	33	46.39
11	3-3-LML	3	22	48.47
12	3-4-HMM	3	79	44.14
13	3-5-MHH	3	54	82.80

Table 7.2 Schedule results (TAT) for T_{PART} = 40° C in ms

Bench #	No Parts	Min. Part.	Equal Part.	Temp. Part.
1	257.4	257.4	257.4	257.4
2	28.36	28.36	28.36	28.36
3	311.72	311.72	311.72	311.72
4	37.24	37.24	37.24	37.24
5	16.84	16.84	16.84	16.84
6	42.35	42.35	31.50	31.52
7	X	500.54	358.68	358.68
8	X	335.56	288.4	269.98
9	X	945.02	757.04	757.04
10	19.65	19.65	16.88	17.04
11	226.23	226.23	150.96	138.03
12	40.66	40.66	30.26	30.24
13	X	900.56	771.70	823.42

because these benchmarks have one or more tests which violates the temperature constraint. This is more common as the number of dies in the 3D-IC increases, which is due to the heat-insulating nature of 3D-ICs. For benchmarks which are not 3D-ICs but instead conventional one-die ICs, all benchmarks have no difference between the partitioning methods. This is because the temperature impact of the tests are minimal. When the number of dies increases, the difference between equal and temperature-based partitioning increases, which is again due to the effect stacking dies has on

Table 7.3 Schedule results (TAT) for T_{PART} = 30° C in ms

Bench #	No Parts	Min. Part.	Equal Part.	Temp. Part.
1	257.4	257.4	257.4	257.4
2	28.36	28.36	28.36	28.36
3	311.72	311.72	311.72	311.72
4	37.24	37.24	37.24	37.24
5	16.84	16.84	16.84	16.84
6	42.35	42.35	35.84	35.84
7	X	500.54	416.60	373.58
8	X	335.56	289.62	274.68
9	X	945.02	759.12	709.98
10	19.65	19.65	17.32	17.32
11	226.23	226.23	192.46	190.72
12	40.66	40.66	32.38	32.26
13	X	900.56	771.70	792.96

temperature. Although temperature-based partitioning does not always yield the best result, it is more likely to give a better result, which is shown in Figure 7.4.

Table 7.3 gives the results when the partitioning temperature is set to 30° C, which shows the effect increasing the number of partitions has on the effectiveness of partitioning. The results of this table show the same general trend as shown in Table 7.2: the partitioning gives better schedules than not partitioning, and temperature-based partitioning is more likely to give superior schedules than equal partitioning, especially when temperature plays a significant role.

Figure 7.4 gives the normalized TAT of all partitioning methods on relevant benchmarks, which in turn gives insight into the effect increasing the number of partitions has on schedule quality. The normalized TAT of the partitioning methods clearly show that more partitions generally lead to a better schedule. This is so because more partitions create more opportunity for test overlap. However, this is not always the case, like with benchmark numbers 4 and 13, since the quality of a partition is dependent on more than just the number of partitions created. Figure 7.4 also shows that the majority of the time temperature-based partitioning performs better than equal partitioning. Again, this is not always the case, such as in benchmark number 13.

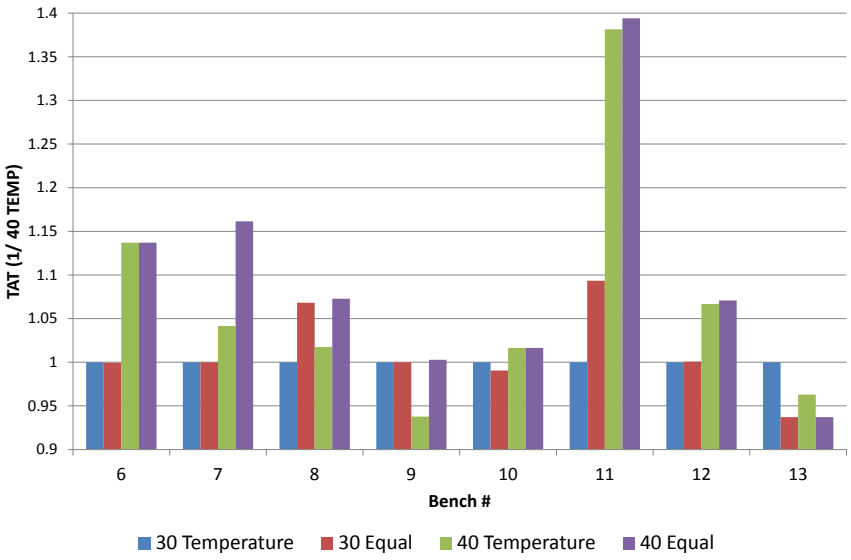

Figure 7.4 Normalized schedule results of different partitioning methods.

7.6 Conclusion

The results of test scheduling under temperature constraints show the great effect stacking dies has on the difficulty of scheduling under temperature constraints. The thermal insulating nature of 3D-ICs has a significant impact during test, for the overheating of dies creates yield loss and increases manufacturing costs. Special attention and techniques need to be applied to temperature constraints while testing 3D-ICs, and the partitioning of tests shows a substantial improvement in schedule quality.

The results have shown both the effect partitioning has on test schedule quality and the effectiveness of partitioning with temperature in mind. Although partitioning has always been known to crate higher quality test schedules, specific methods of partitioning were never evaluated. This study has provided such a partitioning method and evaluated it against other methods, and has shown the effect a specific partitioning method can have on the quality of a schedule, especially in environments where temperature has a significant impact such as 3D-ICs.

Chapter 8

Summary

This chapter summarizes the contributions of this dissertation. This chapter will also give possible future directions that can be taken to further improve the quality of TAT for SoCs.

8.1 Conclusions

With the ever-decreasing feature sizes of silicon-based ICs, it is becoming a continuing challenge to keep IC manufacturing costs low. The manufacturing of silicon-based ICs is a far-from-perfect process, and therefore faulty devices must be prevented form being released to the consumer. To prevent the release of faulty devices, test plays a critical role in detected faulty/defective ICs to prevent the faulty devices from being released. However, as more transistors are packed onto a single die, the testing of all individual ICs is becoming a more difficult task. Not only is the number of tests required is increasing, but also accessing individual transistors is becoming more difficult. These issues are problematic, as an increase in test time directly corresponds to an increase in manufacturing cost.

New design techniques, like SoCs, have not only assisted in faster development of new ICs, but have also made the task of manufacturing such devices more cost-effective. Although SoCs were introduced as a consequence of smaller design footprints and the desire to pack entire systems onto a single die, they have also shown themselves as an efficient design technique for more cost-effective IC development. The SoC design model is especially useful in incorporating 3rd party designs for even faster development. Thankfully for the tester, the SoC design model has also

made the task of applying tests to an IC simpler, so much so that using the SoC model exclusively for applying tests has shown promise in significantly reducing TAT.

As new design technologies are introduced, such technology must be utilized to their fullest extent during test to keep manufacturing costs down, especially as new design constraints are being introduced. With decreasing IC feature sizes, new design issues are being introduced which before were a design afterthought. One such issue, power density, has become such an issue that power during test (and normal operation) must be limited in order to prevent device damage or temporary device failure. For non-test purposes, DVFS was introduced for the sake of reducing device power consumption when full device performance is not required, and has been shown to be very effective. However, for testing purposes, such technology has not been fully utilized. This dissertation has shown new methods of utilizing DVFS during test to achieve higher-quality test schedules, especially under various hardware constraints.

As IC feature sizes continue to scale downwards and new design technologies are introduced, new testing techniques must be utilized to keep manufacturing costs low. Just as power during test (and normal operation) was once a design afterthought, so was temperature. However, the temperature density of modern ICs has risen to the point where it can no longer be ignored, so much so that ignoring temperature can damage a device. Temperature during test is especially problematic, since the time-dependent nature of temperature is more difficult to model than power. Also, temperature during test is higher than during normal operation due to the unique physical environment of test. This dissertation has introduced a method of scheduling temperature-constrained test for SoCs while accurately modeling temperature, and has also given methods for achieving shorter TAT without increasing scheduling complexity through test partitioning.

8.2 Future Directions

Although this dissertation has greatly contributed to the reduction of TAT under various constraints and conditions, a further reduction in TAT is always desirable. As new IC technologies are introduced, the SoC TAT reduction problem will become ever more complex, which will possibly leave current methods incapable of reducing TAT. Also, new IC technologies may be introduced

in the near future (e.g., MuGFET), new testing strategies will need to be developed to test such devices while still keeping TAT low. This section introduces some possible future directions that can be taken to continue the reduction of TAT for SoCs

8.2.1 Heuristic Benefit Analysis

All the SoC test scheduling methods presented in this dissertation have been of the form of optimal scheduling formulations, which may be further improved (in terms of compute time to obtain the solution) by using heuristics. Although optimal formulations have the benefit of providing the best possible result given a series of constraints, heuristics have still been implemented many times for SoC test scheduling [45, 46]. The first possible benefit of using heuristics over optimal formulations is the computation time of heuristics is often much less than that of optimal formulations. This can be especially beneficial for large problem instances (e.g., hundreds of cores), in which case optimal formulations may not find a any result in a practical amount of time. Also, heuristics can sacrifice optimality in exchange for allowing for "loosened" constraints, which in turn can provide reduced TAT. For instance, formulations in this dissertation have presumed power and temperature to be constant during test, but if they can change during test than new overlaps of tests can be found. Although a heuristic is not guaranteed to find a better TAT than an optimal formulation, the comparison of the two is a worthwhile effort.

8.2.2 DVFS Benefit Analysis

Although it has been shown that using DVFS for SoC test scheduling will definitely have a positive impact on TAT, the significance of its impact is still in question. Results from Sections 5.5 and 6.6 clearly show that using DVFS for power-constrained test scheduling will never yield an inferior test schedule, but instead will often give a far superior test schedule. However, a superior test schedule will not always be obtained. There are many benchmarks seen in Sections 5.5 and 6.6 which yielded the same results both with and without DVFS being utilized. Also, there were many instances in which the decrease in TAT was marginal.

Although any decrease in TAT is welcome, such a decrease may not be desirable if the cost to obtain it (through more complex hardware) is greater than the benefit. It is is true that decreasing TAT will decrease the ATE resources required, and therefore reduce testing costs. However, it is also true that implementing DVFS hardware can be costly, and it can also be costly to use existing DVFS hardware for testing purposes. For instance, the cost to implement DVFS hardware can be expressed as a constant in a manner similar to Equation 5.2. Therefore, if the decrease in TAT is marginal (if there is any decrease at all), implementing the cost of test may actually increase by utilizing DVFS hardware. Therefore, work should be done into the benefit of DVFS for SoC test scheduling. More specifically, the environments in which DVFS can be beneficial to SoC test scheduling should be found so that such environments can be utilized in the future.

Initial work was started on this subject with the goal being to determine the usefulness of specific design types, but results thus far have been inconclusive. Correlating results in Chapters 4 and 5 to design layouts have shown improvements in TAT to be more common in designs that can be characterized as "homogeneous", i.e., designs in which no single core dominates the others. On the other hand, an improvement in TAT was seen less often in "heterogeneous" designs, i.e., designs in which one core composes most of the design. Examples of such designs are illustrated in Figure 8.1.

Work was started on an attempt to compare to the two design types with the goal being to conclusively show homogeneous designs are superior (i.e., less TAT is achieved under a power constraint). Results have shown that as a design is "divided" into smaller parts, smaller TAT can be achieved, but such a result is predictable since dividing a design can only increase the scheduling possibilities. Also, when comparing two different design types, the comparison of results was found to be inadequate in determining the superiority of one design over the other, since although there would be a clear "superior" design (shorter TAT under given constraints), never could it be said that the two designs being compared were "equal" in any significant regard. To fairly compare the two design types, a theory must be formulated as to when the two separate design types can be fairly compared, and such a theory is left for future work.

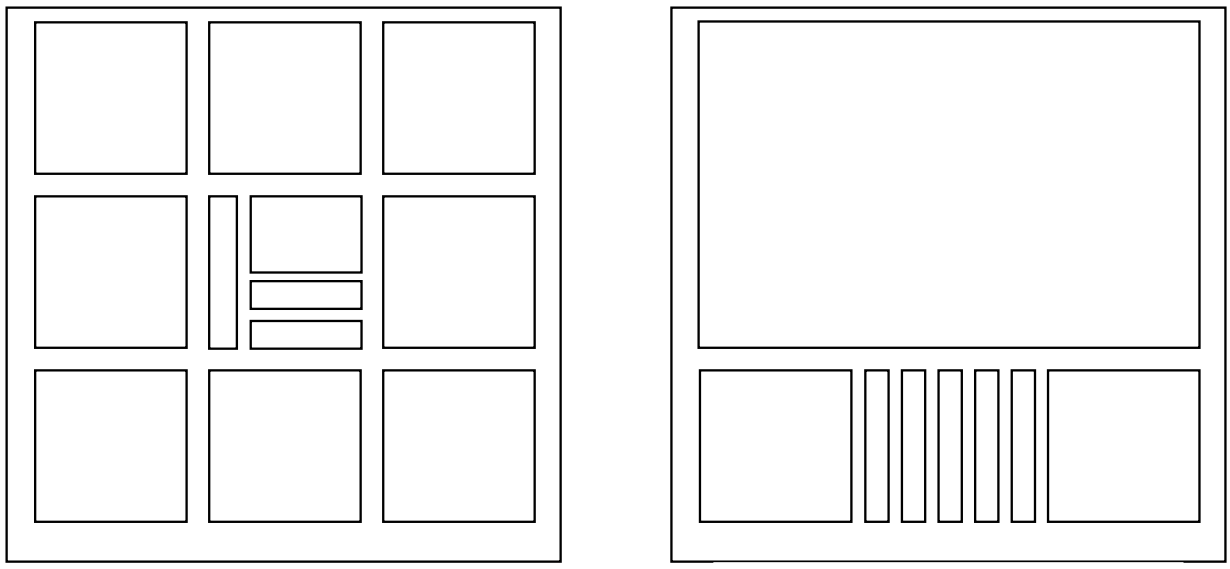

Figure 8.1 Examples of homogeneous and heterogeneous designs.

8.2.3 Power Reduction Hardware

Although this study focused on SoC test scheduling in order to reduce TAT under power and temperature constraints, many other techniques exist, which when combined with SoC test scheduling may reduce TAT even further. This dissertation has made references to scan chains on multiple occasions, as scan chains have become an important method of applying tests. This dissertation has also made reference to test-specific hardware (like scan chains) consuming extra power during test, which in turn makes meeting power limits more difficult. Attempts have been made in the past to reduce the power consumption of test-specific hardware [66, 67, 68], however, few have effectively utilized DVFS. By utilizing DVFS with test-specific hardware, power reduction can be further optimized so as to reduce power by significant amounts, thereby allowing for faster testing under power constraints.

LIST OF REFERENCES

[1] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, 1965.

[2] R. Camposano, D. Gope, S. Grivet-Talocia, and V. Jandhyala, "Moore meets maxwell," in *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1275–1276, IEEE, Mar. 2012.

[3] W. Moore, "A review of fault-tolerant techniques for the enhancement of integrated circuit yield," *Proceedings of the IEEE*, vol. 74, no. 5, pp. 684–698, 1986.

[4] Y. Lin, W. J. Wu, and J. C. Lin, "Controlling defects in thin wafer handling," May 2014.

[5] R. Karri and F. Koushanfar, "Trustworth Hardware," *Proceedings of the IEEE*, vol. 102, pp. 1123–1125, Aug. 2014.

[6] T. Sakurai and A. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE Journal of Solid-State Circuits*, vol. 25, pp. 584–594, Apr. 1990.

[7] L. Mostardini, L. Bacciarelli, L. Fanucci, L. Bertini, M. Tonare, and M. De Marinis, "FPGA-based low-cost automatic test equipment for digital integrated circuits," in *2009 IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, pp. 32–37, IEEE, Sept. 2009.

[8] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits*. New York, New York, USA: Kluwer Acedemic Publishers, 2000.

[9] B. Streetman and S. K. Banerjee, *Solid State Electronic Devices*. Upper Saddle River, New Jersey: Pearson Prentice Hall, 6 ed., 2005.

[10] G. Kovačević and B. Pivac, "Structure, defects, and strain in silicon-silicon oxide interfaces," *Journal of Applied Physics*, vol. 115, p. 043531, Jan. 2014.

[11] R. Porat, H. Eshwege, E. Valfer, D. David, D. Pepper, F. Cricchio, B. Hinschberger, and D. Kolar, "Inline Defect Root Cause Analysis of Cu CMP Shorts Using Dual Beam FIB," in *2008 IEEE/SEMI Advanced Semiconductor Manufacturing Conference*, pp. 53–55, IEEE, May 2008.

[12] K. Yamazaki, T. Tsutsumi, H. Takahashi, Y. Higami, H. Yotsuyanagi, M. Hashizume, and K. K. Saluja, "Diagnosing Resistive Open Faults Using Small Delay Fault Simulation," in *2013 22nd Asian Test Symposium*, pp. 79–84, IEEE, Nov. 2013.

[13] C. Di and J. Jess, "An efficient CMOS bridging fault simulator: with SPICE accuracy," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 9, pp. 1071–1080, 1996.

[14] T. Williams and N. Brown, "Defect Level as a Function of Fault Coverage," *IEEE Transactions on Computers*, vol. C-30, pp. 987–988, Dec. 1981.

[15] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," *IEEE Transactions on Computers*, vol. C-30, pp. 215–222, Mar. 1981.

[16] M. Abadir, "Economics modeling of multichip module testing strategies," *IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part B*, vol. 21, no. 4, pp. 360–370, 1998.

[17] M. Taouil, S. Hamdioui, K. Beenakker, and E. J. Marinissen, "Test Impact on the Overall Die-to-Wafer 3D Stacked IC Cost," *Journal of Electronic Testing*, vol. 28, pp. 15–25, Dec. 2011.

[18] N. Touba, "Survey of Test Vector Compression Techniques," *IEEE Design & Test of Computers*, vol. 23, pp. 294–303, Apr. 2006.

[19] A. Veneris, R. Chang, M. Abadir, and M. Amiri, "Fault equivalence and diagnostic test generation using ATPG," in *2004 IEEE International Symposium on Circuits and Systems (IEEE Cat. No.04CH37512)*, vol. 5, pp. V–221–V–224, IEEE, 2004.

[20] V. Iyengar, K. Chakrabarty, and E. Marinissen, "Test wrapper and test access mechanism co-optimization for system-on-chip," in *Proceedings International Test Conference 2001 (Cat. No.01CH37260)*, pp. 1023–1032, IEEE, 2001.

[21] J. Aerts and E. Marinissen, "Scan chain design for test time reduction in core-based ICs," in *Proceedings International Test Conference 1998 (IEEE Cat. No.98CH36270)*, pp. 448–457, Int. Test Conference, 1998.

[22] "IEEE Standard for Reduced-Pin and Enhanced-Functionality Test Access Port and Boundary-Scan Architecture," *IEEE Std. 1149.7-2009*, 2009.

[23] F. C. Hennine, "Fault detecting experiments for sequential circuits," in *1964 Proceedings of the Fifth Annual Symposium on Switching Circuit Theory and Logical Design*, pp. 95–110, IEEE, Nov. 1964.

[24] Y. Zorian, "A distributed BIST control scheme for complex VLSI devices," in *Digest of Papers, 11th Annual IEEE VLSI Test Symposium*, pp. 4–9, IEEE Comput. Soc. Press, 1993.

[25] V. Iyengar, K. Chakrabarty, and E. Marinissen, "Test wrapper and test access mechanism co-optimization for system-on-chip," in *Proceedings International Test Conference 2001 (Cat. No.01CH37260)*, pp. 1023–1032, IEEE, 2001.

[26] S. Koranne, "On test scheduling for core-based SOCs," in *Proceedings of ASP-DAC/VLSI Design 2002. 7th Asia and South Pacific Design Automation Conference and 15h International Conference on VLSI Design*, pp. 505–510, IEEE Comput. Soc, 2002.

[27] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Test access mechanism optimization, test scheduling, and tester data volume reduction for system-on-chip," *IEEE Transactions on Computers*, vol. 52, pp. 1619–1631, Dec. 2003.

[28] M. Nourani and J. Chin, "Power-time tradeoff in test scheduling for SoCs," in *Proceedings of the 21st International Conference on Computer Design*, pp. 548–553, IEEE Comput. Soc, 2003.

[29] R. M. Chou, K. K. Saluja, and V. D. Agrawal, "Scheduling Tests for VLSI Systems Under Power Constraints," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 5, pp. 175–185, June 1997.

[30] Z. He, Z. Peng, and P. Eles, "Power Constrained and Defect-Probability Driven SoC Test Scheduling with Test Set Partitioning," in *Proceedings of Design, Automation & Test in Europe (DATE)*, pp. 1–6, IEEE, 2006.

[31] D. Zhao and S. Upadhyaya, "Dynamically partitioned test scheduling with adaptive TAM configuration for power-constrained SoC testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, pp. 956–965, June 2005.

[32] S. Samii, M. Selkälä, E. Larsson, K. Chakrabarty, and Z. Peng, "Cycle-Accurate Test Power Modeling and Its Application to SoC Test Architecture Design and Scheduling," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, pp. 973–977, May 2008.

[33] P. Girard, N. Nicolici, and X. Wen, *Power-Aware Testing and Test Strategies for Low Power Devices*. Springer, 2010.

[34] P. Tadayon, "Thermal Challenges During Microprocessor Testing," *Intel Technology Journal*, vol. 4, no. 3, pp. 1–8, 2000.

[35] C. R. Kime and K. K. Saluja, "Test Schduling in Testable VLSI Circuits," in *Twenty-Fifth International Symposium on Fault-Tolerant Computing*, (Santa Monica), pp. 406–412, IEEE, 1982.

[36] K. Chakrabarty, "Test scheduling for core-based systems using mixed-integer linear programming," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 10, pp. 1163–1174, 2000.

[37] E. Larsson and H. Fujiwara, "Power Constrained Preemptive TAM Scheduling," in *Proceedings of the 7th IEEE European Test Workshop*, pp. 119–126, IEEE Comput. Soc, 2002.

[38] K. Nose and T. Sakurai, "Optimization of VDD and VTH for Low-Power and High-Speed Applications," in *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 469–474, IEEE, 2000.

[39] X. Kavousianos, K. Chakrabarty, A. Jain, and R. Parekhji, "Test Scheduling for Multicore SoCs with Dynamic Voltage Scaling and Multiple Voltage Islands," in *20th Asian Test Symposium*, pp. 33–39, IEEE, Nov. 2011.

[40] H. Salamy and H. M. Harmanani, "An optimal formulation for test scheduling network-on-chip using multiple clock rates," in *24th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 215–218, IEEE, May 2011.

[41] V. Sheshardi, V. D. Agrawal, and P. Agrawal, "Optimal Power-Constrained SoC Test Schedules With Customizable Clock Rates," in *IEEE International SOC Conference (SOCC)*, (San Jose, CA), pp. 271–276, Oct. 2012.

[42] V. Sheshardi, V. D. Agrawal, and P. Agrawal, "Optimum Test Schedule for SoC with Specified Clock Frequencies and Supply Voltages," in *26th International Conference on VLSI Design and International Conference on Embedded Systems*, (Pune, India), pp. 267 – 272, Jan. 2013.

[43] S. K. Millican and K. K. Saluja, "Formulating Optimal Test Scheduling Problem with Dynamic Voltage and Frequency Scaling," in *22nd AsianTest Symposium (ATS)*, pp. 165–170, IEEE, Nov. 2013.

[44] G. Craig, C. Kime, and K. K. Saluja, "Test scheduling and control for VLSI built-in self-test," *IEEE Transactions on Computers*, vol. 37, no. 9, pp. 1099–1109, 1988.

[45] C. Yao, K. K. Saluja, and P. Ramanathan, "Power and Thermal Constrained Test Scheduling Under Deep Submicron Technologies," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, pp. 317–322, Feb. 2011.

[46] D. R. Bild, S. Misra, T. Chantemy, P. Kumar, R. P. Dick, X. S. Huy, and A. Choudhary, "Temperature-aware test scheduling for multiprocessor systems-on-chip," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 59–66, IEEE, Nov. 2008.

[47] E. J. Marinissen, V. Iyengar, and K. Chakrabarty, "A set of benchmarks for modular testing of SOCs," in *Proceedings of the International Test Conference*, (Baltimore, MD), pp. 519–528, IEEE, Oct. 2002.

[48] S. Davidson, "ITC'99 Benchmark Circuits - Preliminary Results," in *Proceedings of the International Test Conference*, pp. 1125–1125, Int. Test. Conference, 1999.

[49] B. Noia, K. Chakrabarty, and S. Goel, "Test-Architecture Optimization and Test Scheduling for TSV-Based 3-D Stacked ICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 11, pp. 1705–1718, 2011.

[50] B. Sen Gupta, U. Ingelsson, and E. Larsson, "Scheduling Tests for 3D Stacked Chips under Power Constraints," in *2011 Sixth IEEE International Symposium on Electronic Design, Test and Application*, pp. 72–77, IEEE, Jan. 2011.

[51] S. K. Millican and K. K. Saluja, "Optimal Test Scheduling Formulation under Power Constraints with Dynamic Voltage and Frequency Scaling," *Journal of Electronic Testing*, vol. 30, pp. 569–580, Sept. 2014.

[52] S. K. Millican and K. K. Saluja, "Optimal Test Scheduling of Stacked Circuits under Various Hardware and Power Constraints," in *2015 28th International Conference on VLSI Design*, pp. 487–492, IEEE, Jan. 2015.

[53] E. J. Marinissen, C.-C. Chi, J. Verbree, and M. Konijnenburg, "3D DfT architecture for pre-bond and post-bond testing," in *2010 IEEE International 3D Systems Integration Conference (3DIC)*, pp. 1–8, IEEE, Nov. 2010.

[54] P. Rosinger, B. M. Al-Hashimi, and K. Chakrabarty, "Thermal-Safe Test Scheduling for Core-Based System-on-Chip Integrated Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, pp. 2502–2512, Nov. 2006.

[55] S. K. Millican and K. K. Saluja, "Linear Programming Formulations for Thermal-Aware Test Scheduling of 3D-Stacked Integrated Circuits," in *2012 IEEE 21st Asian Test Symposium*, (Niigata, Japan), pp. 37–42, IEEE, Nov. 2012.

[56] N. Vinay, I. Rawaty, E. Larssonz, M. Gaurx, and V. Singh, "Thermal aware test scheduling for stacked multi-chip-modules," in *2010 East-West Design & Test Symposium (EWDTS)*, pp. 343–349, IEEE, Sept. 2010.

[57] K. Skadron, M. Stan, W. Huang, and D. Tarjan, "Temperature-aware microarchitecture," in *30th Annual International Symposium on Computer Architecture, 2003. Proceedings.*, pp. 2–13, IEEE Comput. Soc, 2003.

[58] C. Yao, K. K. Saluja, and P. Ramanathan, "Thermal-Aware Test Scheduling Using On-chip Temperature Sensors," in *2011 24th Internatioal Conference on VLSI Design*, pp. 376–381, IEEE, Jan. 2011.

[59] P. Gschwandtner, T. Fahringer, and R. Prodan, "Performance Analysis and Benchmarking of the Intel SCC," in *2011 IEEE International Conference on Cluster Computing*, pp. 139–149, IEEE, Sept. 2011.

[60] M. Ni, Q. Su, Z. Tang, and J. Kawa, "An Analytical Study on the Role of Thermal TSVs in a 3DIC Chip Stack," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010*, 2011.

[61] J. Cong, L. Guojie, and S. Yiyu, "Thermal-aware cell and through-silicon-via co-placement for 3D ICs," *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*, pp. 670–675, 2011.

[62] Y. Chen, E. Kursun, D. Motschman, C. Johnson, and Y. Xie, "Analysis and mitigation of lateral thermal blockage effect of through-silicon-via in 3D IC designs," in *IEEE/ACM International Symposium on Low Power Electronics and Design*, pp. 397–402, IEEE, Aug. 2011.

[63] P. Ghosal, H. Rahaman, and P. Dasgupta, "Thermal Aware Placement in 3D ICs," in *2010 International Conference on Advances in Recent Technologies in Communication and Computing*, pp. 66–70, IEEE, Oct. 2010.

[64] C. Yao, K. K. Saluja, and P. Ramanathan, "Partition Based SoC Test Scheduling with Thermal and Power Constraints under Deep Submicron Technologies," in *Asian Test Symposium*, pp. 281–286, IEEE, 2009.

[65] S. K. Millican and K. K. Saluja, "A Test Partitioning Technique for Scheduling Tests for Thermally Constrained 3D Integrated Circuits," in *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems*, pp. 20–25, IEEE, Jan. 2014.

[66] Linfeng Chen and Aijiao Cui, "A power-efficient scan tree design by exploring the Q'-D connection," in *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, pp. 1018–1021, IEEE, May 2013.

[67] M. Imhof, C. Zoellin, H. Wunderlich, N. Maeding, and J. Leenstra, "Scan Test Planning for Power Reduction," 2007.

[68] J. Zhang, T. Zhang, and Q. Zuo, "Multi-phase Clock Scan Technique for Low Test Power," in *High Density Design Packaging and Microsystem Integration, 2007 International Symposium on*, pp. 1–5, IEEE, June 2007.