POWER- AND THERMAL-AWARE TESTING OF SYSTEM-ON-CHIP

RAJIT KARMAKAR

POWER- AND THERMAL-AWARE TESTING OF SYSTEM-ON-CHIP

Thesis submitted to the
Indian Institute of Technology Kharagpur
for the award of the degree of

Master of Science (by Research)

by

Rajit Karmakar

Under the guidance of

Prof. Santanu Chattopadhyay

Department of Electronics & Electrical Communication Engineering
Indian Institute of Technology, Kharagpur - 721302
West Bengal, India

July 2015

To my Parents...

APPROVAL OF THE VIVA-VOCE BOARD

/ /20

Certified that the thesis entitled POWER- AND THERMAL-AWARE TESTING OF SYSTEM-ON-CHIP submitted by RAJIT KARMAKAR to Indian Institute of Technology Kharagpur, for the award of the degree of Master of Science (by Research) has been accepted by the external examiners and that the student has successfully defended the thesis in the viva-voce examination held today.

Signature:
Name:

(Member of DAC(PGS&R))

Signature:
Name:

(Member of DAC(PGS&R))

Signature:
Name:

(Member of DAC(PGS&R))

Signature:
Name:

(Supervisor)

Signature:
Name:

(External Examiner)

Signature:
Name:

(Chairman)

CERTIFICATE

This is to certify that the thesis entitled **POWER- AND THERMAL-AWARE TESTING OF SYSTEM-ON-CHIP**, submitted by **Rajit Karmakar** to Indian Institute of Technology Kharagpur, is a record of bona fide research work under my supervision and is worthy of consideration for the award of the degree of Master of Science (by Research) of the Institute.

Santanu Chattopadhyay

Supervisor

DECLARATION

I certify that

a. the work contained in this thesis is original and has been done by me under the guidance of my supervisors.

b. the work has not been submitted to any other Institute for any degree or diploma.

c. I have followed the guidelines provided by the Institute in preparing the thesis.

d. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e. Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Date: Rajit Karmakar

Acknowledgments

The first person I would like to express my deepest appreciation is my advisor, Prof. Santanu Chattopadhyay, whose guidance, encouragements, enthusiasm to high-quality work and insightful advice helped me in all the time of research for and writing of this thesis. I will cherish the experience of learning and working with him forever.

Many thanks also go to the faculty, administrative, technical and non-technical members in Dept. of Electronics and Electrical Communication Engineering, whose assistance was vital for this research. During the course of work, I had numerous pleasant moments and experiences with several friends and colleagues. Therefore, I would like to thank everybody who has supported me over the last two years.

I must thank my all lab members Priyajit da, Novonil da (kaka), Arpita (hubba), Prasad, Soumya di, Kanchan da, Anupam da, Saptak da, Avishek, Aditya. Without you people it would have been extremely difficult for me to continue my research works with full enthusiasm even in tough times. Adda with tea at Bhabo da's canteen with you people have been extremely refreshing. A special mention to Priyajit da's late night melodious songs, kaka's all time non-sence and obviously Arpita's fighting with me in the lab. All these funny things have given me extra enthusiasm and made my kgp days more colourful.

I must thank my roommates Sumant, Tushar da, Vashim, my previous roommates Pankaj da, Vasu, who have been extremely cooperative throughout my stay at VSRC.

Finally, I can never give enough thanks to my parents, for their constant encouragement and unconditional support throughout the entire tenure of my research work. Without their support, I could not have completed my research work.

Rajit Karmakar

List of Abbreviations

IC	–	Integrated Circuit
SSI	–	Small-Scale Integration
VLSI	–	Very-Large-Scale Integration
SoC	–	System-on-Chip
IP	–	Intellectual Property
APU	–	Application Processor Unit
GPU	–	Graphics Processor Unit
PCB	–	Printed Circuit Board
DFT	–	Design for Testability
ATE	–	Automatic Test Equipment
CUT	–	Circuit Under Test
TAM	–	Test Access Mechanism
TAT	–	Test Application Time
PSO	–	Particle Swarm Optimization
LPT	–	Large Processing Time
FFD	–	First Fit Decreasing
BFD	–	Best Fit Decreasing
ILP	–	Integer Linear Programming
EA	–	Evolutionary Algorithm
SA	–	Simulated Annealing
GA	–	Genetic Algorithm
ACO	–	Ant Colony Optimization
SFLA	–	Shuffle Frog-Leaping Algorithm
TDM	–	Test Data Multiplexer
TDdeM	–	Test Data Demultiplexer
CLP	–	Constraint Logic Programming
MILP	–	Mixed Integer Linear Programming
AOFF	–	Abort-On-First-Fail
ETAT	–	Expected Test Application Time
BIST	–	Built-In-Self-Test
FDR	–	Frequency Directed Run
VIHC	–	Variable Length Input Huffman Coding
RBR	–	Run Based Reordering
LFSR	–	Linear Feedback Shift Register
EDT	–	Embedded Deterministic Test
ATPG	–	Automatic Test Pattern Generation
UDL	–	User Defined Logic
SC	–	Scan Chain
PI	–	Primary Input

PO	–	Primary Output
Bidir	–	Bidirectional Line
BP	–	Break Point List
ATW	–	Available TAM Width Info
PT	–	Power Tracker
GP	–	Global Peak
TP	–	TAM-dependent Peak
WP	–	Window-based Peak
GPMF	–	Global Peak Multi Frequency
WPSF	–	Window-based Peak Single Frequency
WPMF	–	Window-based Peak Multi Frequency
ATR	–	Average transition
DP	–	Dynamic power
TD	–	Thermal Database
CR	–	Cooling Rate
TG	–	Time Gap
TT	–	Temperature Tracker
PVC	–	Power Violation Checker
TVC	–	Thermal Violation Checker
CL	–	Clique List
SI	–	Slice Info
CR	–	Compression Ratio
TR	–	Temperature Reduction
TR	–	Temperature Reduction

List of Symbols

$\in$	–	Belongs to
$\subset$	–	Subset of
$\forall$	–	for all
$\exists$	–	there exists
$=$	–	is equal to
$\neq$	–	does not equal
$\rightarrow$	–	Exclusive-or sequence
$\leq$	–	Less or equal to
$<$	–	Less than
$>$	–	Grater than

Abstract

System-on-Chip (SoC) paradigm has evolved to address the problem of high design turnaround time of complex VLSI systems. With rapid growth in VLSI industry, large numbers of IP cores are being integrated onto a single chip. Testing of SoCs has emerged as a new dimension for the test engineers to explore. Apart from ensuring high fault coverage, it has also become necessary to look into several emerging problems like test power, temperature, test data volume etc. This work addresses the issues of high power consumption and rise in temperature during testing and also the memory requirement to store the huge amount of test data.

Main objective in SoC testing is to reduce the Test Application Time (TAT) required to test all the cores in the SoC. This work proposes a test architecture optimization and test scheduling strategy that uses the available test resources efficiently to reduce the test cost, without violating system level resource, power and thermal constraints. While the existing approaches use either a fixed power value for the entire test session of a core, or the cycle-accurate power values, the proposed work divides the power profiles of cores into fixed-sized windows. This reduces the number of power values to be handled by the test scheduling approaches, while reducing the amount of pessimistic over-estimations of instantaneous power. As a result, the power model can be integrated with more exhaustive meta-search techniques for generating power constrained test schedules. Test Application Time (TAT) can be further improved by using a multi-frequency test approach, compared to single-frequency one. A superposition principle-based thermal model has been incorporated with the test scheduling algorithm to ensure thermal safety during testing. Although this model avoids run-time thermal simulations, it has the capability of estimating temperature of a core very fast with high degree of accuracy. These two proposed power and thermal models, integrated with a Particle Swarm Optimization (PSO) based test scheduling technique, generate a power and thermal safe test schedule for SoC. This work also proposes a dictionary based test data compression technique. While most of the works addresses power, our work performs a trade-off between compression and temperature, that can be used as a tool by the test engineers to get how much compression is sufficient considering the thermal constraints and also the constraints from automated test equipment (ATE). Finally a novel thermal-aware test data compression scheme has been proposed, which fills the don't-care bits present in the test patterns, in a way to help reduce the temperature without compromising the compression factor.

Keywords: System-on-Chip; Test scheduling; Test Application Time; Particle Swarm Optimization; Window-based peak power model; Superposition principle based thermal model; Test data compression; Don't-care bit; Clique partition.

Contents

Certificate of Approval vii

Certificate ix

Declaration xi

Acknowledgments xiii

List of Abbreviations xv

List of Symbols xvii

Abstract xix

1 Introduction 1

1.1 System-on-Chip . . . 1
1.2 Testing Of System-on-Chip . . . 3
1.2.1 Primary Test Challenges For System-on-Chip . . . 3
1.2.1.1 **Infrastructure To Make A Core Testable** . . . 5
1.3 Motivation And Objective Of The Thesis . . . 5
1.4 Work Carried Out In The Thesis . . . 7
1.4.1 Power-Aware Test Scheduling . . . 7
1.4.2 Thermal-Aware Test Scheduling . . . 7
1.4.3 Thermal-Aware Test Data Compression . . . 8
1.5 Contribution Of The Thesis . . . 9
1.6 Organization Of The Thesis . . . 9
1.7 Conclusion . . . 10

2 Background And Related Work 11

2.1 Introduction . . . 11
2.2 Test Architecture Optimization And Test Scheduling . . . 11
2.2.1 Wrapper Design . . . 11
2.2.2 Test Access Mechanism (TAM) . . . 13
2.2.3 Test Scheduling . . . 13
2.3 Prior Work On Power-Aware Testing . . . 15
2.4 Prior Work On Thermal-Aware Testing . . . 16
2.5 Prior Work On Test Data Compression . . . 19
2.5.1 Different Test Data Compression Strategies . . . 20
2.5.1.1 **Code-Based Schemes** . . . 21

2.5.1.2 **Linear-Decompression Based Schemes** 22
2.5.1.3 **Broadcast Scan Based Schemes** 23
2.5.2 Power-Aware Test Compression Techniques 23
2.6 Conclusion 24

3 Power-Aware Test Scheduling 25
3.1 Introduction 25
3.2 Overview Of The Problem 27
3.3 Wrapper Design 28
3.4 Window-Based Peak Power Model 29
3.5 Multi-frequency Test Infrastructure 34
3.5.1 Multi-frequency architecture 35
3.6 Test Scheduling 36
3.6.1 Problem Formulation 36
3.6.2 Test Schedule Generation 37
3.6.3 Particle Swarm Optimization 38
3.7 Experimental Results 43
3.7.1 Test Schedule Without Power Constraints 44
3.7.2 Test Scheduling With Global Peak Power Model 45
3.7.3 Test Scheduling With Window-Based Peak Power Model 48
3.7.4 Test Scheduling In Multi-Frequency Test Environment 54
3.8 Conclusion 56

4 Thermal-Aware Test Scheduling 57
4.1 Introduction 57
4.2 Motivation 58
4.3 Summary Of Contribution 58
4.4 Superposition Principle Based Thermal Model 60
4.5 Test Scheduling Strategy 64
4.5.1 Problem Formulation 64
4.5.2 Test Schedule Generation 64
4.6 Experimental Results And Discussions 72
4.6.1 Experimental Setup 73
4.6.2 Experimental Results 75
4.6.2.1 **Thermal-Aware Test Scheduling** 76
4.6.2.2 **Comparison Between Thermal Models** 76
4.7 Conclusion 81

5 Thermal-Aware Test Data Compression Using Dictionary Based Coding 83
5.1 Introduction 83
5.2 Background Of Dictionary Based Test Data Compression 84
5.3 Motivation 86
5.4 Variation Of Temperature And Compression Ratio For Different Clique Partitions 87
5.4.1 Thermal Simulation Steps 89
5.5 Observation 90
5.6 A Trade-off Between Temperature And Compression Ratio 91
5.6.1 Experimental Results 96

5.7 Temperature Reduction Without Compromising Compression Ratio . . 97
5.7.1 Experimental Results 101
5.8 Conclusion 106

6 Conclusions and Future Works **107**
6.1 Conclusions 107
6.2 Future Works 108

Bibliography **113**

List of Figures

1.1 Microprocessor transistor counts 1971-2011 & Moore's Law [2] 2
1.2 An example SoC with different components [42] 3
1.3 Design Development Strategies for PCB and SoC [149] 4
1.4 Test infrastructure for SoC 6

2.1 Internal structure of wrapper[88] 12
2.2 A typical example of fixed-width TAM architectures 14
2.3 A typical example of flexible-width TAM architectures 14
2.4 Temperature profile and maximum temperature limit of core 2 of $k10$ using the thermal model [137, 136] 19
2.5 Test data compression[124] 20

3.1 Power profiles of module no 1 of $d695$ for different wrapper configurations 30
3.2 Comparison of window-based peak power model with other power models 31
3.3 Variation of power-profiles of core number 6 of SoC $d695$ (wrapper length 18) for different window-sizes 32
3.4 Variation of schedule generation time with window-size for $p93791$ for a single iteration 33
3.5 Test schedule using global peak power model 33
3.6 Test schedule using window-based peak power model 34
3.7 Multi-frequency test architecture 36
3.8 Sample particle structure of 4 cores with W_{max} =16 ($B = 4$) (for single-frequency test environment) 39
3.9 Sample particle structure of 4 cores with W_{max} =16 ($B = 4$, $F = 5$) (for multi-frequency test environment) 39
3.10 Evolution of a particle (i) current particle, (ii) local best and (iii) evolved particle (for single-frequency test environment) 41
3.11 Evolution of a particle (i) current particle, (ii) local best and (iii) evolved particle (for multi-frequency test environment) 41
3.12 Scheduling of $d695$ with $P_{max} = 2000$ and $W_{max} = 64$ (window-based peak power model with single frequency test environment) 54

4.1 Temperature profile and maximum temperature limit of core 2 of $k10$ using the thermal model [137, 136] 59
4.2 Pre-scheduling increase of the initial temperature of core 1 of SoC $k10$ due to leakage power 61
4.3 Temperature profile of core 1 of SoC $k10$ during its scheduling 62
4.4 Temperature profile of core 1 as a effect of core 2 schedules before core1 of SoC $k10$ 62

4.5 Different possibilities of temperature effect of core C_j on core C_i . . . 65
4.6 Different conditions of test parallelism between two cores C_j and C_i . . 65
4.7 SoC design set-up flow 74
4.8 Exact floorplan of $k10$ and $k25$ obtained using $Blobb$ and $Plottool$ . . 75
4.9 Graphical representation of the variation in test application time (TAT) for different SoCs with the variation of P_{max} and T_{max} 77
4.10 Temperature profile and maximum temperature limit of core 2 of $k10$ using our thermal model 81

5.1 An example of test slices 85
5.2 Architecture of dictionary coding based test data compression scheme[81] 86
5.3 An example of test data for multiple scan chain set-up and corresponding compatibility graph. 89
5.4 Different clique-sets for different start nodes 90
5.5 Flow of thermal simulation 91
5.6 Variation of temperature for different clique partitions 93
5.7 Variation of compression ratio for different clique partitions 94
5.8 Variation of temperature and CR with Wt for different scan chain set-up for different ISCAS'89 and ITC'99 benchmarks. 100
5.9 An example flow of thermal-aware test data compression 103

List of Tables

3.1 Variation Of Different Test Parameters With Operating Frequency Of Circuit Under Test 35
3.2 Comparisons Of PSO With Other Scheduling Procedure For Different ITC'02 SoCs For Different W_{max} Without Power Constraints 45
3.3 Comparisons Of PSO With Other Scheduling Procedure For Different ITC'02 SoCs For Different W_{max} Without Power Constraints (Contd.) . 46
3.4 Comparisons Of PSO With Other Scheduling Procedure For $d695$ With Different P_{max} And W_{max} (Considering Global Peak Power Model) . . 47
3.5 Comparisons Of PSO With Other Scheduling Procedure For $p22810$ With Different P_{max} And W_{max} (Considering Global Peak Power Model) 48
3.6 Comparisons Of PSO With Other Scheduling Procedure For $p93791$ With Different P_{max} And W_{max} (Considering Global Peak Power Model) 49
3.7 Comparisons Of PSO With Other Scheduling Procedure For $h953$ With Different P_{max} And W_{max} (Considering Global Peak Power Model) . . 50
3.8 Comparison Of CPU Time For Different Power Models For All ITC'02 Benchmarks (Single Particle, Single Generation) 50
3.9 Comparisons Of GP And TP With WP Using PSO For Different ITC'02 SoCs For Different P_{max} And W_{max} 51
3.10 Comparisons Of GP And TP With WP Using PSO For Different ITC'02 SoCs For Different P_{max} And W_{max} (Contd.) 52
3.11 Comparisons Of GP And TP With WP Using PSO For Different ITC'02 SoCs For Different P_{max} And W_{max} (Contd.) 53
3.12 Comparison Of Different Test Scheduling Approaches For Different P_{max} And W_{max} For ITC'02 Benchmarks 55

4.1 Detail Information Of The Cores Of Newly Formed SoCs 75
4.2 Variation In Test Application Time (TAT) For Different SoCs With The Variation Of P_{max} And T_{max} 77
4.3 Comparison Of Validation Of Test Schedule Of Our Thermal Model With The Thermal Model Presented In [137, 136] 79
4.4 Comparison Of Validation Of Test Schedule Of Our Thermal Model With The Thermal Model Presented In [137, 136] (Contd.) 80

5.1 Clique Set Information For Different Scan Chains 92
5.2 Variation Of Temperature And Compression Ratio For Different Clique Selections (For D =128) 92
5.3 Variation Of Temperature And Compression Ratio For Different Values Of Wt For Different Scan Chains (For D = 128) 98

5.4 Variation Of Temperature And Compression Ratio For Different Values Of Wt For Different Scan Chains (For D = 128) (Contd.) 99
5.5 Reduction In Temperature For Same Compression Ratio And Improvement Of CR For Same Temperature For Different Benchmarks 105

Chapter 1
Introduction

With increasing demand for high performance and low-power chips, present day's semiconductor industry is heading towards smaller feature size with reduced chip area. Device dimensions are reducing drastically, while the system design is becoming more and more complex. According to Gordon Earle Moore, the scale of Integrated Circuits (ICs) is expected to double every 18 months [4, 2]. Although this observation was made by him in the year 1965, the prediction, known as Moore's Law, has been proven accurate. Semiconductor industry still follow the trend and is expected to continue for the foreseeable future as well. In early 1960s Small-Scale Integration (SSI) devices merely had tens of transistors, while in the early 1980s, Very-Large-Scale Integration (VLSI) devices with hundreds of thousands of transistors were introduced. Figure 1.1 is a plot of CPU transistor counts against dates of introduction. It shows that the exponential growth in transistor count in every two years.

In 1960s, testing of a chip with tens of transistors was not a big issue, however present days ICs with hundreds of millions of transistors in a single device unfolds several testing challenges for the test engineers. Increasing device density and the reduction in feature size increases the probability that a manufacturing defect in the IC will result in a faulty chip. To ensure that a chip is defect free, a test engineer has to test it for different probable faults, such as, delay faults, stuck-at-faults, stuck-open-faults, transient faults and some other subtle faults. The amount of test data needed to improve fault coverage also increases manifold, increasing total testing time. Testing is one of the most important parts of the VLSI design flow as it occupies about 60% of the total design-turnaround time.

1.1. System-on-Chip

Driven by the market demands, electronics industry continuously requires products with greater flexibility in terms of higher reliability, improved functionality, lower cost and shorter time-to-market. The need of smarter, faster and smaller products motivates the system designers to develop complex chips with a wide

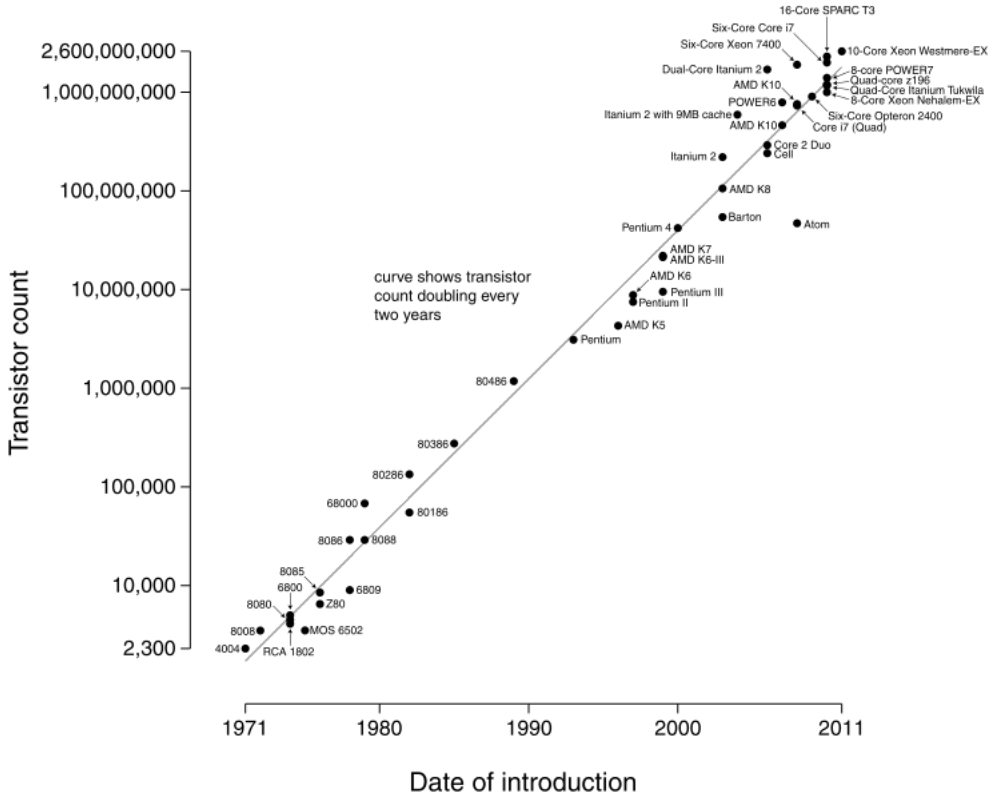


Figure 1.1: Microprocessor transistor counts 1971-2011 & Moore's Law [2]

range of functions that build an entire system into a single die, called System-on-Chip (SoC). SoC paradigm has evolved to address the problem of high design turnaround time of complex VLSI systems. Design time can be reduced by following the core-based system design paradigm. Pre-designed logic blocks (processor, memory etc.), commonly known as Intellectual Property (IP) cores, are integrated on the same silicon floor by the system integrator. Compared to a board-based system, in SoC, the communication between the system components has become on-chip. This aids in improving the system performance.

The generic definition of a SoC is the on-chip integration of a variety of functional hardware blocks to suit a specific product application. A SoC can thus be as simple as a basic connectivity chip which combines some mixed-signal and digital circuitry. A more complex SoC may include the on-chip integration of an Application Processor Unit (APU) and a Graphics Processor Unit (GPU). Even more functional SoCs further integrate various other hardware blocks (e.g. image processor, audio/video decoder and modem). It is this ability to continue to integrate disparate functionality on a chip that has enabled SoC technology to rapidly evolve from supporting a simple feature-phone to a smart-phone and all the way to a tablet computer. Figure 1.2 shows an example SoC with different components as cores.

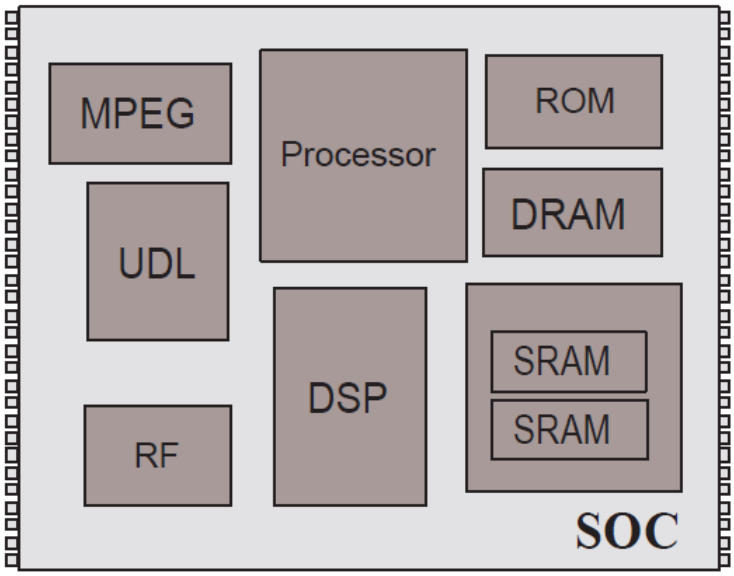


Figure 1.2: An example SoC with different components [42]

1.2. Testing Of System-on-Chip

Although a SoC is made by simple "plug-and-play" of cores to make functions similar to the traditional Printed Circuit Board (PCB) design or System-on-Board (SoB) design, the SoC paradigm brings many challenges. For SoB design, the ICs are manufactured and tested individually before being integrated into a PCB. So, the system integrators may assume that the ICs are fault free and have to be concerned only about the testing of the interconnects. However, in SoC design, the embedded cores are not yet manufactured when the system integrator puts them together; thus the core vendors cannot test their products before the chip is actually fabricated. Therefore, SoC test engineers are responsible for manufacturing and testing of not only the interconnects between the cores, but also the cores themselves. Figure 1.3 shows the difference between the design development strategies of PCB and SoC.

1.2.1. Primary Test Challenges For System-on-Chip

SoC test development is especially challenging for several reasons.

1. Generally, in SoC, embedded cores are delivered from different vendors, mostly are not integrated with on-chip test generator. Also core vendors are reluctant to divulge structural information about their cores to users. Thus, users cannot access core netlists and insert design-for-testability (DFT) hardware that can ease test application from the surrounding logic. Instead, the core vendor provides a set of test patterns that guarantees a specific fault

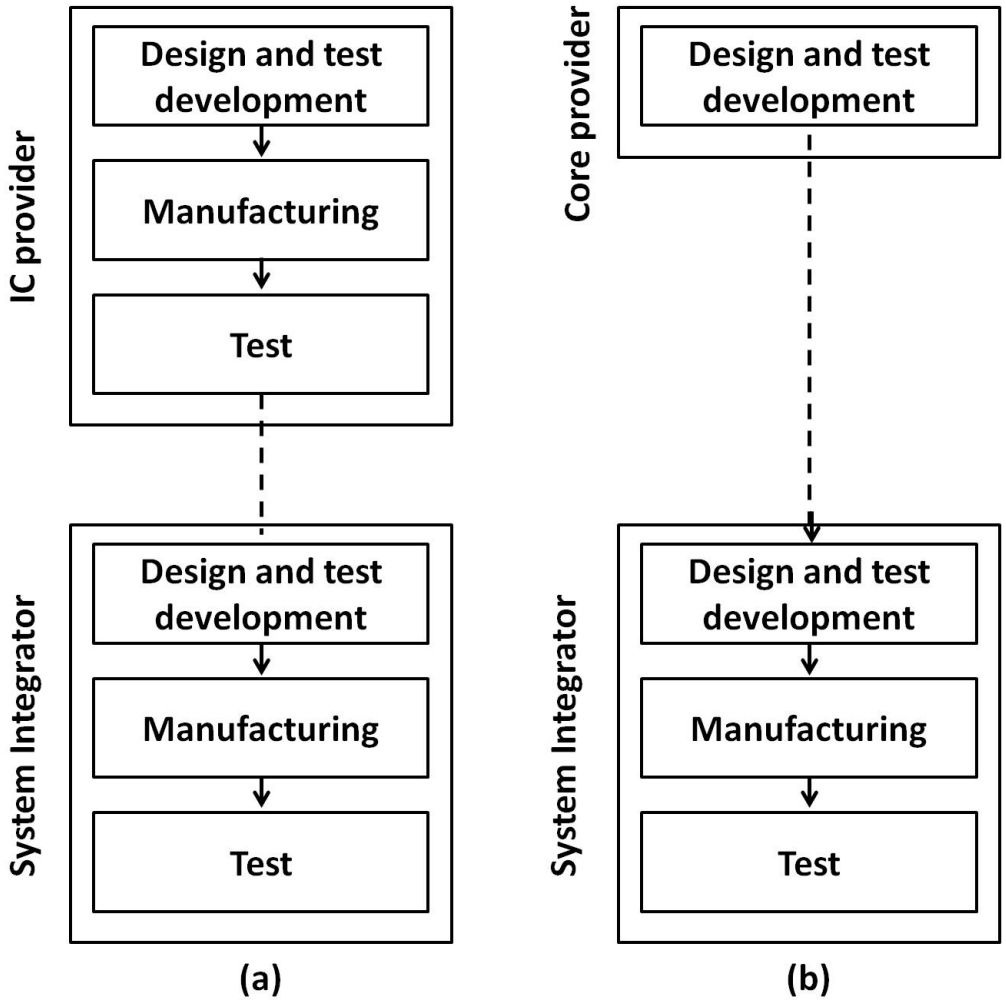


Figure 1.3: Design Development Strategies for PCB and SoC [149]

coverage. To test a chip, system integrator applies all the test patterns specified by the vendors of individual cores.

2. With increasing complexity of present day's integrated circuits (IC), the amount of test data needed to test a SoC has increased manifold. Automatic Test Equipment (ATE), which stores and transports test data to the cores, has to take care of this large volume of test data, increasing the test data storage cost.

3. Cores are often embedded in several layers of user-designed or other core-based logic and no physical access to the pins of the cores is possible from chip I/Os. So it is needed to provide a test access path from the SoC primary pins to the embedded cores and vice-versa with sufficient bandwidth to fulfill the test requirements of the cores.

1.2.1.1. Infrastructure To Make A Core Testable

The followings are the primary infrastructures that are required to make a core testable.

- **Source and Sink:** Source and sink are mainly used to transport the test stimuli to the core under test (CUT) and to analyze the test responses coming out of the CUT. These source and sink can be either on-chip or off-chip. On-chip source and sink refer to the BIST-ed cores, where test patterns are generated internally using on-chip test generator, while for off-chip source and sink, both are represented by the Automatic Test Equipment (ATE). For these kind of cores, ATE stores the test patterns provided by the core vendors.

- **Test Access Mechanism (TAM):**

  Test Access Mechanism (TAM) need to be designed to transport test stimuli from the source (ATE) to the core under test (CUT), and also the responses from the core to the sink (ATE). Basically a TAM is a set of wires that makes physical communication between the sources, core and sink.

- **Test Wrapper:**

  To simplify test access and test application, a common practice is to perform modular testing. To facilitate modular test, an embedded core must be isolated from the surrounding logic via a test wrapper. A wrapper for a core is a thin shell around the core that acts as a communication bridge between the TAM and the core. It provides width adaptation between core I/O pins and TAM pins.

Figure 1.4 shows a typical example of test infrastructure for SoC.

1.3. Motivation And Objective Of The Thesis

The primary objective of testing is to improve fault coverage. However, in SoC, cores are provided with their respective test pattern sets ensuring a specified fault coverage. So, the system integrator does not have much to do with the fault coverage. Rather, his main objective is to finish the testing of all the cores as early as possible, reducing the Test Application Time (TAT) and test cost. Test scheduling of all the cores is required to make testing faster to meet the shrinking product development schedule. Hence, test scheduling is one of the major challenges for the SoC test engineers.

Continuous scaling of the feature size of CMOS technology has resulted in exponential growth in device density, thus enabling more functionality to be placed

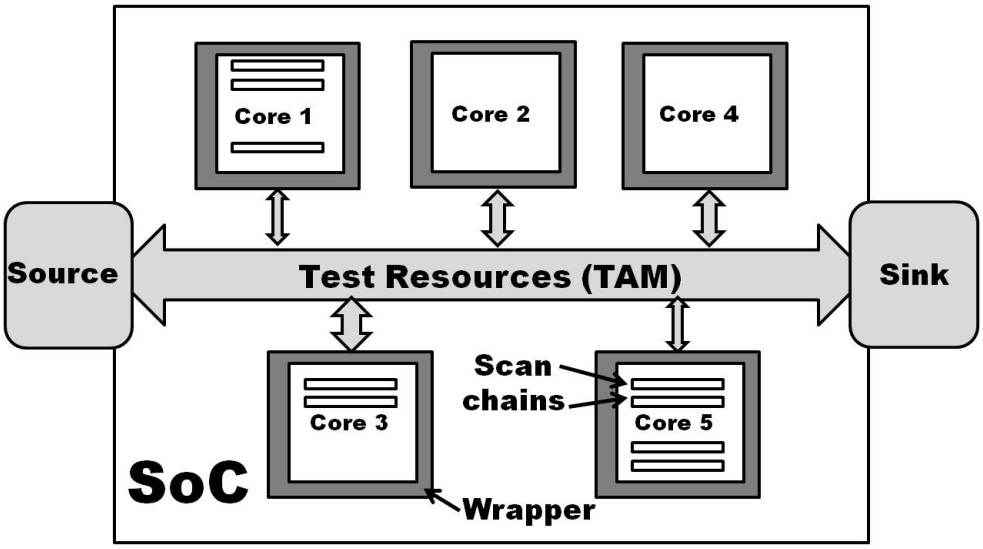


Figure 1.4: Test infrastructure for SoC

on a silicon die. At the same time, it has increased the power consumption of the circuit manifold. Moreover, test mode power is about 30 times higher than functional mode of operation. Managing the power consumption of VLSI systems is now considered to be one of the most important challenges and must be taken into consideration when developing test infrastructure of SoC.

Temperature of the chip during testing is another major concern of the test engineers. High power density of a certain core may cause local heating, which leads to high temperature of that core, while the overall temperature of the chip may be relatively low. These local-hotspot cores may cause overheating, hence, permanent damage or burning of the chip. Thus, ensuring thermal safety cannot be ignored.

To meet the increasing demand of the users, more number of cores with lots of functionality, are being integrated into a single chip. The amount of test data to test all these complex cores increases dramatically. As most of the cores are not provided with on-chip tester, the test patterns of these cores have to be stored in the ATE, increasing the memory requirement as well as ATE cost. To cope up with the increasing test data volume, upto 60% of the invested capital for ATE upgrade is required to meet the new memory capacity requirement. Test data has to be stored in the memory in a compressed manner to reduce the ATE cost and also the test application time.

The objective of this research work is to develop a SoC test scheduling strategy, which can minimize TAT without violating the resource, power and thermal constraints. In recent times Particle Swarm Optimization [66] has evolved as a new evolutionary search strategy that often performs better than other evolutionary approaches because of its faster convergence towards the optimal solution. We

have used PSO based evolutionary technique to guide the scheduling algorithm. A power model consisting of sufficiently detailed power information with a feasible computational overhead and an accurate thermal model have been integrated with the test scheduling strategy. The other objective of this work is to propose a test data compression technique, which can handle high temperature issues during testing. The flexibility of our proposed method helps to choose a suitable balance between thermal safety and test cost to store huge amount of test data by efficient filling of don't care bits in test patterns.

1.4. Work Carried Out In The Thesis

The work carried out in the thesis can be summarised as follows.

1.4.1. Power-Aware Test Scheduling

This work addresses the issue of power-aware test scheduling of cores in a SoC. While the existing approaches either use a fixed power value for the entire test session of a core or cycle-accurate power values, the proposed work divides the power profiles of cores into fixed-sized windows. This approach reduces the number of power values to be handled by the test scheduling algorithms while reducing the amount of pessimistic over-estimations of instantaneous power consumption. As a result, the power model can be integrated with more exhaustive meta-search techniques for generating power constrained test schedules. The proposed power model has been integrated with a Particle Swarm Optimization (PSO) based 3D-bin packing technique to generate test schedules. Experimental results prove the quality of the approach to be high, compared to the existing scheduling techniques.

As present day's cores are provided with the flexibility of multi-frequency operation, shifting of test data at different frequency levels in those cores is also possible. In multi-frequency test environment, the cores can operate at different frequency levels while the ATE, which transports the test data to the cores, operate at a single frequency. So some architectural modification is needed to synchronize the data rate between the cores and the ATE. To provide multi-frequency test facility, we have extended our power-aware test scheduling to the multi-frequency one. A new simple multiplexer based architecture capable of sending test data at different frequencies, has been proposed to facilitate multi-frequency test infrastructure.

1.4.2. Thermal-Aware Test Scheduling

Temperature of cores in the scheduling interval is required to get a thermal safe test schedule. It requires thermal simulation of the cores at the time of scheduling. However, one major drawback of the thermal simulators like HotSpot [118] is

their execution time, which restricts us from invoking it inside the scheduling procedure. An alternate solution is to incorporate some kind of thermal model in the scheduling procedure, which can predict the temperature of the cores during scheduling without thermal simulations.

Thermal simulators like Hotspot [118] follow linear RC thermal model [137]. We have used a superposition principle based thermal model, which takes the help of linearity of thermal model of HotSpot. Our thermal model uses offline HotSpot [118] thermal simulations for each core in different conditions and creates thermal databases for all those conditions. These database information are used for the scheduling purpose. The superposition principle based thermal model produces the same temperature as generated by Hotspot. It helps us to generate thermal safe test schedule much faster than the techniques that use online Hotspot simulations. Post scheduling thermal simulations show that our thermal model does not violate thermal constraint in a single case. This thermal model and the window-based power model have been integrated with the PSO guided 3D-bin packing approach to generate thermal-safe test schedule.

To get exact thermal behaviour of cores in a chip, the detail of floorplan, area, test patterns are required. However, ITC'02 benchmarks contain none of the above mentioned details for the SoCs. For this, we have developed two SoCs named $k10$ and $k25$ having 10 and 25 cores respectively using ITC'99, ISCAS'89 and ISCAS'85 circuits. The newly formed SoCs have every detail regarding test vectors, area, floorplan, dynamic and leakage power information of each core. These helps to observe exact thermal behaviour of the chip.

1.4.3. Thermal-Aware Test Data Compression

The don't-care bits present in test patterns can be filled in two ways- 1) compression-aware filling and 2) thermal-aware filling. Experimentation shows that, although compression-aware don't-care filling produces high compression ratio, it fails to reduce temperature. Similarly, thermal-aware don't-care filling may reduce temperature, but produces poor compression ratio. In this work, we have combined both temperature reduction and compression into a single problem and solved it. We present an intermediate approach that performs a trade-off between temperature and compression ratio.

Clique partitioning guided dictionary based coding technique has been adopted for test data compression. Any clique partitioning technique uses the flexibility of don't-care bits to get the maximum number of matching test sub-vectors. Most of the don't-care bits present in original patterns get filled up (either 0 or 1) at the time of generation of final patterns after clique partitioning.

We have filled some of the don't-cares considering compression-aware filling

and the rest are filled targeting thermal-aware filling. The number of don't-care bits that are filled up using compression-aware or thermal-aware approach, has been decided using a weight factor. Experimental results on ISCAS'89 and ITC'99 benchmarks show that a balance between temperature and compression ratio can be achieved by the proposed method. We have also proposed a temperature reduction technique that can reduce the temperature upto a certain limit without compromising in the maximum compression ratio achieved by compression-aware don't-care bit filling.

1.5. Contribution Of The Thesis

- Surveyed the techniques for power and thermal-aware SoC test scheduling and test data compression techniques.
- Proposed a new power model, named window-based peak power model, to estimate the power profiles of the cores during testing.
- Proposed a superposition principle based thermal model to estimate temperature of the cores during testing, without online thermal simulations.
- Proposed a PSO guided 3D-rectangular bin packing heuristic for test scheduling. The proposed power and thermal models have been integrated with this PSO guided 3D-bin packing heuristic to generate power and thermal-safe test schedules.
- Proposed a novel thermal-aware test data compression technique.

1.6. Organization Of The Thesis

The rest of the thesis is organized as follows.

▷ **Chapter 2:** This chapter presents a survey of the existing SoC test architecture development and test scheduling strategies under power and thermal constraints. It also discusses different test data compression and power management techniques at the time of data compression reported in the literature.

▷ **Chapter 3:** A PSO guided 3D-bin packing based power-aware test scheduling strategy has been described in this chapter. A new multi-frequency test infrastructure has been proposed to facilitate multi-frequency testing. Finally we have shown the experimental results for ITC'02 benchmark suite. The result section covers both single-frequency and multi-frequency test scheduling results with and without power constraints.

- ▷ **Chapter 4:** This chapter presents a thermal-aware test scheduling strategy for SoC. A new superposition principle based thermal model has been proposed. Formation of two new SoCs using ISCAS'85, ISCAS'89 and ITC'99 benchmarks as cores, have been included.
- ▷ **Chapter 5:** A thermal-aware test data compression technique has been proposed in this chapter.
- ▷ **Chapter 6:** This chapter summarizes the contribution of this research and presents outlines of future problems.

1.7. Conclusion

High power consumption and temperature are the two major challenges that the VLSI test engineers are facing at the time of development of test infrastructure. The test engineers not only have to reduce the test application time, but also honour the SoC power and temperature limits. Test architecture design and test scheduling itself is an NP-hard problem. Extra power and thermal constraints make the problem more complex. In this research work, we have tried to solve this problem using a PSO guided 3D-bin packing heuristic with the integration of our proposed power and thermal models to take care of power and thermal issues during testing. We have also proposed a dictionary based test data compression technique that can produce a test pattern set with high compression ratio and low temperature. The next chapter reviews the works reported in the literature, in this domain.

Chapter 2
Background And Related Work

2.1. Introduction

Increasing complexity of core-based SoCs has increased the test complexity and test cost. Test engineers have to take care of several test parameters like test resources, test power and temperature etc. at the time of development of test architecture for the SoC. Huge test data volume also has been a major concern now a days. For each and every step of testing, it is needed to reduce the test cost by proper utilization of all the resources. Hence, a vast amount of research has endeavored to provide a better understanding of these areas. A large number of test strategies and algorithms have been developed to reduce the test cost.

In the following section, we have presented a detailed survey of the existing techniques corresponding to each of the areas considered in this research work.

2.2. Test Architecture Optimization And Test Scheduling

Test architecture optimization and test scheduling are the most important parts of SoC testing. An extensive research has been carried out in this area and still it is the subject of interest for the researchers. For a SoC with specified parameters of its cores and given the system level values of maximum utilizable test resources, maximum tolerable power and thermal limits, a test architecture and a test schedule are needed to be designed carefully so as to minimize the test cost like test application time and test area overhead.

Since wrapper design, TAM optimization and test scheduling are correlated to each other, the review has been started with the wrapper design optimization.

2.2.1. Wrapper Design

To make an embedded core testable, it must be isolated from surrounding logic, and test access must be provided from the I/O pins of the SoC. Test wrappers are used to isolate the core. Design of test wrapper is important during system inte-

gration since it has direct impact on test time. IEEE Std. 1500 [91] standardizes the wrapper interfaces. Therefore, the internal structure of the wrapper can be adapted to the specific SoC test requirement.

Wrapper consists of core internal scan chains, primary inputs and primary outputs. Wrapper design mainly attaches internal scan chains of the cores and few of the input and output cells to form wrapper chains, with an objective to minimize core test time. Figure 2.1 shows an example construction of wrapper. Core test application time depends on the length of the wrapper scan chains. The wrapper chain of maximum length determines the core test application time. The main objective in wrapper optimization is to minimize the test time.

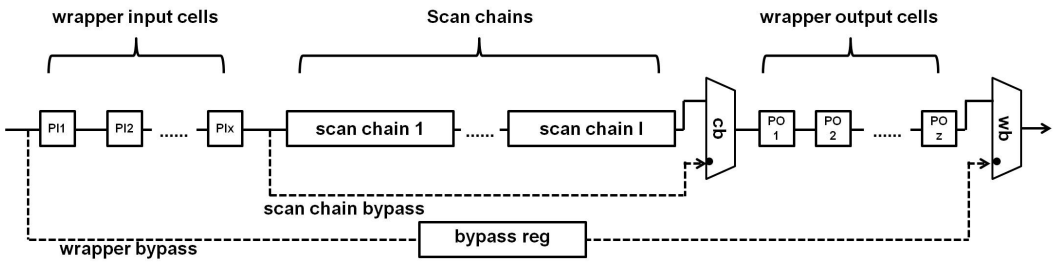


Figure 2.1: Internal structure of wrapper[88]

Wrapper design can be of two types; 1) *Balanced Wrapper* and 2) *Unbalanced Wrapper*. The wrapper design problem has been first addressed in [88], where the authors have proposed two polynomial time heuristic approaches to design balanced wrapper scan chains. The LPT (Large Processing Time) approach has been adopted to solve the wrapper design problem in a very short computational time. Another heuristic, named COMBINE, that obtained better results by using LPT as a starting solution, followed by a searching method (linear search) over the wrapper scan chain lengths with the First Fit Decreasing (FFD) heuristic has also been presented in this work. *Design_Wrapper* algorithm, based on the Best Fit Decreasing (BFD) heuristic, has been proposed in [57] to partition the scan chain elements into several wrapper scan chains. Lengths of wrapper chains have been kept as equal as possible to create balanced chains. Authors have also shown that the test application time varies with the TAM width in a "staircase" manner. These discrete widths are called *pareto-optimal* points.

However, the authors in [150] have shown that the *Design_Wrapper* algorithm does not always give lower test application time for the cores having no internal scan chain. They have also proposed an unbalanced wrapper design to get lesser test time for this kind of cores.

Test wrapper design for cores in a hierarchical SoC has been presented in [111]. For hierarchical SoC, core itself can have a core embedded within it. Both the parent and child cores can be tested using different test modes. A reconfigurable

core wrapper approach with the flexibility of dynamic change in TAM width during the execution of core test to improve test schedule has been presented in [72]. In [75] reconfigurable core wrappers have been used in SoC test scheduling. Although reconfigurable wrappers lead to efficient test schedules, more gate and routing overheads are imposed. It also increases the control complexity of the wrapper. Wrapper design for cores with multiple clock domains has been proposed in [133]. It can handle cores with different operating frequencies for different scan chains embedded within it.

2.2.2. Test Access Mechanism (TAM)

After designing optimized wrapper for individual cores to minimize core test times, the major challenge of test scheduling is resource allocation to the cores. Optimized resource allocation and core ordering in schedule play key roles to reduce overall TAT. Resource allocation problem has been approached in two different ways in the literature.

- ***Fixed-Width TAM Architecture:*** In this TAM architecture [57], total TAM width is partitioned into several test buses of fixed widths. Each core is then assigned to exactly one of the partitioned test buses.

  However, fixed width TAM architecture suffers from the problem of inefficient use of TAM buses, resulting in longer TAT. For example, in fixed-width TAM architecture, as partitioning of test buses is performed prior to core assignment, a core may be assigned to a TAM of width w and the same testing time can be achieved using lesser test resources. It leads to wastage of test resources, which could have been used efficiently by allocating the extra test resources to the other cores and that might help to reduce TAT.

- ***Flexible-Width TAM Architecture:*** The shortcoming of resource wastage of fixed width TAM architecture can be resolved using flexible width TAM architecture [52, 56, 132, 150, 131, 145, 7, 39], where the total TAM width is not completely partitioned before scheduling. Rather, the resources are allocated to the cores according to the availability and requirement. This flexibility in TAM partitioning explores a huge search space for a reasonably large SoC; hence, there is a chance of getting better schedules, thus, minimizing the TAT.

Figure 2.2 and 2.3 show typical examples of two different TAM architectures.

2.2.3. Test Scheduling

The integrated problem of TAM partitioning and test scheduling is NP-hard [56]. It has been solved using different optimization techniques in the literature.

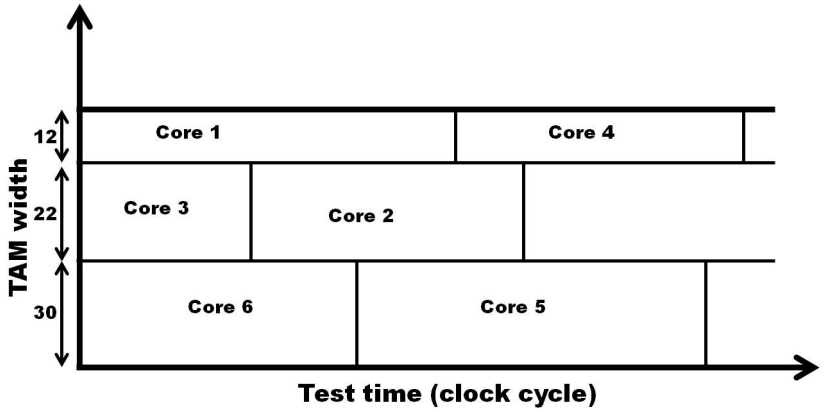


Figure 2.2: A typical example of fixed-width TAM architectures

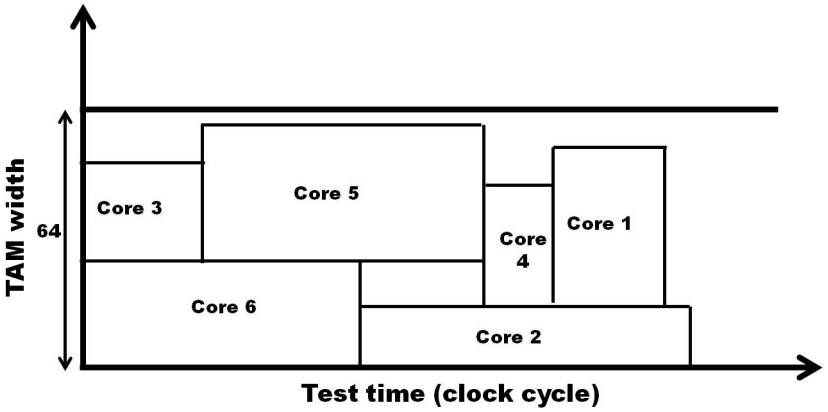


Figure 2.3: A typical example of flexible-width TAM architectures

Authors in [57] have integrated the wrapper design, TAM partitioning and core scheduling problems into a single one and proposed an Integer Linear Programming (ILP) formulation to solve this problem. Similar problem has been solved in [55] using a heuristic method. A graph based approach has been proposed in [71], where it considered the minimum average completion time criteria. Instead of the NP-nature of the problem, the graph-theoretic approach that considers the minimum weight perfect bipartite graph matching, takes polynomial time. The authors in [70] also have formulated the test scheduling as a network transportation problem and solved it using a 2-approximation algorithm utilizing the result of the single source unsplittable flow problem. All these approaches have considered fixed-width TAM architecture.

Due to the limitation of fixed-width TAM architecture, various other approaches used the flexible-width test architecture. However, it cannot always be concluded that flexible-width architecture is better than fixed-width architecture [134]. An architecture-independent theoretical lower bound on test time has been presented in [40]. The work also presents a heuristic named TR-ARCHITECTURE, suitable for both the fixed and flexible-length scan chains. In [52], the scheduling

problem has been formulated as a two-dimensional bin-packing problem. Here, bin height is equal to the total number of available test signal lines. Test requirement of each core is represented as a rectangle having height equal to the number of signal lines allocated and width corresponding to the associated test time. These rectangles are to be packed into the bin so that the width of the bin is minimized. An Evolutionary Algorithm (EA) based approach using sequence pair representation and distributed bin-packing has been presented in [132]. Simulated Annealing (SA) based two-dimensional bin-packing algorithm to solve SoC test scheduling problem has been proposed in [150]. The work also proposes a wrapper design procedure for cores with no scan cells. B*-tree based floorplanning technique has been used in [131] transforming the scheduling problem to floorplanning. Height and width of the floorplan represent the TAM width and test time of the core respectively. The idea is to find a fixed height, minimum width floorplan, while there should be no overlap between the tiles in the floorplan. Simulated annealing has been used to solve this problem. Two stage Genetic Algorithm (GA) based approach to solve SoC test scheduling problem has been presented in [145]. Ant Colony Optimization (ACO) has been used in [7] to solve the SoC scheduling problem. The work reported in [39] uses another Genetic Algorithm (GA) based approach to select the rectangles to pack in a bin for the rectangular 2D-bin packing.

2.3. Prior Work On Power-Aware Testing

Several works in the literature consider power restricted test schedule generation. Different power models of the cores have also been proposed. The concept of considering maximum power consumption value of a core as the fixed power value throughout its testing, has been presented in [28, 53, 98, 18, 76, 38, 29]. This global peak power model, although simple, is quite pessimistic, as power consumption is never same throughout the duration of testing of the core. It may reach its peak at certain time instants only. Also, all the cores being tested in parallel, may not reach their peak power values at the same time instant. The resulting conservative schedule increases the test application time (TAT). In [53], rectangular 3-D bin packing approach has been adopted to solve SoC test scheduling problem under power constraint. Power has been considered as the third dimension with test resource and test time as other two dimensions of the bin. The authors, in [98], have considered multiple test sets for testing of the core. An Integer Linear Programming (ILP) based solution has been presented in [18]. Here the authors have presented a reward model that allows the system integrator to incorporate preferences arising from place-and-route constraints in allocating cores to test buses. A greedy algorithm based technique has been proposed in [76]. A Genetic Al-

gorithm based approach, to minimize TAT has been reported in [38]. Here the authors have used chromosome structure, mutation and crossover operators to select representative rectangles for rectangular 3-D bin packing approach for solving the scheduling problem optimally. A shuffle frog-leaping algorithm (SFLA) that follows evolution of frog population has been used in [77] to solve the scheduling problem under power constraint. All these power-aware test scheduling techniques have used the global peak power model. Exact power model of each core has been presented in [109]. The work approximates the power values for each cycle during testing of the core, to be equal to the total transition in the scan cells, during test pattern shift and capture in that particular clock cycle. The scheduling procedure has been implemented using Best-Fit heuristics with fixed width TAM architecture [109, 110]. Handling huge number of power values and keeping track of the power profile for each clock cycle during scheduling makes this approach extremely time-consuming. Authors in [106, 104] have tried to reduce power overestimation by a less complex method using test vector reordering, test sequence expansion and rotation. The manipulated power profile obtained using these methods has an initial long low power part followed by a short high power part. However, the use of this two local peak power approximation model (2LP-PAM) is limited to the cases when testing is performed exclusively using ATPG-generated test vectors and where the order of the test vectors can be changed. Unfortunately, in most of the cases this pattern reordering may not be feasible as the ATE is preloaded with test patterns.

Recently, introduction of multi-frequency operation of embedded cores has given the test engineers the flexibility of multi-frequency testing. As power consumption is directly proportional to the frequency, a suitable selection of operating frequency for different cores can increase test parallelism without violating power limit. Several other approaches, using virtual TAM for bandwidth matching and test data rate synchronization between ATE and cores operating at different frequencies have been presented in [141, 148, 147]. A test data multiplexer (TDM) / test data demultiplexer (TDdeM) based approach has been used in [141] to fulfill the gap between the frequency of ATE and the cores. A dynamic reconfigurable multi-port ATE based multi-frequency test scheduling strategy has been proposed in [147]. However, all these works consider fixed average or peak power consumption of the core during scheduling.

2.4. Prior Work On Thermal-Aware Testing

Thermal-aware SoC test scheduling problem has drawn the attention of many researchers in recent time. The authors in [102] have used a test session based thermal model for test scheduling. Their RC model based approach tries to max-

imize heat dissipation through lateral neighbourhood of active cores, in a test session. The work assumes negligible heat transfer between two active neighbouring cores. The assumption may not be valid, as we have shown in Chapter 4, the temperature of a core largely depends on the parallelly active neighbours. The concept of thermal ground for idle cores does not hold, as there is a fair amount of leakage power consumption of the idle cores, which actually increase their initial temperature. In [84] the authors have tried to minimize the temperature of the hottest core to achieve a balanced thermal distribution across the chip during testing. The floorplan information has been used during scheduling to avoid concurrent testing of two hot neighbouring cores. They have tried to maintain the temperature of each core below a threshold value, during testing. Hand crafted floorplan of the chip has been used for experimentation. In [103], the authors have presented two algorithms for thermal aware testing. First, an Integer Linear Programming (ILP) based solution has been proposed. It produces optimal solution, but is computationally inefficient. It requires large number of thermal simulations to get a thermal safe test schedule. A faster thermo-resistive model based heuristic has also been proposed, however the model also has the drawbacks similar to [102].

Test set partitioning and interleaving based SoC test scheduling strategies have been presented in [49, 45]. In [49] the authors have divided the total test set into several smaller identical test sub-sequences. The idea behind this approach is that if a core is tested for a long period of time, the temperature of the core will increase. To restrict the temperature of the core under a certain limit, equal length cooling periods have been introduced in between the test sub-sequences. A constraint logic programming (CLP) model and a heuristic approach have been used to solve this problem. In [45] the authors have removed the restrictions on the equality of the length of the test sub-sequences and the cooling periods. This makes the procedure more efficient. However, as it has been shown in Chapter 4 (Section 4.4), temperature of a core increases very fast and reaches its peak value quickly. Thereafter not much variation in temperature is observed with time. So the concept of increase of temperature with time is not always true. Thus, we need to incorporate more numbers of cooling periods between the test sub-sequences, adding extra overhead in scheduling. This may result in longer test application time.

A simplified thermal model based approach has been presented in [143]. The model reduces the number of thermal simulations required to determine thermal violations. The idea behind this work is to test the hotter cores as fast as possible (small test duration) to minimize its effect on the surrounding cores. They have considered Test Access Mechanism (TAM) dependent average power model of the

cores. Scheduling problem has been converted into a 3-D bin packing one. A fast thermal simulator ISAC has been incorporated in [46]. Here, the authors have considered test set partitioning and interleaving based approach for scheduling. Lateral temperature dependency between the cores have been taken. A Mixed Integer Linear Programming (MILP) based thermal solution has been proposed in [14]. The authors have also presented a seed-based clustering approach in this paper. A test-schedule reshaping, test-set partitioning and interleaving based approach has been enumerated in [144]. The authors have tried to minimize the temperature of the hottest core by an iterative process of test reshaping. Test set partitioning, interleaving and bandwidth matching based techniques have been used to cool down the temperature of the cores, which fail to reduce their temperature during the test reshaping process. Fixed width TAM architecture and cycle-accurate power model have been used in this paper. A computationally tractable thermal model based on average power consumption, thermal resistance and test overlapping time has been incorporated for the purpose of calculation simplicity. A thermal-aware test scheduling in an abort-on-first-fail (AOFF) environment, suitable for volume production, has been presented in [47]. In this kind of environment, if a fault is detected in any of the circuit under test (CUT), the test process gets terminated. Test set partitioning and interleaving based approach to minimize the expected test application time (ETAT) has been presented in this paper.

Multi-temperature SoC testing has first been proposed in [48]. As different faults can be detected at different temperature levels, it is important to test an SoC at different temperature levels, to detect all the faults. The authors have tested all the cores in a certain temperature interval. The temperature during testing cannot go beyond the temperature interval. Some passive cooling sequences and heating sequences have been introduced here, to keep the temperature of a core within a certain interval during testing. The concept of global temperature range for the SoC, presented in this paper, has been modified in [138]. Here, the authors have considered different temperature ranges for each individual test. They have also considered multiple tests of a core in different temperature ranges. A list scheduling based algorithm has been used to minimize TAT. On-chip temperature sensor for thermal-aware test scheduling has been proposed in [139]. To overcome the inaccuracies of fast thermal models, the authors have suggested reading the temperature data from on chip sensors and use these data for dynamic test scheduling. However, non-availability of such sensor data may be a problem in this type of testing.

Superposition principle based thermal models have been presented in [137, 136]. In these works, the linearity of RC thermal model has been used to calculate ther-

mal profile of a core during scheduling, using superposition principle. As the CPU execution time is the main bottleneck of thermal simulation during scheduling, they have tried to avoid invoking the thermal simulator in the scheduling process. Instead, they have used the HotSpot [118] thermal simulator to create offline thermal profiles of the cores. These thermal profiles are used for scheduling purpose. This type of thermal model is fast. They have also partitioned the tests with different power and test lengths, to get a better thermal and power constrained test schedule. However, while working with the approach reported in [137, 136], we have noticed that it may result in thermal violations due to inaccurate modelling of heating and cooling rates of cores. One such situation has been shown in Figure 2.4, in which the actual thermal profile during testing violates the temperature threshold used in the test schedule generation procedure. More such cases have been reported in Section 4.6.2.

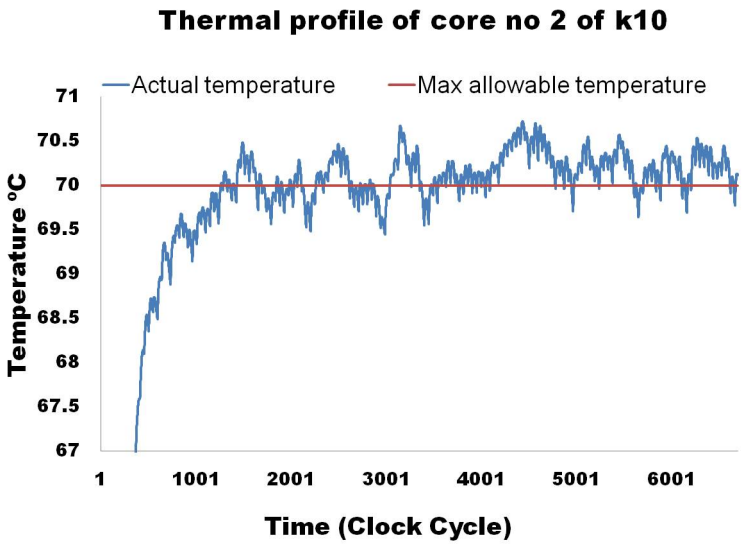


Figure 2.4: Temperature profile and maximum temperature limit of core 2 of $k10$ using the thermal model [137, 136]

2.5. Prior Work On Test Data Compression

Due to the integration of more number of cores into a single SoC and also the increasing complexity of the individual cores, total test data volume has increased enormously to achieve desired fault coverage. For an industrial ASIC, the test data volume can be as high as several gigabits [50]. However, the system designers does not replace the costly ATE, to fulfill the requirement of larger tester memory and high bandwidth of the new SoC design. Rather, he has to opt for some test data compression techniques to overcome the associated problems. Built-in-self-test (BIST) is an alternative solution which avoids the usage of external storage to store test data. However, it suffers from random-resistant faults and

bus contention during test application, leading to inadequate fault coverage [81]. Also, most of the cores are not BIST-ed. External testing is the only option to test these cores.

The test data compression techniques for external testing mainly incorporate some additional on-chip hardware before and after the scan chains. This additional hardware decompresses the test stimulus coming from the tester and also compacts the response after the scan chains and before it goes to the tester. This facilitates storing the test data in a compressed form on the tester.

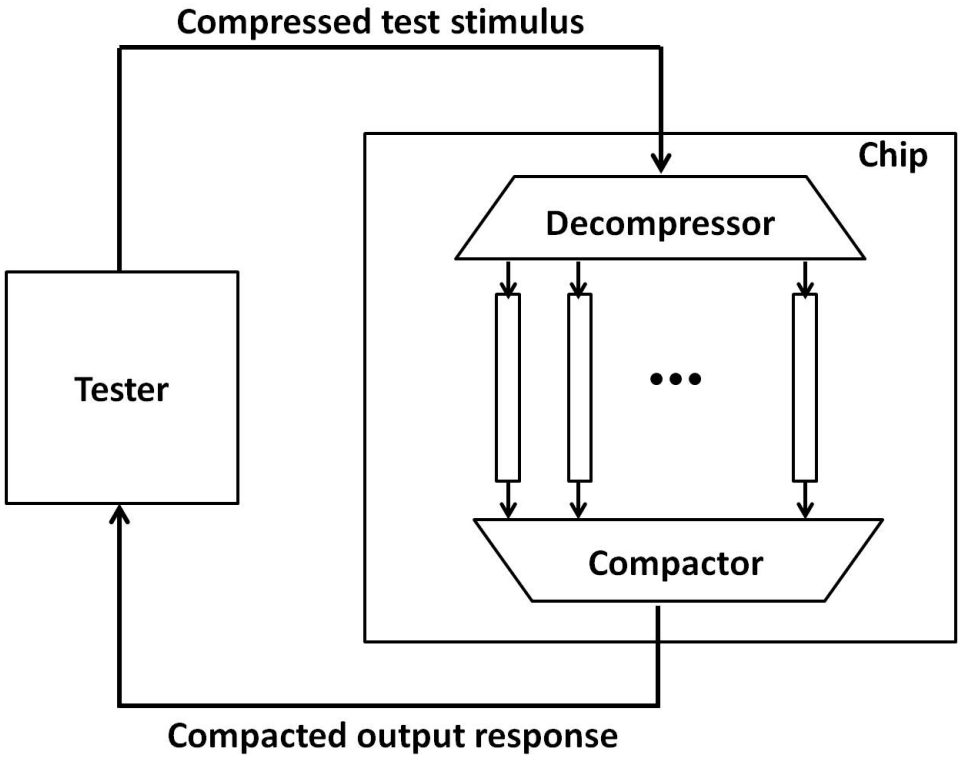


Figure 2.5: Test data compression[124]

The patterns generated by ATPG tools have a huge amount of unspecified bits ("don't-care" or 'X'), which can be assigned either a 0 or a 1. Test data volume can be reduced by intelligent filling of these don't-cares. At the same time, test power and temperature can also be reduced by efficient filling of don't-cares. Although, the works reported in the literature, have proposed test power reduction during testing, none of the approaches has come up with a thermal-aware test data compression technique. Following sections review the test data compression techniques and test power reduction during compression.

2.5.1. Different Test Data Compression Strategies

Test vector compression schemes fall broadly into three categories [124].

- Code-based techniques

- Linear-decompression based techniques
- Broadcast-scan-based techniques

2.5.1.1. Code-Based Schemes

In code-based techniques, test vectors are encoded by partitioning the test cubes into smaller symbols and then replacing each symbol with a code word to form the compressed test data. Finally, a decoder is used to convert each code word to get back original test data from the compressed one.

Depending upon the encoding techniques, code-based techniques can be classified into different categories. They are as follows.

Run-Length Codes: This coding technique tries to exploit the consecutive occurrences of a particular bit ('0' or '1') to get better compression. Run-length codes can be of two types depending upon the length of the codewords. It can be either fixed length code words or variable length code words. The authors in [62] has proposed a scheme based on runs of 0's using fixed length code words. To increase the runs of 0's, the scheme used a cyclical scan architecture to allow the application of difference vectors. Methods like Golomb coding [21], Frequency Directed Run length coding (FDR) [20], Extended FDR [32], Alternate run length coding using FDR [22], MTC coding [105], mutation encoding [100], mixed run length and Huffman coding [123, 97], Variable Prefix Dual-Run-Length code [146], Variable length input Huffman coding (VIHC) [41] used the presence of runs of repeated values (0's or 1's) and then encoded into either fixed length or variable-length code words. The use of variable-length code words allows efficient encoding of longer runs, although it requires a synchronization mechanism between the tester and the chip. A data-independent pattern run-length (PR) compression scheme has been proposed in [107]. The technique applies the run-length coding to equal and complementary consecutive patterns of the pre-computed test data set. In [54], the authors have proposed a compression scheme based on JPEG VLC algorithm targeting multimedia SoCs. A compression algorithm, named *RunBasedReordering* (RBR), based on EFDR, has been proposed in [33]. In this algorithm authors have applied scan chain reordering, scan polarity adjustment and test pattern reordering.

Statistical Codes: This coding technique partitions the original test data into m-bit symbols and each symbol is assigned variable length code words based on its frequency of occurrence. Shorter code words are assigned to the more frequently occurred symbols and less frequently occurred symbols are coded with longer code words. This strategy minimizes the average length of a code word. Huffman code is an optimal statistical code, but the problem is that the decoder size grows exponentially with the number of symbols. Selective Huffman coding

[59] encodes only the most frequently occurring symbols, which has been extended further in [65]. Complementary Huffman coding [86] further improves the test compression. In [60] frequently occurring blocks are encoded using variable length indices. A new compression scheme that codes exactly nine code words (9C) have been proposed in [122]. A multilevel Huffman coding based scheme, where each Huffman codeword corresponds to three different kinds of information, have been proposed in [64].

Dictionary Codes: In dictionary based coding, original data is partitioned into n bit symbols and a dictionary is used to store these symbols. The dictionary entries can be coded using a much smaller code word than the original one, hence reducing total code length. Two types of dictionary based coding schemes are there. In complete dictionary coding [101, 130] all representatives are stored on-chip in a dictionary. This leads to dictionary with large number of entries. With the increase of the dictionary size, the decompressor area overhead also increases. So, to keep dictionary size small, partial dictionary based coding has been proposed in [82, 81]. Test data compression techniques based on LZ77 and LZW algorithm, which use dynamic dictionary, have been proposed in [129] and [68]. In [8] authors analyzed the concept of entropy and proposed arithmetic coding. In [10], the authors have presented a novel matrix-based software data compression scheme. In [11] the authors have combined the advantages of dictionary-based compression and bitmask-based compression to improve test compression. A multiple dictionary approach to reduce the number of bits required to indicate dictionary indexes, along with a slice bit reordering to increase compression of bitmask method, has been proposed in [58]. This work also proposed a power reduction technique in compression. A dictionary based test data compression technique that reuses the parts of the dictionary elements to increase the virtual capacity of the dictionary has been proposed in [115]. It helps to achieve better compression ratio at the cost of extra hardware overhead. The same authors have integrated bit-flipping with reuse of the parts of the dictionary entries for better compression [116]. They also have proposed a technique to enhance test compression of dictionary based coding using the ATE repeat instruction [114].

2.5.1.2. Linear-Decompression Based Schemes

These schemes utilize a linear decompressor to achieve test data compression. In [128], the authors have proposed a compression method that uses statistical transformation for linear decompressor. Decompression for deterministic vector using linear operations has been proposed in [9]. The hybrid test approach [61] generates both random and deterministic patterns for the tested chip while it uses an on-chip linear feed-back shift register (LFSR) to generate the random patterns.

A combinational network, to compress test cubes, has been proposed in [12, 13]. LFSR reseeding based approaches, to achieve test data reduction has been proposed in [73, 125, 69, 78, 140]. An embedded deterministic test (EDT) [99], which uses a ring generator instead of an LFSR, has been proposed for scan vector decompression based on linear sequential decompressor. In [67, 93] authors used a combinational decompressor in which a scan chain is driven by XORing some of the tester channels. A method for improving the encoding effciency of a combinational linear decompressor by dynamically adjusting the number of scan chains loaded in each clock cycle has been proposed in [74]. A test compression technique that retains unused non-pivot free variables to provide better test compression in sequential linear decompressor scheme has been proposed in [95].

2.5.1.3. Broadcast Scan Based Schemes

In broadcast scan based schemes, the same value is broadcasted to multiple scan chains. This is a special degenerate case of linear decompression in which the decompressor consists of only fan-out wires. The main advantage of this technique is that the automatic test pattern generation (ATPG) algorithm can be instructed to produce only encodable test cubes according to the decompressor. Although linear decompression based schemes produce higher compression, more number of faults can be detected using broadcast scan based techniques. This scheme was proposed in [79], in the context of independent circuits i.e. same test set generated by ATPG can be used for different circuits. Illinois Scan architecture has been proposed in [44] to overcome the problem of reduced fault coverage of the single channel driven multiple scan chain architectures. The Illinois scan architecture provides a mechanism to reduce application time and data storage requirement [51]. However, fault simulation and test generation are necessary as post processing to get high fault coverage in this architecture. In [127, 121], the authors have presented efficient test compression schemes based on broadcast scan architecture that supports multiple tester channels.

2.5.2. Power-Aware Test Compression Techniques

The compression/decompression technique may intensively increase the scan-in and scan-out power dissipation of scan chain elements. To reduce power consumption, several works have been done. Test vector ordering [30, 37, 34] scan-cell ordering [36], gated-clock scheme [15], scan latch partitioning [96], don't-care filling [117, 85] are among the proposed methods. A capture power-aware test data compression scheme with efficient don't-care filling has been proposed in [80]. A low scan test power consumption scheme that works with test compression without requiring on-chip clocking has been proposed in [108]. A low power test

compression technique, based on insertion of some inverters in the scan chains has been proposed in [35]. This modifies the original test set to attain low scan power with better compression. In [24, 25] scan power has been reduced by identifying power-friendly seeds without extra hardware overhead for linear test compression schemes. However, none of the methods have proposed temperature minimization in test data compression.

2.6. Conclusion

In this chapter, we have reviewed various state-of-the-art testing strategies for reducing test cost. Though several techniques have been proposed to resolve the testing challenges, none of the proposed methods has come up with a solution that performs reasonably good in all test scenarios. This justifies the search for newer techniques. Moreover, with the increasing complexity of the present day's semiconductor chips, new testing strategies are needed to be devised to tackle the emerging problems. From next chapter onwards different test resource optimization strategies developed in this research work have been presented. Chapter 3 starts with the power-aware test scheduling strategy for the core based SoCs targeting test time optimization.

Chapter 3
Power-Aware Test Scheduling

3.1. Introduction

One of the basic objectives of SoC testing is to reduce the Test Application Time (TAT). To keep TAT within a reasonable limit, one solution is to perform parallel testing of several cores. Overlapped testing of cores may again be prohibited by the system power limit. To keep the power budget of a system low, several design techniques, such as, clock-gating [94, 23, 135, 83, 87], power-gating [83, 87] etc. are generally adopted. Power is also controlled with the knowledge about the run-time activities of different modules within the system. For power reduction, designer has to be concerned only with the modules (cores) active simultaneously. However, the test engineer may need to test some modules simultaneously, which are running in an exclusive fashion during normal system operation. Moreover, for a single module, test mode power is often much higher than functional mode power. Successive input signals in functional mode are generally highly correlated. In test mode, successive test patterns are made as uncorrelated as possible with the expectation that further patterns will excite different regions of the module, possibly identifying newer faults. Thus, in SoC testing, the test engineer is limited by the following.

1. Limited number of channels of Automated Test Equipment (ATE) to transport test patterns to the chip.

2. Limited amount of on-chip test resources (TAM wires). It may be a few signal lines running through the chip to carry the test signals.

3. Different number of input-output pins (including scan input/output) for different cores coming from different IP vendors. Scan chain lengths are also varying.

4. Huge amount of test data (pattern and response) to be stored in the ATE and transported to/from the chip.

The problem gets aggravated with the addition of power limitation. A set of cores can be tested in parallel only if it does not violate the system level power limit at every point of the schedule. Determining such a schedule is difficult as it is necessary to ensure power limit validity at every time instant. Conventionally, a global peak power model has been assumed [28, 53, 98, 18, 76, 38, 77, 29] for the entire test session of a core. On the other extreme, a cycle-accurate power model has been followed in [109, 110]. A global peak power model may generate a too restrictive schedule inhibiting potential parallel testing of some cores, even if the sum of exact instantaneous power values of them is well below the power limit. Cycle-accurate power model does exact power sums; however, the scheduling algorithm needs to handle a large number of power values. It is worth mentioning that excessive power consumption causes overheating, hence may cause permanent damage to the chip. Thus, power validation cannot be ignored. At the same time, using a cycle-accurate power model may become a stumbling block for the scheduling algorithm in search-space exploration.

Most of the approaches [28, 53, 98, 18, 76, 38, 77, 29, 109, 110] assume that the cores and the tester operate at a single frequency. In recent times multi-frequency operation of embedded cores has drawn the interest of many researchers. Test scheduling under multi-frequency environment has emerged to be a potential research problem. It may be noted that power consumption is directly proportional to the frequency. Having the capability of multi-frequency testing, scan-shifting for a power hungry core can be done at lower frequency to minimize its power consumption. Although, it may increase the test time of the core, at the same time, it encourages parallel scheduling of multiple cores due to its reduced power consumption. On the other hand, for less power hungry cores, shifting can be done at faster frequency without violating the power budget, thus minimizing the test time. However, as ATE is capable of sending data at a particular frequency, some architectural modification is needed to synchronize the data rate between the cores and the ATE.

In this chapter, we propose to use an intermediary approach to handle huge amount of power values for the cores in the scheduling process. For a core and a particular wrapper width, the cycle-accurate power values are obtained. Switching activities in multiple scan chains of cores, caused by the test stimulus and responses during *scan-in*, *launch-and-capture* and *scan-out* operations, have been utilized to obtain the power estimation. However, instead of taking cycle-accurate power, the peak power values over a time-window have been taken to approximate the core power over that interval of test time. The process, though introduces some inaccuracy in the power model, works well for most of the test cases. Moreover, the reduction in the number of power values has also enabled us to design a 3D-

bin packing heuristic, guided by Particle Swarm Optimization (PSO) [66] based search procedure, to evolve better test schedules than many of the contemporary SoC testing approaches, while working with ITC'02 benchmarks [90, 89]. CPU time needed to generate the test schedule improves by two order of magnitude compared to a similar scheme using cycle-accurate power model. To facilitate multi-frequency testing, a new simple multiplexer based architecture capable of sending test data at different frequencies from ATE to the cores has also been proposed in this chapter. It may be noted that the proposed power-aware SoC test scheduling approach can also be extended to work in conjunction with variants, such as, multiple voltage islands [63], voltage and frequency scaling [112, 92, 113], etc. The approach can augment those power models as well, as these techniques assume global peak power values only.

3.2. Overview Of The Problem

In SoC environment, different cores from different vendors are provided to the system integrator. These cores have to be tested using a common test methodology. A core-based system is said to be testable if all the cores present in the SoC is equipped with a test method and corresponding test sets. A testable unit can be a core, user defined logic (UDL) or interconnections. In our work, we consider only core testing.

Let the SoC design consist of N cores, and each core C_i $(1 \leq i \leq N)$ has

- PI_i primary input terminals,
- PO_i primary output terminals,
- $Bidir$ bi-directional I/Os,
- S_i scan chains,
- for each scan chain k, the length of the scan chain (number of flip-flops) $l_{i,k}$ and
- p_i number of test patterns to test the core C_i.

Test patterns are stored in an Automated Test Equipment (ATE). Hence TAM wires are required for the transportation of test data from ATE to the testable unit. However, larger cores typically have hundreds of terminals and the total number of TAM channels available depends on the limited number of SoC pins. A test wrapper need to be designed on top of each core to facilitate test width adaptation when the TAM width is not equal to the number of core terminals.

Again, at any point of time, only one testable unit can use a TAM wire. For a certain period of time each test is assigned to some TAM wires. Hence, the

problem that we mainly concentrate on is, how to assign a start time, an end time and the set of TAM wires for each test in such a way that total test time is minimized. The assignment should consider the constraint that at any point of time, a particular TAM wire can be used by only a single testable unit.

For the test scheduling problem under power constraint, a system level power limit is provided to the test engineer. The cumulative power consumption of all cores tested concurrently must not violate this power limit. To generate the test schedule under power constraint, the information regarding power profiles of the individual cores are required. Hence, the test mode power consumption of each core needs to be calculated.

Considering different test environments, power-aware test scheduling problem can be classified into two categories. In single-frequency test environment, all the cores and the ATE operate at the same frequency. In multi-frequency test environment, the cores have the flexibility to operate at different frequency levels, while the ATE can transport the test data only at a single frequency. So, a multi-frequency test infrastructure needs to be developed to facilitate testing of cores at different frequency levels.

In this chapter, we have addressed the power-aware SoC testing problem in the following steps.

- **Wrapper Design:** A wrapper on top of each core has been designed to make it testable and to facilitate modular testing.
- **Power Profile Generation of Each Core:** Power profile of each core has been generated in this step. A new window-based peak power model to estimate the power profiles of individual cores has been proposed in this part.
- **Single-Frequency Power-Aware Testing:** This part proposes power-aware test scheduling strategy of SoC under single-frequency test environment.
- **Multi-Frequency Power-Aware Testing:** A multi-frequency test architecture and multi-frequency power-aware test scheduling strategy has been proposed in this part.

3.3. Wrapper Design

Test wrapper design is important during system integration since it has a direct impact on test time. IEEE Std. 1500 standardizes the wrapper interfaces. Therefore, the internal structure of the wrapper can be adapted to the specific SoC test requirement.

Designing the wrapper mainly involves the construction of wrapper scan chains (SCs) that comprises of a number of wrapper boundary cells and/or core internal scan chains. This wrapper SCs determine the test time of the core and need to be interfaced with the TAM channels. Core test application time depends on the length of the wrapper scan chains. The wrapper chain of maximum length determines the core test application time. Hence, the main objective in wrapper design is to optimize the test time.

During testing of a core, test stimuli shift into the wrapper scan-in chains, launch the test (single capture cycle) and corresponding test responses shift out through the wrapper scan-out chains. Transitions occur in the scan cells at the time of *shift-in*, *launch-and-capture* and *shift-out* operations. While a test response shifts out through the scan-out chains, the next test stimulus shifts into the scan-in chains. Suppose, a certain wrapper configuration of a core C_i has l wrapper chains with wrapper scan-in lengths SI_1, SI_2, $SI_3 \cdots SI_l$ and wrapper scan-out lengths SO_1, SO_2, $SO_3 \cdots SO_l$. The core is tested with p_i test patterns. Total test time (T) is calculated as [88],

$$T = (1 + max(SI_j, SO_j)) \times p_i + min(SI_j, SO_j)(1 \leq j \leq l) \tag{3.1}$$

The length of test stimuli is equal to the sum of primary inputs (PI), Bidirectional lines ($Bidir$) and number of scan cells (SC), while for test responses, it is the sum of primary outputs (PO), $Bidir$ and SC. The test stimuli and test responses split themselves according to the length of the wrapper-scan-in chains and wrapper-scan-out chains respectively in case of multiple wrapper-chain configurations. Sometimes we need to pad some extra bits to the test patterns to match with the length of max(SI_j, SO_j).

To design the wrapper for cores with internal scan chains, we have used the $Design\_Wrapper$ algorithm proposed in [57]. It produces balanced wrappers. From all the wrapper configurations for a core C_i, a smaller set of wrapper configurations can be considered in the test scheduling. It is based on *pareto-optimal* design [57] principle, where for a range of TAM widths, test time remains unchanged. Obviously, only *pareto-optimal* points are of interest since they make use of the lowest possible number of TAM channels to reach a certain test time.

3.4. Window-Based Peak Power Model

Power consumption of a core can be described by the following equation

$$P = 1/2CV_{DD}^2\alpha f \tag{3.2}$$

Where, C is the output capacitance, V_{DD} is supply voltage, α is the number of switching activities in scan chains during test pattern shift and f is the operating frequency. As output capacitance and supply voltage does not change with time and operating frequency also remains unchanged for single frequency operation, the only varying factor is α, which changes with time. This switching activity is the main contributor of power during testing. We consider the total switching activities in wrapper chains in a particular clock cycle as the power consumption of that cycle. However, switching activity depends on the length of the wrapper chains. Different wrapper chain configurations have different switching activities, hence different power profiles. For multiple scan chains, multiple test data are shifted to the scan chains at each clock cycle while only single bit test data is shifted at each cycle for a single scan chain. Obviously, total transition count in the wrapper chains vary with the number of wrapper chains. Hence, a core has several power profiles corresponding to each wrapper configuration of it. Figure 3.1 shows the variation in the power profile with variation in the number of wrapper chains of module number 1 of ITC'02 benchmark circuit, $d695$, when the module is tested with a single test pattern. Transition count increases gradually at the time of $shifting\text{-}in$ the test stimulus. Abrupt change in the power value indicates the *launch-and-capture* operation. This happens since at this time, some scan flip-flops change their content, getting values from the circuit under test. $Shift\text{-}out$ operation is clearly indicated by the gradual decrease in transition count.

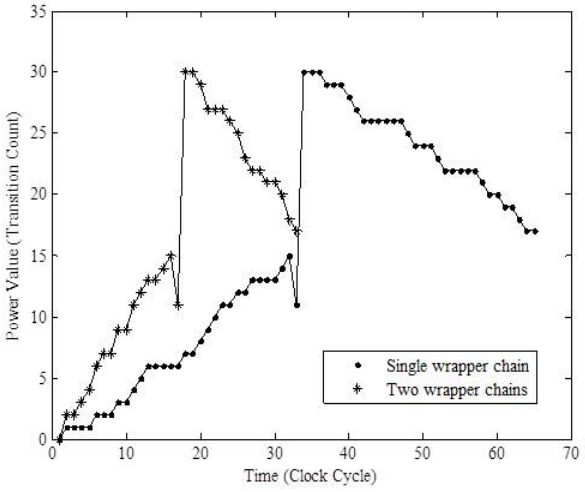


Figure 3.1: Power profiles of module no 1 of $d695$ for different wrapper configurations

Cycle-accurate power values can be computed by calculating total transition count in scan chains at each cycle for all sets of wrapper configurations of each core. However, for bigger circuits with longer test time values, it is very difficult to handle a large number of associated power values in the test schedule generation process. From the cycle accurate power profiles, it can also be observed that for

most of the time instants, there is not significant variation in the power profile in the neighborhood of it. This has motivated us to make the power-model coarse-grained via windowing.

In window-based peak power model, we partition the total test time of a core into some smaller sized time windows and consider a single peak power in that interval to represent the power value for that interval. The window-based power model gives us a faster test scheduling strategy than cycle-accurate power model since now we need to work with less number of power values. A single global peak-power model introduces high amount of overestimation compared to the actual instantaneous power values. Window-based power model has the capability to reduce the amount of this overestimation. Very small window interval leads towards the cycle-accurate power model while large window-size incorporates more false power values in the schedule. Figure 3.2 illustrates the concept of window-based peak power model. It shows the cycle-accurate power values, global peak power of 15 units and the window-based model with window-size 10. For example, for the time interval 0 to 10, the peak value is 8 units and thus, it has been used as the power value for the entire window. From the figure, it can be observed that the amount of overestimation of power is much less in the window-based model compared to the global peak power model. For the example in Figure 3.2, the number of power values to be remembered is only 5 in the window-based model compared to 50 for the cycle-accurate case. For the full 50 cycle operation, overestimation in global peak power model is 353 units (computed by summing up the differences between the global peak of 15 units and the instantaneous power values), while in window-based model, it reduces to 143 units.

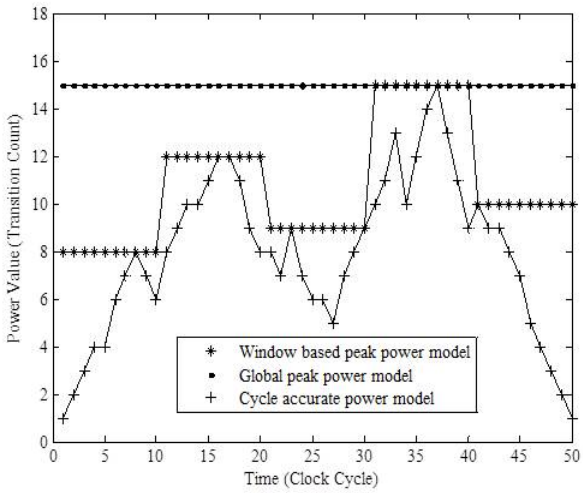


Figure 3.2: Comparison of window-based peak power model with other power models

The accuracy of window-based peak power model depends on window-size to a large extent. In our formulation, the length of window interval has been

kept flexible and can be tuned according to the requirements and computational resources available. Figure 3.3 shows the variation of power profiles of core number 6 of SoC $d695$ (wrapper length 18) for different window-sizes. It may be noted from the figure that, window-size of 5000 clock cycles considers large amount of power overestimation. However, if we gradually reduce the window-size from 5000 clock cycles to towards 500 clock cycles, the power overestimation can also be minimized. Further reduction in the window-size will lead towards cycle-accurate power model. Obviously schedule generation time depends on the length of the window interval. Figure 3.4 shows the variation of CPU time required to generate a valid schedule for different window-sizes for ITC'02 benchmark $p93791$ using the test scheduling algorithm *Schedule_Rectangle* mentioned in Section 4.2.2. It is clear from the figure that CPU time increases exponentially with a reduction of window-size from global peak (GP) to cycle-accurate (CA). However, it is worth mentioning that CPU times reported in Figure 3.4 for different window-sizes are the times required only to generate a single valid schedule, which do not ensure near optimal TAT. Generation of near-optimal solution for NP-Hard problems like test scheduling requires some evolutionary technique, where schedule generation time not only depends on scheduling algorithm, but also on the total number of iterations required to evolve a solution. So, final schedule generation time is much higher than the time required for a single solution and may be too costly for cycle-accurate power model.

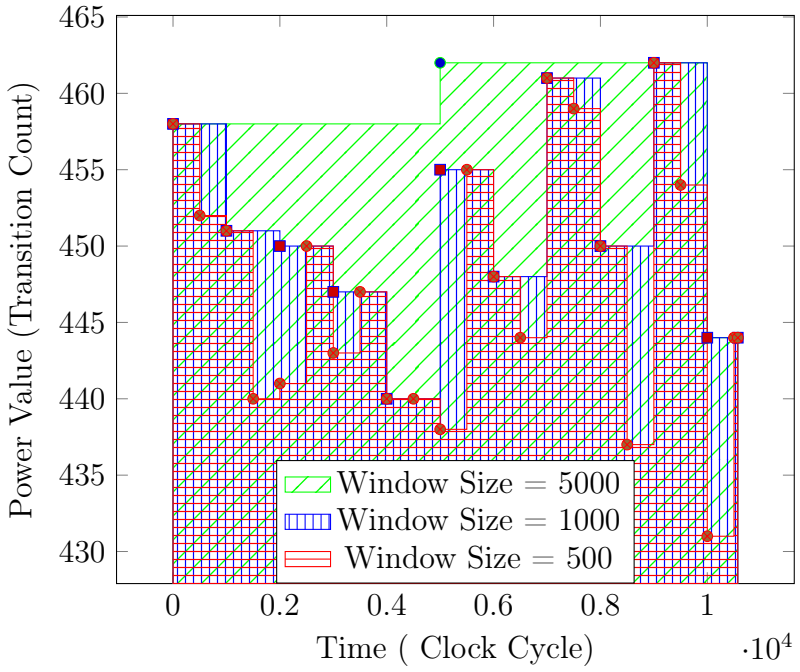


Figure 3.3: Variation of power-profiles of core number 6 of SoC $d695$ (wrapper length 18) for different window-sizes

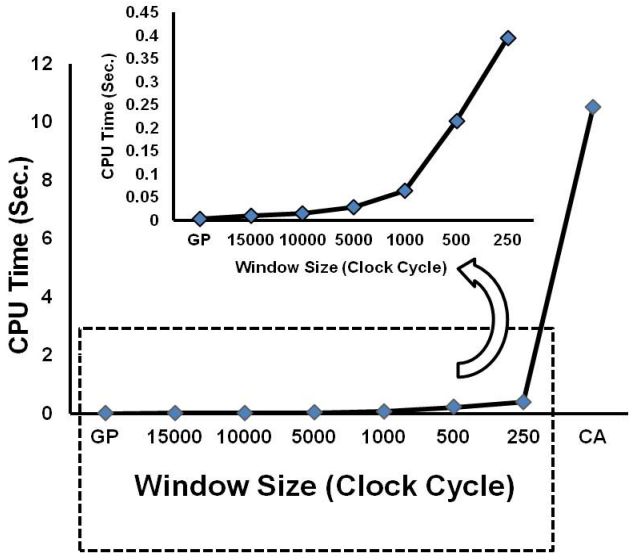


Figure 3.4: Variation of schedule generation time with window-size for $p93791$ for a single iteration

The benefit of using window-based peak power model over global peak power model can be explained using Figures 3.5 and 3.6. Let us assume that we have three cores core 1, core 2 and core 3 with test times 4700, 1550 and 2680 clock cycles respectively. Global peak power values of the cores are assumed to be 13, 8 and 11 units respectively and the maximum system-level power limit is 22 units. We tried to schedule the cores using both global peak power model and window-based peak power model. Figure 3.5 shows the scheduling of the cores using global peak power model while Figure 3.6 depicts the test schedule using window-based peak power model.

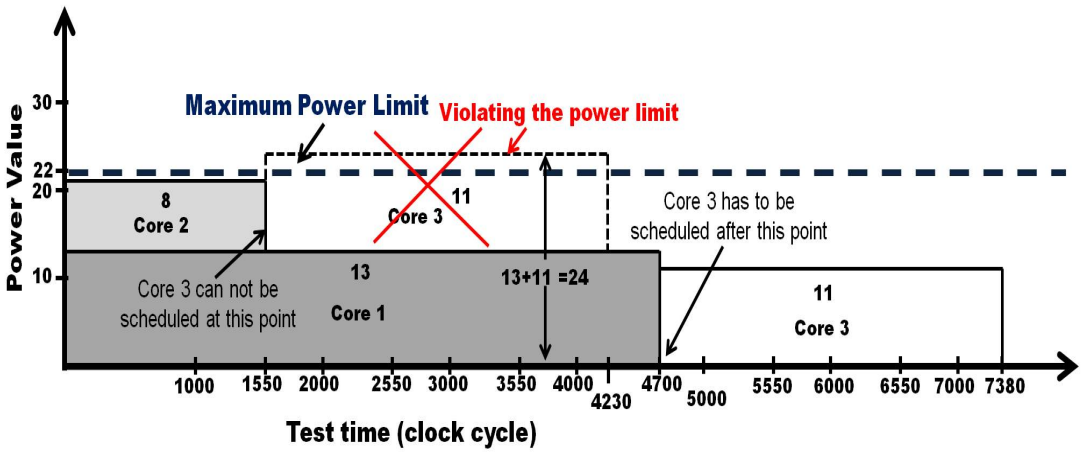


Figure 3.5: Test schedule using global peak power model

We have considered window-size of 1000 clock cycles for the window-based peak power model. Window-based peak power value for core 1 in the window interval 1 to 1000 is assumed to be 10 units while these values are 7, 13, 10 and 4 units for successive window-intervals. It may be noted that core 1 attains its peak power

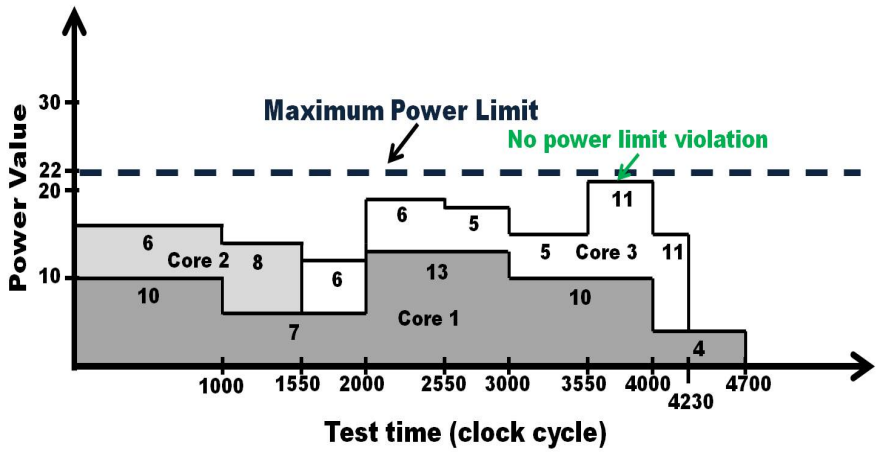


Figure 3.6: Test schedule using window-based peak power model

in the time interval 2000 to 3000 clock cycles while its power values are less for other window-intervals. Similarly, the window-based peak power values for core 2 and 3 have been pictorially described in Figure 3.6. Now, core 1 and core 3 cannot be scheduled in parallel using global peak power model (Figure 3.5), as the sum of their peak power values (i.e. 13+11 = 24) violates the system level power limit value of 22 units. To satisfy the power constraint, we have to serialize the test schedule by scheduling core 3 only after core 1 finishes, making the TAT unnecessarily longer. However, it may be noted from Figure 3.6 that core 1 and core 3 do not attain their respective peak power values at the same time. The window-based peak power values of core 3 in the interval 2000 to 3000 clock cycles (where core 1 attains its peak power value of 13 units) are 6 and 5 units. Similarly, core 3 attains its peak power value of 11 units in the time interval 3550 to 4230 clock cycles, where the window-based peak power values of core 1 are 10 and 4 units. It is clear from Figure 3.6 that, there is no power limit violation even if we test cores 1 and 3 in parallel using window-based peak power model. This reduces the TAT by reducing unnecessary power over-estimations and increasing test concurrency over global peak power model.

3.5. Multi-frequency Test Infrastructure

In modern day SoCs, most of the cores have the facilities of being tested at multiple frequency levels. This allows us to shift the test data from ATE to the scan chains at different frequencies. Different shift frequencies of a core produce different power profiles of the core. It may be noted from Eqn. 3.2 that power is directly proportional to the shifting frequency. Thus, an increase in the shift frequency increases the power consumption by the same proportion. However, the test time of the core reduces by same ratio. Similarly for low frequency operation although the power consumption reduces, test time of the core increases. These

variations in the test time and power profile of cores with shift frequency introduce lots of flexibilities in the schedule. A test controller needs to select a suitable operating frequency of a core among the available frequency ranges to get a better schedule.

However, it should be noted that ATE can be operated only at a single frequency, while the cores can be tested at different frequency levels. This frequency mismatch between cores and ATE can be resolved by bandwidth matching between ATE and cores. The cores, operating at higher frequencies should be provided with more resources than normal frequency operation condition, while the low frequency cores require less resource to fulfil the bandwidth requirement criteria.

3.5.1. Multi-frequency architecture

To control the test data rate between ATE and the cores operating at different frequency levels, TAM architecture needs to be modified. We propose a multiplexer based architecture that coordinates between ATE and cores, to select proper resources. Figure 3.7 shows the proposed MUX based multi-frequency architecture. For the sake of simplicity we have assumed that a core can operate at five frequency levels ($f/4$, $f/2$, f, $2f$ and $4f$), while the ATE can send data only at a single frequency f. Let, n be the number of TAM wires required to test the core if both the ATE and the core operate at frequency f. Suppose a core operates at frequency $4f$, total bandwidth (BW) required to test the core is $BW = n \times 4f$. So, the ATE, which sends data at a rate of f, has to allocate $4n$ numbers of TAM lines to test the core to fulfil the bandwidth criteria. A test controller decides the operating frequency of individual cores and generates a three bit signal, which chooses the appropriate number of TAM resources required to test the core. A 4:1 and a 2:1 MUX are used for this purpose. Test controller generated signals and corresponding variation in required TAM, test time and power values have been noted in Table 3.1.

Table 3.1: Variation Of Different Test Parameters With Operating Frequency Of Circuit Under Test

Controller Signal			Core Frequency	ATE Frequency	TAM Required	Test Time	Power
0	×	×	f	f	n	t	p
1	0	0	$f/4$	f	$n/4$	$4t$	$p/4$
1	0	1	$f/2$	f	$n/2$	$2t$	$p/2$
1	1	0	$2f$	f	$2n$	$t/2$	$2p$
1	1	1	$4f$	f	$4n$	$t/4$	$4p$

Our simple MUX based approach can dynamically select the operating frequency of the core during scheduling and allocate the resources. However, it

may be noted that the test controller is restricted by the maximum bandwidth of $BW_{max} = W_{max} \times f_{ATE}$ (f_{ATE} is the frequency of ATE, i.e. f). Total allocated resources to all the cores tested in parallel is also limited to W_{max}.

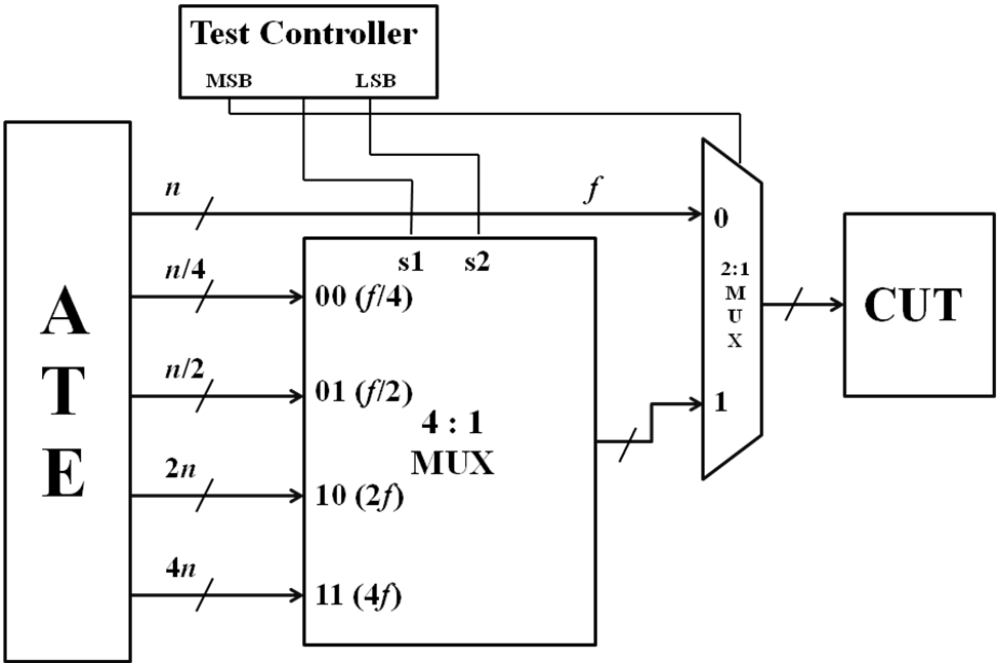


Figure 3.7: Multi-frequency test architecture

3.6. Test Scheduling

The power-aware SoC test scheduling problem under single and multi-frequency test environment are described in the following.

3.6.1. Problem Formulation

- **Single-Frequency Test Environment:**

  Suppose a SoC with N cores $C_1, C_2 \cdots C_N$ is to be tested with a maximum of W_{max} TAM resources and a maximum power limit P_{max}. The test scheduling problem is to allocate TAM resources and test times to the cores so that, the total test application time (TAT) is minimized and the power consumed by the SoC at each instant of time during test is less than P_{max}.

- **Multi-Frequency Test Environment:**

  Suppose a SoC with N cores $C_1, C_2, C_3, \cdots C_N$ is to be tested with a maximum of W_{max} TAM resources and a maximum power limit P_{max}. Each core can be tested at any of the five frequency levels ($f/4, f/2, f, 2f$ and $4f$). The test scheduling problem is to choose appropriate operating frequencies of individual cores and allocate TAM resources and test times to them so that, the total test application time (TAT) is minimized. Also, the power

consumed by the SoC at each instant of time during test has to be less than P_{max}.

Due to its flexibility of TAM width attachment to individual cores, rectangular 2-D bin packing has evolved to be a popular method to solve the test scheduling problem for embedded cores [56]. Each core C_i $(1 \leq i \leq N)$ is represented by a set of wrapper configurations R_i. The test resource requirement of core C_i with j^{th} wrapper configuration can be represented by a rectangle whose height and width represent allocated TAM width (w_{ij}) and the corresponding test time $(T(w_{ij}))$ respectively. For multi-frequency approach, these representative rectangles for each core are formed considering that the core and the ATE operate at the same frequency (i.e. f_{ATE}). However, actual test time and allocated TAM width is decided based on the operating frequency of the core selected by a test controller.

To get a schedule for the full SoC, the rectangles are to be packed into a bin of fixed height (W_{max}) so that TAT (width of the bin) is minimized. Power constrained scheduling takes this problem to a 3-D bin packing problem, where power represents the third dimension. We have used the window-based peak power profile for each wrapper configuration of each core. Bin packing is NP-Hard [56]. In the following, we present a Particle Swarm optimization based approach to solve the scheduling problem.

3.6.2. Test Schedule Generation

The process consists of the following components.

- **Single-Frequency Test Environment:**
  1. Generation of test rectangles $(TR_i, 1 \leq i \leq N)$ for individual cores.
  2. Selecting one test rectangle for each core.
  3. Scheduling the selected test rectangles $(TR_i, 1 \leq i \leq N)$.
- **Multi-Frequency Test Environment:**
  1. Generation of test rectangle $(TR_i, 1 \leq i \leq N)$ for each core.
  2. Selection of one test rectangle for each core.
  3. Selection of operating frequency for each core and calculation of frequency factor $FF_{C_i} = \frac{f_{C_i}}{f_{ATE}} (1 \leq i \leq N)$
  4. Modification of each test rectangle according to the corresponding frequency factor.

5. Scheduling the modified test rectangles ($MTR_i, 1 \leq i \leq N$).

The first step is a stand-alone one, while the remaining stages need to be solved in an integrated fashion. We have used the *Design_Wrapper* algorithm [57] to generate different wrapper configurations for each core. For a core, the wrapper configurations corresponding to only the pareto-optimal TAM widths [57] are noted. Corresponding window-based power profiles are determined. Selection of one test rectangle per core has been performed using PSO. Each particle gives a set of rectangles, one for each core. Fitness of the particle has been evaluated by performing a scheduling of these rectangles.

3.6.3. Particle Swarm Optimization

Particle Swarm Optimization (PSO) [66] is a population based stochastic technique developed by Eberhart and Kennedy in 1995. PSO is initialized with a group of particles with random positions and it searches for optima by updating their positions through generations. In PSO system, multiple candidate solutions coexist and collaborate simultaneously. Each solution, called a particle, flies in the problem space according to its own experience as well as the experience of neighboring particles. In PSO, each particle is a single solution in the search space, having a fitness value. The quality of a particle is evaluated by its fitness. Particles evolve over generations guided by self- and group-intelligence, and also via their inertia. Any PSO formulation involves choosing a proper representation of the particles, their fitness calculation, and defining an evolution policy. Inspired by its success in solving problems in continuous domain, several researchers have attempted to apply it in discrete domain as well [126]. Next, we elaborate each of these in the context of power-aware test scheduling of cores in a SoC.

- **Particle structure for single-frequency test environment:**

Each core has a set of test rectangles. Considering the power values, we can say that each core has a set of test cuboids associated with it - the dimensions being wrapper width, test time, and power. However, since we are not considering pattern reordering, the two cuboids of a core cannot differ only in their power values - we will conveniently call them rectangles only. Let the number of cores in the SoC be N, and the maximum number of rectangles for any core be M. Let $B = \lceil \log_2^M \rceil$. A particle consists of $N \times B$ number of bits. First B bits identify the test rectangle selected for the first core, second B bits for the second core, and so on. Figure 3.8 shows a sample particle for single-frequency test environment with $N = 4$ and $B = 4$. In this case, test rectangles 9, 2, 8 and 13 are selected for cores 1, 2, 3 and 4 respectively. For the initial generation, particles are generated

randomly; however care has been taken to ensure that the indices generated for a core are always within the total number of rectangles of it.

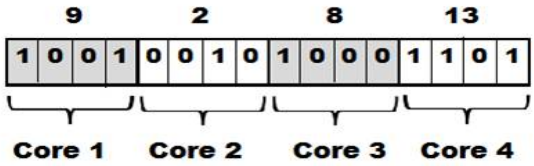


Figure 3.8: Sample particle structure of 4 cores with W_{max} =16 (B = 4) (for single-frequency test environment)

- **Particle structure for multi-frequency test environment:**

In multi-frequency test environment, a particle not only selects the rectangle corresponding to each core but also the core operating frequency. So, the particle structure is modified in multi-frequency testing, to incorporate the frequency component. Let the number of cores in the SoC be N, the maximum number of rectangles for any core be M and the number of frequency values be F. Let $B = \lceil \log_2 M \rceil$ and $K = \lceil \log_2 F \rceil$. A particle consists of $B \times N + K \times N$ number of bits. First $B \times N$ bits select the rectangle indices for the cores, while next $K \times N$ bits select their corresponding frequencies. The decimal equivalent of first B bits identifies the test rectangle selected for the first core, second B bits for the second core, and so on. The decimal equivalent of each K-bit value of remaining $K \times N$ bits selects a frequency. For example, for $K = 3$ and $F = 5$, values are in the range 0 to 4 ($0 \leftarrow f/4, 1 \leftarrow f/2, 2 \leftarrow f, 3 \leftarrow 2f$ and $4 \leftarrow 4f$). Figure 3.9 shows a sample particle for multi-frequency test environment with $N = 4$ and $B = 4$, $F = 5$. In this case, test rectangles 9, 2, 8 and 13 are selected for cores 1, 2, 3 and 4 and their corresponding frequencies are $f/2, 2f, 4f$ and f respectively. Here also particles are generated randomly for the initial generation. Extra care has been taken to ensure that the bandwidth ($w_{ij} \times f_{C_i}$) allocated to any core C_i is within the limit of maximum allowable bandwidth value of $W_{max} \times f_{ATE}$.

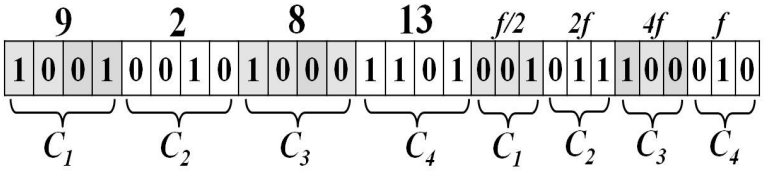


Figure 3.9: Sample particle structure of 4 cores with W_{max} =16 (B = 4, F = 5) (for multi-frequency test environment)

- **Local Best ($pbest$) and Global Best ($gbest$):**

In a PSO formulation, evolution of a particle is guided by three factors - its own intelligence, global (swarm) intelligence, and the inertia factor. A particle always remembers its history about its best structure over generations. This is called the local best (*pbest*) of the particle. In a particular generation, the particle with the best fitness value is the global best (*gbest*) of the generation. For the initial generation, *pbest* of each particle is initialized to itself while the *gbest* of the generation is the best one of the first generation. In the successive generations, new particles are created using the *replace* operator noted next.

- **Evolution of a particle**

In this step, the *replace* operator attempts to align a particle with its *pbest* and the *gbest* particles, with some probability. For the sake of this alignment, the *replace* operator is applied at each bit position of a particle. For the bit position i of a particle, the bit is replaced by the corresponding bit of *pbest* particle with probability α. After the operator has been applied for *pbest*, the same is done with reference to *gbest* with probability of replacement, β. After both the replacement operators have been applied to all bit positions for a core, a consistency check is performed. If the new rectangle number for the core becomes larger than the total number of rectangles available for the core, the rectangle number is reverted back to its value in the original particle. Similar replacement operations and consistency checks are also done on the frequency part of the particle in multi-frequency test environment. In our experimentation, we have kept both α and β values at 0.1. Figures 3.10 and 3.11 show two examples of alignment of a particle towards its local best in single-frequency and multi-frequency test environments respectively.

- **Fitness of a particle**

Fitness of a particle is equal to the total test time (TAT) of the SoC after scheduling the test rectangles using the heuristic mentioned in the following.

The algorithm takes as input the rectangle set TR_i, $(1 \leq i \leq N)$ corresponding to the particle, the maximum TAM width W_{max}, and the maximum power limit P_{max}. It performs scheduling of the rectangles, honouring the constraints that at no instant of time, the total TAM width requirement exceeds W_{max}, and the instantaneous power value does not exceed P_{max}. The resulting total test time (TAT) is the fitness of the particle.

However, these rectangles cannot be directly used for scheduling due to bandwidth matching problem under multi-frequency environment. For multi-frequency

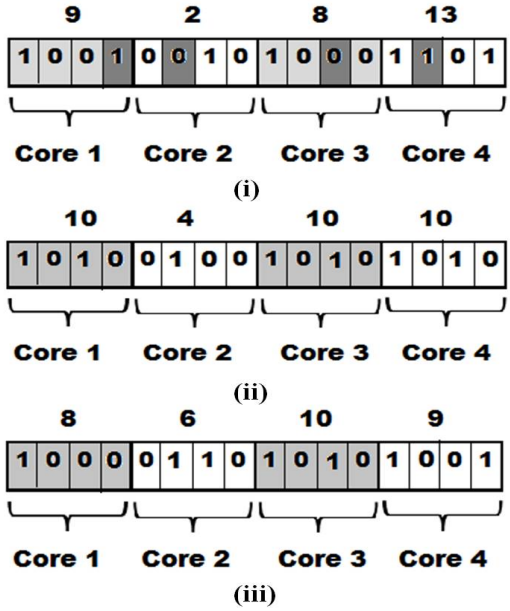


Figure 3.10: Evolution of a particle
(i) current particle, (ii) local best and (iii) evolved particle (for single-frequency test environment)

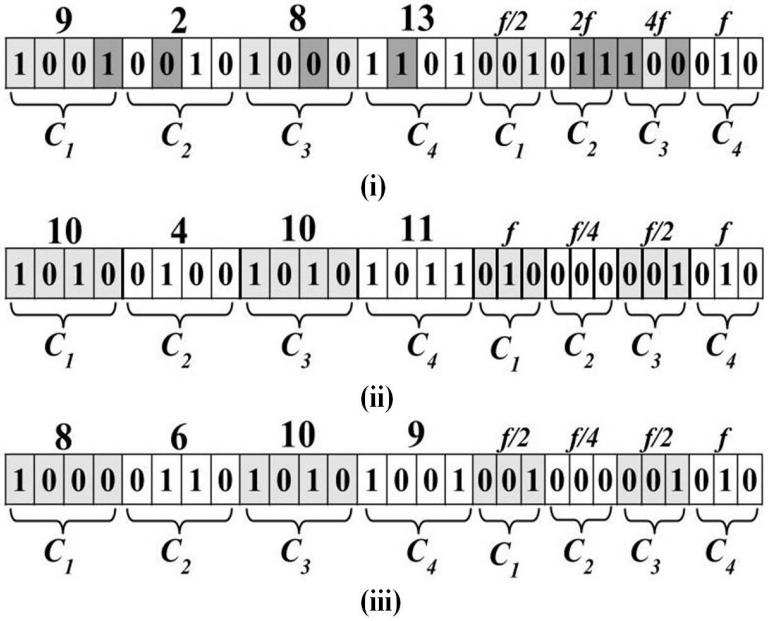


Figure 3.11: Evolution of a particle
(i) current particle, (ii) local best and (iii) evolved particle (for multi-frequency test environment)

environment, the rectangle set TR_i, $(1 \leq i \leq N)$ is modified to a new set of rectangles MTR_i, $(1 \leq i \leq N)$ according to the frequency f_{C_i} of each core C_i. The height w_{ij} of TR_i is multiplied by FF_{C_i} to get the height of MTR_i, while the width of MTR_i can be calculated by dividing the width of TR_i (*i.e.* $T(w_{ij})$) by FF_{C_i}. The power profiles of the cores corresponding to the modified rectangle set are modified accordingly, keeping the total energy during testing same for both

TR_i and MTR_i.

At any point of time, the algorithm maintains the following data structures to arrive at a decision about scheduling the next core.

Break_ Point_ List (BP):

A set of time instants at which the power requirement of the schedule has changed from its value in the previous instant. The next core can be scheduled at any of the breakpoints, $bp_k \in BP$.

Available_ TAM_ Width_ Info (ATW):

A set with cardinality same as BP. The value atw_k is equal to the total free TAM width available at break point instant bp_k.

Power_ Tracker (PT):

This is also a set with cardinality same as BP and holds the total power consumed by the already scheduled cores at the corresponding break-point instant.

As an unscheduled cores get scheduled, the list BP, ATW and PT also get updated. The bin packing procedure also needs to prioritize the next unscheduled rectangle to be selected for packing (scheduling). For this purpose, the rectangles are sorted on their area values (TAM width (w) $\times$ test time (T)) in a descending order. The break-point list BP is scanned from the minimum to the maximum value. To make the schedule compact, we try to utilize any available TAM resource and power budget at every break-point. Hence, for the break-point bp_k, the algorithm scans the unscheduled rectangle list to check for the largest rectangle that can be scheduled at bp_k. If none are feasible, the algorithm advances to the next break-point. Power requirements are also checked to ensure satisfaction of power limit at every break-point till the end of the schedule for the current core. When rectangles corresponding to all cores have been scheduled, the maximum end time of testing of all cores gives the total test application time for the SoC. The proposed *Schedule_Rectangles* algorithm to produce the schedule is presented in Algorithm 3.1.

- **Termination conditions**

If for the last 200 generations, best result, that is the particle which has the maximum fitness, does not change, we stop running the algorithm. We have also used a maximum iteration condition, after which we stop, even if solution is still improving.

Algorithm 3.2 *PSO_Pseudocode* notes the proposed PSO.

Algorithm 3.1: ***Schedule_ Rectangles***

Input : List of rectangles to be scheduled; W_{max}, the maximum TAM width; P_{max}, the power limit
Var : BP**:** A list of breakpoints; ATW**:** List of available TAM widths at each break point $bp \in BP$; PT**:** List of total power values at each break point $bp \in BP$;

```
1  begin
2      BP ⟵ 0;
3      ATW ⟵ W_max;
4      PT ⟵ 0;
5      Sort list of rectangles on decreasing area;
6      Mark all rectangles as unscheduled;
7      while there exists unscheduled rectangles do
8          while all entries of BP not checked do
9              Let bp_k be the next entry of BP;
10             atw_k ⟵ ATW[bp_k];
11             pt_k ⟵ PT[bp_k];
12             Check if any unscheduled rectangle picked up in sorted order can
               be scheduled at bp_k with available TAM resource and power
               budget;
13             if yes then
14                 Update BP, ATW, PT and Rectangle List;
15                 Mark corresponding rectangle scheduled;
16             else
17                 Continue with next bp_k ∈ BP;
18     Return the maximum test end time among all rectangles.;
```

Algorithm 3.2: PSO_Pseudocode

```
1  begin
2      Initialize all particles;
3      while Max iterations not reached and gbest has changed in last 200
       generations do
4          for all Particles do
5              Calculate fitness value using Algorithm 3.1;
6              if fitness is better than current pbest then
7                  Update pbest;
8              if fitness is better than current gbest then
9                  Update gbest;
10         for all Particles do
11             Apply the replace operator to find the new position;
```

3.7. Experimental Results

In this section we present results of experimentation with ITC'02 [89, 90] benchmark SoCs. SoC test scheduling approaches can broadly be considered in three

stages of complexities. The initial basic approach does not consider the power limits in the scheduling process. An improved version may consider test power values of the cores and test scheduling under a system level power limit. A further improvement is the multi-frequency test environment that introduces variation in the selection of core operating frequency and also provides the test engineers more flexibility of optimization.

Most of the power-constrained SoC test scheduling approaches consider global peak power model, while few others consider cycle-accurate power model. However, in our formulation, our main focus is to divide the total test time into some window-intervals and consider peak power values for all those intervals (i.e. window-based peak power model). As our approach does not consider the same power profile suggested in the literature, it is not justifiable to have a direct comparison of our results with those presented in the literature.

To have a better comparative analysis for all different categories of SoC test scheduling, we have divided our results into four subsections. The first one compares our PSO results with the approaches that do not take power into consideration. The comparison establishes the suitability of PSO to solve the problem. The second subsection considers the same power model (i.e. global peak power model) suggested in the literature and compares the quality of our PSO guided test scheduling algorithm with other solution strategies reported in the literature. The third subsection compares the quality of our proposed window-based peak power model with global peak as well as TAM dependent peak power models. It also performs comparison with cycle-accurate power model results, in terms of solution quality and CPU time requirements. Finally, in the fourth subsection, we have shown the quality of multi-frequency approach over single-frequency approach in reducing test application time for both global-peak and our proposed window-based peak power models.

3.7.1. Test Schedule Without Power Constraints

Several techniques have been developed to solve the basic test scheduling problem of SoC. Among them evolutionary approaches [150, 131, 145, 7, 39, 132] do better than other iterative heuristics. Since PSO is an evolutionary algorithm, we have compared our results with several other approaches [150, 131, 145, 7, 39, 132] noted in Tables 3.2 and 3.3. All these approaches use a flexible TAM architecture. The work LBT [40] provides a theoretical lower bound under certain assumptions about the test application process. In the table, we have included this result also to act as a yardstick for other methods. However, it should be noted that LBT predicts the lower bound under certain assumptions only. Hence, results better than LBT could be obtained by other methods, including ours. From Tables 3.2 and 3.3,

it can be inferred that no single method could produce consistently good results. For example B*-SA has done consistently good for SoC $d695$, our PSO does better than others in most of the cases of $p22810$ and all the cases of $p93791$. SoC $p34392$ contains a bottleneck core with exceptionally long test rectangle. This makes all optimization strategies equally bad.

Table 3.2: Comparisons Of PSO With Other Scheduling Procedure For Different ITC'02 SoCs For Different W_{max} Without Power Constraints

	GA[39]	**B*-SA[131]**	**PSO**
$g1023$			
W_{max}	TAT (Clock Cycle)		
16	30755	**29765**	31059
24	20498	**20032**	21135
32	15843	**14913**	15896
40	**14794**	**14794**	**14794**
48	**14794**	**14794**	**14794**
56	**14794**	**14794**	**14794**
64	**14794**	**14794**	**14794**
$f2126$			
W_{max}	TAT (Clock Cycle)		
16	357088	**350030**	357088
24	**335334**	**335334**	**335334**
32	**335334**	**335334**	**335334**
40	**335334**	**335334**	**335334**
48	**335334**	**335334**	**335334**
56	**335334**	**335334**	**335334**
64	**335334**	**335334**	**335334**
$t512505$			
W_{max}	TAT (Clock Cycle)		
16	10530995	**10504020**	10530995
24	**10453470**	**10453470**	**10453470**
32	5268868	**5228420**	5268868
40	**5228420**	**5228420**	**5228420**
48	**5228420**	**5228420**	**5228420**
56	**5228420**	**5228420**	**5228420**
64	**5228420**	**5228420**	**5228420**

3.7.2. Test Scheduling With Global Peak Power Model

Since many of the power-aware SoC test scheduling approaches consider a single power value for the entire duration of testing a core, first we show the results of our approach using this global peak power model. It may be noted that, only SoC $h953$ has been provided with a list of global peak power values of the cores in ITC'02 benchmark suite. No information regarding power values of the rest of the SoCs are available in the benchmark suite. However, global peak power values of the cores of SoC $d695$, $p22810$ and $p93791$ have been reported in [98]. We have considered these values for the sake of experimentation. We could not carry out the comparison on other benchmarks as no reported data regarding test scheduling under global peak power is available in the literature. Tables 3.4, 3.5, 3.6 and 3.7 note the power-aware test scheduling results for the benchmarks $d695$,

Table 3.3: Comparisons Of PSO With Other Scheduling Procedure For Different ITC'02 SoCs For Different W_{max} Without Power Constraints (Contd.)

	LBT[40]	**GA[39]**	**ACO[7]**	**2stageGA[145]**	**B*-SA[131]**	**EA(C)[132]**	**EA(nC)[132]**	**SA[150]**	**PSO**
$d695$									
W_{max}	**TAT(Clock Cycle)**								
16	40951	41604	41737	40691	**39489**	41553	41809	41604	41749
24	27305	27767	28080	28060	**26203**	27982	27989	28064	27979
32	20482	20957	21098	20977	**19773**	21014	27989	21161	21136
40	16388	16913	17075	16894	**16149**	16908	17015	16993	16962
48	13659	14273	14310	14129	**13649**	14240	14236	14183	14310
56	11709	12084	12110	11453	**11285**	11988	12134	12085	12134
64	10247	10723	10783	10573	**9885**	10571	10788	-	10723
$p22810$									
W_{max}	**TAT(Clock Cycle)**								
16	419466	428852	**424889**	438619	438619	438783	438619	438619	425113
24	279644	286352	289190	288565	287999	292824	289237	289287	**284851**
32	209734	216570	218035	216747	216747	226545	228732	218855	**214934**
40	167787	175946	177214	177633	178223	**167792**	183433	175946	175946
48	139823	147898	147898	148832	149592	153260	153525	147944	**147756**
56	119848	127071	130479	**123857**	129624	133094	130949	126947	126570
64	104868	112498	115791	**103321**	115406	117638	116625	109591	116625
$p93791$									
W_{max}	**TAT(Clock Cycle)**								
16	1746657	1750830	1747504	-	1782067	1754980	1754980	1757452	**1720725**
24	1164442	1170620	1175988	-	1190565	1171190	1184630	1169945	**1150467**
32	873334	877073	891103	-	890092	886038	900388	878493	**864460**
40	698670	704272	716112	-	707664	706820	724758	718005	**695200**
48	582227	587117	598286	-	609580	600986	611029	594575	**579761**
56	499053	505586	517692	-	517017	501057	520868	509041	**496559**
64	436673	442455	452951	-	452245	445748	458389	447974	**435838**
$p34392$									
W_{max}	**TAT(Clock Cycle)**								
16	932790	939855	**931588**	-	935649	939855	939855	944768	935649
24	**621903**	626122	631035	-	635237	641514	637263	628602	626122
32	**544579**	**544579**	**544579**	-	**544579**	**544579**	**544579**	**544579**	**544579**
40	**544579**	**544579**	**544579**	-	**544579**	**544579**	**544579**	**544579**	**544579**
48	**544579**	**544579**	**544579**	-	**544579**	**544579**	**544579**	**544579**	**544579**
56	**544579**	**544579**	**544579**	-	**544579**	**544579**	**544579**	**544579**	**544579**
64	**544579**	**544579**	**544579**	-	**544579**	**544579**	**544579**	**544579**	**544579**

$p22810$, $p93791$ and $h953$ respectively, corresponding to different power limits. While the works reported in [53, 98, 38, 29, 77] consider only this global peak power values, [109, 110] have reported scheduling results for both global and cycle-accurate power models. Our results noted under the column marked "PSO GP" are better than other techniques including the cycle-accurate model of [109, 110], for most of the cases. It may be noted that, benchmark $h953$ does not show any improvement in result with the increase in the available test resources, although it is expected to have better TAT for higher values of W_{max}, which encourages more test concurrency. This is because all cores of $h953$ have *pareto-optimal* points with lesser TAM values. Individual test times of the cores do not improve with higher assigned TAM. However, other three benchmarks show a significant improvement in TAT for higher resource values. The fact that the test time results in "PSO GP" are often better than cycle-accurate power model based results of [109, 110] indicates the quality of the formulated PSO. It also encourages us to see the effect of using PSO under more elaborate power models, such as, window-based one, suggested in this chapter.

Table 3.4: Comparisons Of PSO With Other Scheduling Procedure For $d695$ With Different P_{max} And W_{max} (Considering Global Peak Power Model)

$d695$	**MC[98]**	**3D-Bin[53]**	**Cycle Accurate[109]**		**GA[38]**	**ACO[29]**	**SFLA[77]**	**PSO GP**
			GP	**CA**				
P_{max} = **1500**								
W_{max}	**TAT(Clock Cycle)**							
16	43541	45560	47009	44936	**42189**	42658	41226	42268
24	32663	31028	31458	30663	30054	**29401**	30418	29571
32	26973	27573	27544	23169	23297	23267	23076	**23059**
40	24369	20914	23937	19200	**18883**	-	-	19277
48	23425	20914	20842	**17013**	17083	18597	18348	18604
56	19402	16841	18909	**15230**	15670	15856	16727	16191
64	19402	16841	16875	**12941**	17695	-	-	16191
P_{max} = **2000**								
W_{max}	**TAT(Clock Cycle)**							
16	42450	43221	44870	44502	41804	41977	**41295**	41771
24	29106	29419	30926	30506	**27999**	28389	28951	28211
32	21942	24171	25048	22544	21592	21524	23076	**21375**
40	18691	19206	20925	18799	**17067**	-	-	17139
48	17467	17825	18553	16506	15652	**15625**	17186	15872
56	14563	14128	17013	13185	**12987**	**12987**	14945	13267
64	14469	14128	14397	**11526**	14434	-	-	13778
P_{max} = **2500**								
W_{max}	**TAT(Clock Cycle)**							
16	41847	43221	44502	44502	41718	41872	**41295**	41740
24	29106	29023	30926	30336	**27999**	28289	41295	**27999**
32	21931	23721	23525	22544	21236	21484	23076	**21042**
40	18691	19206	18988	18799	**16965**	-	-	17071
48	17257	15847	16506	16506	**14434**	14974	17211	14896
56	13963	14128	14834	13185	12862	12877	**12327**	12826
64	13394	12993	13098	**11526**	11646	-	-	11985

Table 3.5: Comparisons Of PSO With Other Scheduling Procedure For $p22810$ With Different P_{max} And W_{max} (Considering Global Peak Power Model)

$p22810$	**MC[98]**	**Cycle Accurate[109, 110]**		**GA[38]**	**ACO[29]**	**SFLA[77]**	**PSO GP**
		GP	**CA**				
P_{max} = **3000**							
W_{max}	**TAT(Clock Cycle)**						
16	482963	664511	536978	431475	443285	394466	427890
24	392525	607451	395389	290113	295752	291545	288228
32	309255	543358	349530	222095	231924	246524	221770
40	356215	427876	314067	181489	-	-	180979
48	311632	363299	287642	156041	160936	161449	155450
56	293528	362487	280548	138346	148143	149060	134486
64	293021	350162	280548	120414	-	-	118626
P_{max} = **5000**							
W_{max}	**TAT(Clock Cycle)**						
16	472026	504509	458812	428852	436930	433920	425113
24	382507	365562	331169	286352	296466	287967	285086
32	321930	320386	275106	217083	225300	234565	215940
40	264038	289649	240829	175946	-	-	175946
48	266166	229998	225179	147898	152414	165436	148268
56	257600	219244	215183	127382	133101	147483	127591
64	246110	219244	197593	116625	-	-	115628
P_{max} = **10000**							
W_{max}	**TAT(Clock Cycle)**						
16	473418	450546	450051	-	436730	433920	425113
24	352834	329419	308374	-	293783	287967	284701
32	236186	260711	241841	-	225300	220594	214934
40	195733	241182	199748	-	-	-	175946
48	159994	216632	179482	-	151256	160605	147171
56	138542	207606	166585	-	131821	147483	126570
64	128332	185309	160485	-	-	-	116625

3.7.3. Test Scheduling With Window-Based Peak Power Model

Working with the window-based power model requires detailed knowledge about the test patterns and the corresponding core power profiles. In the absence of this information for the benchmarks, we have randomly generated the test pattern sets for cores, guided by the number of inputs, bidirectional lines, outputs, scan cells. For each core, appropriate numbers of scan chains have been generated and transitions in them for the test sets have been calculated for each cycle. Total number of scan transitions in a cycle has been taken as a measure of power consumption by the core at that cycle. Since power profiles are dependent on the wrapper configurations, we have also computed the global peak values for individual wrapper configurations of cores. This we call TAM width dependent, global peak power (marked as TP in Tables 3.9, 3.10 and 3.11). Window-based peak power is marked as WP in the tables. For different power limits, test time results have been noted in Tables 3.9, 3.10 and 3.11, for different SoC benchmarks. Here also, in most of the cases, results corresponding to the window-based peak power model are better than those for simple global peak power model (GP) and TAM-width dependent peak-power model (TP). The last

Table 3.6: Comparisons Of PSO With Other Scheduling Procedure For $p93791$ With Different P_{max} And W_{max} (Considering Global Peak Power Model)

p_{93791}	MC[98]	Cycle Accurate[109, 110]		GA[38]	ACO[29]	SFLA[77]	PSO GP
		GP	CA				
P_{max} = **10000**							
W_{max}	**TAT(Clock Cycle)**						
16	1827819	-	-	1770954	1815761	1803224	**1744001**
24	1220469	-	-	1192015	1227691	1210668	**1175050**
32	1117385	-	-	1033988	1031103	1222874	**980153**
40	1091210	-	-	**841594**	-	-	871920
48	691866	-	-	609751	639255	626725	**606544**
56	629051	-	-	573720	573720	573720	**556735**
64	568734	-	-	552410	-	-	**441808**
P_{max} = **20000**							
W_{max}	**TAT(Clock Cycle)**						
16	1827819	1835416	1829232	1759656	1787856	**1660342**	1730385
24	1220469	1233680	1233716	1174517	187161	1203156	**1159733**
32	957921	932323	934069	886869	912503	993822	**871780**
40	821575	766353	769378	712053	-	-	**702252**
48	658132	640602	640615	600632	623013	611527	**583762**
56	549669	550636	539815	508947	573720	598228	**507406**
64	493599	485297	492463	450977	-	-	**444511**
P_{max} = **25000**							
W_{max}	**TAT(Clock Cycle)**						
16	1827819	1829176	1829232	1756326	-	-	**1730030**
24	1220469	1233680	1233716	1173939	-	-	**1155331**
32	965383	929974	934069	883885	-	-	**868726**
40	821475	748140	748154	708150	-	-	**697041**
48	639217	612046	625476	599339	-	-	**580743**
56	549669	545322	539815	508437	-	-	**500678**
64	493599	485297	481893	449657	-	-	**438386**

columns of the Tables 3.9, 3.10 and 3.11 have reported the TAT results without power constraint. It may be noted that TAT reduces with relaxation of the power budget (P_{max}). Finally, it reaches its lowest value where power is not considered as a design constraint (i.e. $P_{max} = \infty$). In some of the cases, it reaches its minimal value at a higher P_{max} suggesting that no further improvement is possible with more relaxation of the power budget. Some of the SoCs show no improvement in TAT with increase in the power budget or allocated test resources. This is due to the presence of bottleneck cores with much longer test time compared to other cores present in the SoC. This kind of cores dominate the schedule leaving no space of improvement by increasing P_{max} and W_{max} values.

To determine the CPU time saving achievable by our power model we have run the algorithm "*Schedule_Rectangle*" (Algorithm 3.1) for a typical particle using three different power models global peak, window-based peak and cycle-accurate one. Experiments have been carried out on a system with 2.40 GHz Intel Core2 Duo processor having 3 GB main memory. Table 3.8 notes the CPU times for the three approaches (single particle, single generation). It also reports the window-size that we have considered for each SoC. It can be noted that our window-based power model takes time one order of magnitude higher than the global peak based

Table 3.7: Comparisons Of PSO With Other Scheduling Procedure For $h953$ With Different P_{max} And W_{max} (Considering Global Peak Power Model)

$h953$	**EA[132]**	**3D-Bin[53]**	**GA[38]**	PSO
$P_{max} = 6 \times 10^9$				
W_{max}	**TAT(Clock Cycle)**			
16	122636	122636	**119357**	**119357**
24	122636	122636	**119357**	**119357**
32	122636	122636	**119357**	**119357**
40	122636	122636	**119357**	**119357**
48	122636	122636	**119357**	**119357**
56	122636	122636	**119357**	**119357**
64	122636	122636	**119357**	**119357**
$P_{max} = 7 \times 10^9$				
W_{max}	**TAT(Clock Cycle)**			
16	**119357**	**119357**	**119357**	**119357**
24	**119357**	**119357**	**119357**	**119357**
32	**119357**	**119357**	**119357**	**119357**
40	**119357**	**119357**	**119357**	**119357**
48	**119357**	**119357**	**119357**	**119357**
56	**119357**	**119357**	**119357**	**119357**
64	**119357**	**119357**	**119357**	**119357**
$P_{max} = 8 \times 10^9$				
W_{max}	**TAT(Clock Cycle)**			
16	**119357**	**119357**	**119357**	**119357**
24	**119357**	**119357**	**119357**	**119357**
32	**119357**	**119357**	**119357**	**119357**
40	**119357**	**119357**	**119357**	**119357**
48	**119357**	**119357**	**119357**	**119357**
56	**119357**	**119357**	**119357**	**119357**
64	**119357**	**119357**	**119357**	**119357**

model. On the other hand, the cycle accurate power model takes time almost two orders of magnitude higher than the window-based model. This justifies the usage of window-based power model for test scheduling approaches using evolutionary techniques, like PSO, to arrive at good scheduling results.

Table 3.8: Comparison Of CPU Time For Different Power Models For All ITC'02 Benchmarks (Single Particle, Single Generation)

Circuit Name	CPU Time (Sec.)			**Window Size**
	GP	**WP**	**CA**	**(Clock Cycle)**
$d695$	0.002	0.006	0.279	1000
$p22810$	0.002	0.031	12.279	1000
$p93791$	0.002	0.028	17.386	5000
$p34392$	0.002	0.027	10.343	1000
$g1023$	0.002	0.005	0.421	1000
$t512505$	0.004	0.035	13.724	5000
$q12710$	0.003	0.021	1.24	1000
$f2126$	0.002	0.012	0.341	1000
$d281$	0.002	0.009	0.327	1000
$u226$	0.001	0.005	0.179	1000
$h953$	0.002	0.018	7.42	1000
$a586710$	0.003	0.032	9.21	5000

Table 3.9: Comparisons Of GP And TP With WP Using PSO For Different ITC'02 SoCs For Different P_{max} And W_{max}

	GP	**TP**	**WP**	**GP**	**TP**	**WP**	**GP**	**TP**	**WP**	
$d695$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **1500**			P_{max} = **2000**			P_{max} = **3000**			$P_{max} = \infty$
16	42083	42075	**41981**	41981	**41771**	41833	41843	**41771**	**41771**	**41749**
24	28682	28682	**28132**	28292	28242	**28132**	28240	28009	**27979**	**27997**
32	**21476**	**21476**	**21476**	**21314**	21375	21318	21306	21375	**21136**	**21136**
40	18245	**17725**	18023	17307	**17050**	17093	17039	17050	**16962**	**16962**
48	16284	16379	**15898**	14974	14974	**14688**	14538	**14310**	**14310**	**14310**
56	**14797**	**14797**	**14797**	12642	12642	**12547**	12192	12158	**12155**	**12134**
64	14797	14797	**14680**	11322	11978	**10875**	10869	11040	**10817**	**10723**
$p22810$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **8000**			P_{max} = **10000**			P_{max} = **12000**			$P_{max} = \infty$
16	434786	434786	**433430**	433015	432638	**431966**	433015	432638	**431966**	**425113**
24	294960	293732	**292548**	290637	**288933**	289845	**288633**	288933	**288633**	**284851**
32	223394	224203	**222487**	**218322**	221748	218613	**218322**	218613	**218322**	**214934**
40	178402	178015	**177763**	178402	178015	**177763**	177763	177763	**177312**	**175946**
48	152237	152106	**151001**	152237	152106	**151001**	151796	**150413**	**150413**	**147756**
56	135571	129520	**129292**	135571	129520	**129292**	129624	129399	**129292**	**126570**
64	119852	117180	**116625**	**116625**	**116625**	**116625**	**116625**	**116625**	**116625**	**116625**
$p93791$	**TAT (Clock Cycle)**									
	P_{max} = **15000**			P_{max} = **20000**			P_{max} = **25000**			$P_{max} = \infty$
16	1780678	1775971	**1771797**	1769798	1775971	**1769673**	1766795	1772665	**1763891**	**1720725**
24	1199309	1188150	**1178986**	1197543	1185322	**1178986**	1195045	1183200	**1178741**	**1150467**
32	894150	897678	**888064**	890300	891100	**888064**	890300	891100	**888038**	**864460**
40	722541	722286	**718005**	722541	721215	**710508**	718005	718005	**710508**	**695200**
48	603951	602658	**592200**	600710	599453	**592200**	600710	592798	**592200**	**579761**
56	520719	520868	**513658**	516802	517238	**512188**	512614	514462	**509565**	**496559**
64	456848	452943	**450523**	451659	452943	**449633**	451659	447244	**447244**	**435838**
$p34392$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **3700**			P_{max}= **5000**			P_{max} = **7000**			$P_{max} = \infty$
16	1058834	1032630	**1014485**	999487	993478	**981450**	993721	992627	**962527**	**957124**
24	876014	876014	**862168**	862168	792553	**781178**	702852	698818	**663193**	**655607**
32	858762	858762	**781178**	792553	792553	**781178**	544579	544579	**544579**	**544579**
40	838643	792553	**781178**	792553	792553	**781178**	544579	544579	**544579**	**544579**
48	792553	792553	**781178**	792553	792553	**781178**	544579	544579	**544579**	**544579**
56	792553	792553	**781178**	792553	792553	**781178**	544579	544579	**544579**	**544579**
64	785910	785910	**781178**	785910	781178	**781178**	544579	544579	**544579**	544579

Table 3.10: Comparisons Of GP And TP With WP Using PSO For Different ITC'02 SoCs For Different P_{max} And W_{max} (Contd.)

	GP	**TP**	**WP**	**GP**	**TP**	**WP**	**GP**	**TP**	**WP**	
$f2126$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **5000**			P_{max} = **7000**			P_{max} = **10000**			$P_{max} = \infty$
16	613237	496251	**474497**	492447	492447	**474497**	**357088**	**357088**	**357088**	**357088**
24	613237	496251	**474497**	492447	492447	**470693**	357088	357088	**335334**	**335334**
32	613237	496251	**474497**	492447	492447	**470693**	357088	357088	**335334**	**335334**
40	613237	496251	**474497**	492447	492447	**470693**	357088	357088	**335334**	**335334**
8	613237	496251	**474497**	492447	492447	**470693**	357088	357088	**335334**	**335334**
56	613237	496251	**474497**	492447	492447	**470693**	357088	357088	**335334**	**335334**
64	613237	496251	**474497**	492447	492447	**470693**	357088	357088	**335334**	**335334**
$q12710$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **7000**			P_{max} = **10000**			P_{max} = **15000**			$P_{max} = \infty$
16	5032534	4895391	**4640215**	3443783	3483203	**3177502**	**2222349**	**2222349**	**2222349**	**2222349**
24	5032534	4895391	**4640215**	3443783	3483203	**3177502**	**2222349**	**2222349**	**2222349**	**2222349**
32	5032534	4895391	**4640215**	3443783	3483203	**3177502**	**2222349**	**2222349**	**2222349**	**2222349**
40	5032534	4895391	**4640215**	3443783	3483203	**3177502**	**2222349**	**2222349**	**2222349**	**2222349**
48	5032534	4895391	**4640215**	3443783	3483203	**3177502**	**2222349**	**2222349**	**2222349**	**2222349**
56	5032534	4895391	**4640215**	3443783	3483203	**3177502**	**2222349**	**2222349**	**2222349**	**2222349**
64	4895391	4675639	**4640215**	3443783	3483203	**3177502**	**2222349**	**2222349**	**2222349**	**2222349**
$t512505$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **21000**			P_{max} = **21500**			P_{max} = **22000**			$P_{max} = \infty$
16	10531175	10531171	**10531159**	10531171	10531171	**10531159**	10531048	10530995	**10530995**	**10530995**
24	10531633	1453565	**10453512**	10454163	10453565	**10453512**	10454163	10454163	**10453470**	**10453470**
32	5359882	5326372	**5268991**	5268875	5268875	**5268868**	5268936	5269411	**5268868**	**5268868**
40	5269239	5228759	**5228474**	5228827	5228759	**5228420**	5228420	5228420	**5228420**	**5228420**
48	5269139	5269139	**5228474**	5228474	5228474	**5228420**	5228420	5228420	**5228420**	**5228420**
56	5269139	5269139	**5228452**	5228474	5228474	**5228420**	5228420	5228420	**5228420**	**5228420**
64	5269139	5269139	**5228452**	5228474	5228474	**5228420**	5228420	5228420	**5228420**	**5228420**
$a586710$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **500**			P_{max} = **750**			P_{max} = **1000**			$P_{max} = \infty$
16	42355169	**42198943**	**42198943**	42117536	42117536	**42067708**	**41547436**	**41547436**	**41547436**	**41547436**
24	28797909	28450601	**28329306**	28329306	28316901	**28148166**	27907180	**27785885**	**27785885**	**27785885**
32	22973201	22498601	**22006640**	21694396	21694396	**21656012**	21343768	21343768	**21058768**	**21058768**
40	**19122170**	**19122170**	**19122170**	**19048835**	**19048835**	**19048835**	**19041307**	**19041307**	**19041307**	**19041307**
48	**16764592**	**16764592**	**16764592**	15315467	15315467	**15212440**	15112162	15112162	**15031299**	**15031299**
56	**14090716**	**14090716**	**14090716**	14669018	14669018	**14669018**	13481897	**13401034**	**13401034**	**13401034**
64	13401034	13401034	**12754584**	12754584	12700205	**12510356**	11567464	11567464	**11486601**	**11486601**

Table 3.11: Comparisons Of GP And TP With WP Using PSO For Different ITC'02 SoCs For Different P_{max} And W_{max} (Contd.)

	GP	**TP**	**WP**	**GP**	**TP**	**WP**	**GP**	**TP**	**WP**	
$g1023$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **550**			P_{max}= **800**			P_{max} = **1000**			$P_{max} = \infty$
16	33597	33491	**33133**	31730	31560	**31503**	31431	31209	**31184**	**31059**
24	29400	29338	**26212**	23586	22149	**22038**	21909	21445	**21252**	**21135**
32	27584	27527	**25530**	20733	20733	**19742**	17165	17212	**17090**	**15896**
40	27584	27526	**25530**	20733	20733	**19742**	16945	**16945**	**14794**	**14794**
48	27584	27526	**25530**	20733	20733	**19742**	15444	15299	**14794**	**14794**
56	27584	27526	**25530**	20733	20733	**19738**	15444	15299	**14794**	**14794**
64	27584	27526	**25526**	20733	20733	**19733**	15444	15299	**14794**	**14794**
$d281$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **400**			P_{max} = **500**			P_{max} =**700**			$P_{max} = \infty$
16	8718	8393	**8290**	8127	8127	**8051**	7906	7988	**7906**	**7906**
24	7214	7123	**6957**	6622	6622	**6532**	5567	5590	**5428**	**5428**
32	6379	6379	**6284**	5987	6049	**5987**	4163	4163	**3926**	**3926**
40	6125	6125	**5904**	5711	5749	**5512**	4163	4163	**3926**	**3926**
48	5808	5808	**5615**	5353	5353	**5353**	4163	4163	**3926**	**3926**
56	5693	5561	**5360**	5277	5277	**5194**	4163	4163	**3926**	**3926**
64	5596	5456	**5360**	5353	**5036**	**5036**	4163	4163	**3926**	**3926**
$u226$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **700**			P_{max} = **1000**			P_{max} = **1200**			$P_{max} = \infty$
16	**18663**	**18663**	**18663**	**18663**	**18663**	**18663**	**18663**	**18663**	**18663**	**18663**
24	**13331**	**13331**	**13331**	**13331**	**13331**	**13331**	**13331**	**13331**	**13331**	**13331**
32	**10665**	**10665**	**10665**	**10665**	**10665**	**10665**	**10665**	**10665**	**10665**	**10665**
40	**8084**	**8084**	**8084**	**8084**	**8084**	**8084**	**8084**	**8084**	**8084**	**8084**
48	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**
56	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**
64	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**	**7999**
$h953$	**TAT (Clock Cycle)**									
W_{max}	P_{max} = **900**			P_{max} = **1500**			P_{max} = **2000**			$P_{max} = \infty$
16	241151	212081	**211533**	179940	179940	**177496**	**119357**	**119357**	**119357**	**119357**
24	241151	212081	**211533**	179940	179940	**177496**	**119357**	**119357**	**119357**	**119357**
32	241151	212081	**211533**	179940	179940	**177496**	**119357**	**119357**	**119357**	**119357**
40	241151	212081	**211533**	179940	179940	**177496**	**119357**	**119357**	**119357**	**119357**
48	241151	212081	**211533**	179940	179940	**177496**	**119357**	**119357**	**119357**	**119357**
56	241151	212081	**211533**	179940	179940	**177496**	**119357**	**119357**	**119357**	**119357**
64	241151	212081	**211533**	179940	179940	**177496**	**119357**	**119357**	**119357**	**119357**

3.7.4. Test Scheduling In Multi-Frequency Test Environment

In this subsection, we have the shown the comparison between single-frequency and multi-frequency test environment. Table 3.12 shows comparative study between window-based power model and global peak power model as well as between single frequency testing and multi-frequency testing. To have a better understanding of the effect of different power models on TAT, we have compared the window-based power model results with global peak power model, keeping multi-frequency test environment as constant for both the cases. Similarly the comparison between multi-frequency approach and single frequency approach has been carried out under same window-based peak power profile. In Table 3.12, the global peak multi-frequency approach is marked as "GPMF", while "WPSF" and "WPMF" indicate window-based peak power single frequency approach and window-based peak power multi-frequency approach respectively.

It is clear from Table 3.12 that a significant improvement in TAT can be achieved by using multi-frequency test environment. It may be noted from column 3 that our multi-frequency approach is able to generate test schedule under low power constraint, while single frequency approach fails to do that. It is justifiable as under multi-frequency test environment, power hungry cores can be tested at low frequency to minimize their power consumption. A proper selection of operating frequency of each core actually helps to reduce TAT as well as encourage low power test scheduling. It may also be noted that our window-based peak power model performs reasonably better than global peak power model under same multi-frequency environment.

Finally, in Figure 3.12, we have reported a resulting test schedule of window-based power model for $d695$ with $P_{max} = 2000$ and $W_{max} = 64$ under single-frequency test environment.

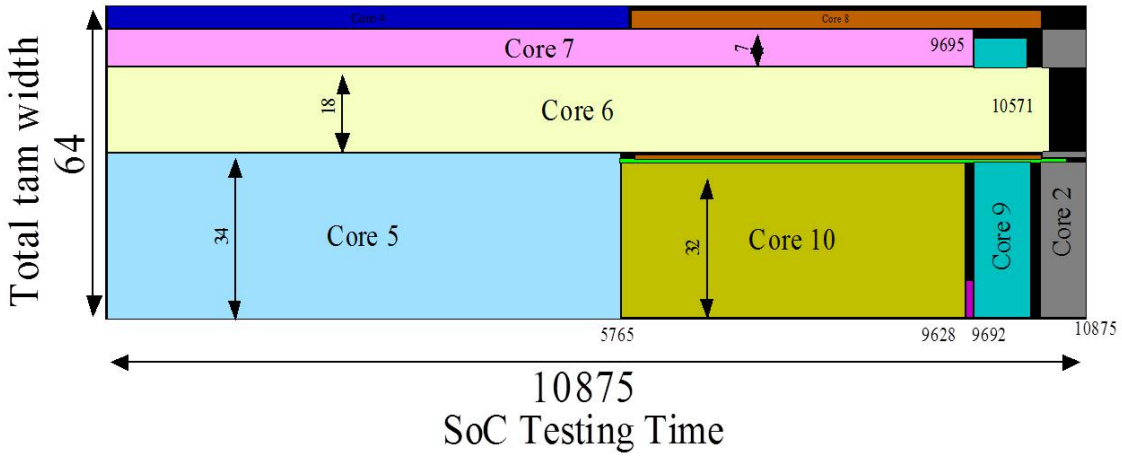


Figure 3.12: Scheduling of $d695$ with $P_{max} = 2000$ and $W_{max} = 64$ (window-based peak power model with single frequency test environment)

Table 3.12: Comparison Of Different Test Scheduling Approaches For Different P_{max} And W_{max} For ITC'02 Benchmarks

$d695$	P_{max} = **1000**			P_{max} = **1500**			P_{max} = **2000**			P_{max} = **3000**		
W_{max}	Test Time (Clock Cycle)											
	GPMF	**WPSF**	**WPMF**	**GPMF**	**WPSF**	**WPMF**	**GPMF**	**WPSF**	**WPMF**	**GPMF**	**WPSF**	**WPMF**
16	41200	NS*	**40824**	40824	41981	**40477**	38722	41833	**38587**	38254	41771	**38144**
24	30288	NS	**30195**	27744	28132	**26964**	27979	28132	**26964**	27489	27489	**26486**
32	**21767**	NS	**21767**	21126	21476	**20638**	20640	21318	**20568**	20638	21136	**20472**
40	18695	NS	**18416**	17093	18023	**16962**	16915	17093	**16914**	**16701**	16962	**16701**
48	18549	NS	**17889**	14453	15898	**14348**	14388	14688	**14240**	**14016**	14310	**14016**
56	18509	NS	**17889**	12579	14797	**12365**	12324	12547	**12320**	12247	12155	**12155**
64	18323	NS	**17889**	12157	14680	**11720**	10856	10875	**10817**	10611	10817	**10571**

$p22810$	P_{max} = **6500**			P_{max} = **8000**			P_{max} = **10000**			P_{max} = **12000**		
W_{max}	Test Time (Clock Cycle)											
	GPMF	**WPSF**	**WPMF**	**GPMF**	**WPSF**	**WPMF**	**GPMF**	**WPSF**	**WPMF**	**GPMF**	**WPSF**	**WPMF**
16	383956	NS	**381990**	381990	433430	**381990**	**381990**	431966	**381990**	**381990**	431966	**381990**
24	**290834**	NS	**290834**	278641	292548	**271142**	274227	289845	**271142**	274227	288633	**271142**
32	**213318**	NS	**213318**	211202	222487	**206365**	206961	218613	**201162**	203404	218322	**201162**
40	174223	NS	**168529**	170372	177763	**167868**	168291	177763	**167868**	168210	177312	**167490**
48	147622	NS	**145962**	**145962**	151001	**145962**	145962	151001	**145102**	145662	150413	**145102**
56	140040	NS	**134266**	129574	129292	**125909**	125356	129292	**125278**	125356	129292	**123938**
64	**128399**	NS	**128399**	**116625**	**116625**	**116625**	114832	116625	**111401**	110450	116625	**109040**

$p93791$	P_{max} = 12000			P_{max} = 15000			P_{max} = 20000			P_{max} = 25000		
W_{max}	Test Time (Clock Cycle)											
	GPMF	**WPSF**	**WPMF**	**GPMF**	**WPSF**	**WPMF**	**GPMF**	**WPSF**	**WPMF**	**GPMF**	**WPSF**	**WPMF**
16	1555974	NS	**1500754**	1541789	1771797	**1500754**	1520095	1769673	**1479144**	1483264	1763891	**1479713**
24	1097268	NS	**1086643**	1086643	1178986	**1078119**	1078119	1178986	**1072684**	1074274	1178741	**1052881**
32	862866	NS	**845676**	862749	888064	**835715**	836302	888064	**835442**	836302	888038	**835442**
40	704919	NS	**704271**	700943	718005	**691604**	674746	710508	**664247**	666076	710508	**664247**
48	590197	NS	**589752**	588713	592200	**580268**	586362	592200	**581577**	582601	592200	**572976**
56	515930	NS	**513953**	513953	513658	**513658**	505525	512188	**498674**	503408	509565	**460304**
64	454672	NS	**448202**	442768	450523	**441430**	440796	449633	**440241**	440796	447244	**440241**

* NS - No Schedule Possible

3.8. Conclusion

In this chapter, a new window-based power model has been proposed to generate power-aware test schedule for SoC. The developed model has been shown to reduce the power overestimation in the scheduling process. The model has been incorporated into a PSO based formulation to select test rectangles for cores and scheduling them using a 3-D bin-packing approach. Our PSO guided test scheduling approach produces better results in terms of test application time (TAT) in most of the cases for both test scheduling with and without power constraint. Our multi-frequency test approach helps to reduce overall test application time as well as capable of low power test scheduling. However, with a small modification in the MUX based multi-frequency architecture, our scheduling procedure can be used for multi-clock domain SoC testing.

Chapter 4
Thermal-Aware Test Scheduling

4.1. Introduction

In the last chapter, we have proposed power-aware test scheduling techniques for System-on-Chip. It is worth mentioning that ensuring power validation may not be sufficient to guarantee thermal safety of the chip. Although, power has a major contribution in the temperature increase of a chip, there are some other parameters like floorplan, power density of cores, core size and their relative positions etc, which play important role to determine the temperature of the chip. Efficient cooling technology, quality packaging of the chip may be good thermal safe solutions, but not cost effective. Low frequency testing may also avoid high temperature during testing, but also leads to longer test application time (TAT), which is not desirable for at-speed testing. Other solution is incorporation of some thermal constraint in scheduling strategy to ensure no thermal violation during testing. Many thermal simulators, such as, Hotspot [118] have been proposed that can calculate temperature of cores within the SoC. It may be noted that thermal simulation is time consuming. As, test scheduling problem is NP-Hard [56], scheduling strategy has to explore a large search space to obtain minimal test application time of the chip, which itself is time consuming. Integrating a thermal simulator with test scheduling strategy to check thermal validation at every scheduling point may generate accurate thermal safe test schedule, but may not be feasible as it takes long time to generate test schedule with relatively minimal TAT.

To overcome this problem, most of the thermal-aware test-scheduling strategies use some thermal models [28, 143, 14, 144, 47, 137, 136] to predict temperature of the cores during scheduling without using thermal simulations. However, it may be noted that these thermal models may not always predict temperature with reasonably good accuracy due to lots of assumptions in them.

In this chapter, we have proposed a superposition principle based fast thermal model to estimate the temperature of the cores accurately. It requires only few summations of some precomputed temperature values, hence can estimate the

temperature very fast. It may be noted that, exact thermal behaviour of a chip not only requires an accurate thermal model, but also exact power profiles of the cores as well as accurate floorplan information of the chip. In absence of all these information of commonly used ITC'02 benchmarks, most of the works consider single fixed peak power [28, 53, 98, 18, 76, 38, 77, 29] throughout the testing of a core, which introduces inaccuracy in the predicted thermal behaviour of the chip. Few other works although consider power at each cycle, take good amount of time to generate the schedule. To deal with the problem of power profile generation of the cores of the SoC, we have used the window-based peak power model proposed in Section 3.4. This is an intermediate approach to reduce the power overestimation of global peak power model as well as the computational complexity of cycle-accurate power model. Both of the power and thermal models have been integrated with a Particle Swarm Optimization (PSO) guided 3D-bin packing approach to generate a power and thermal-safe test schedule of the SoC.

4.2. Motivation

Thermal simulators like HotSpot follows linear RC thermal model [137]. This linearity of thermal model can be exploited using the superposition principle. The work presented in [137, 136] tried to exploit the linearity of HotSpot and used a superposition principle based thermal model to calculate the thermal profile of a core during scheduling. As the CPU execution time is the main bottleneck of thermal simulation during scheduling, the authors in [137, 136] have tried to avoid invoking the thermal simulator in the scheduling process. Instead, they have used the HotSpot [118] thermal simulator to create offline thermal profiles of the cores. These thermal profiles are used for scheduling purpose. This thermal model is fast and requires few summations to calculate the temperature of a core. However, while working with this approach [137, 136], we have noticed that it may result in thermal violations due to inaccurate modelling of heating and cooling rates of cores. One such situation has been shown in Figure 4.1, in which the actual thermal profile during testing violates the temperature threshold used in the test schedule generation procedure. More such cases have been reported in Section 4.6.2.

This has motivated us to develop a more elaborate thermal model based on superposition principle. It uses relatively more detailed and accurate thermal information to get efficient yet fast thermal-safe test schedule.

4.3. Summary Of Contribution

The basic requirements to make a core testable and the problem of SoC test scheduling under power constraint has been mentioned in Chapter 3. The problem

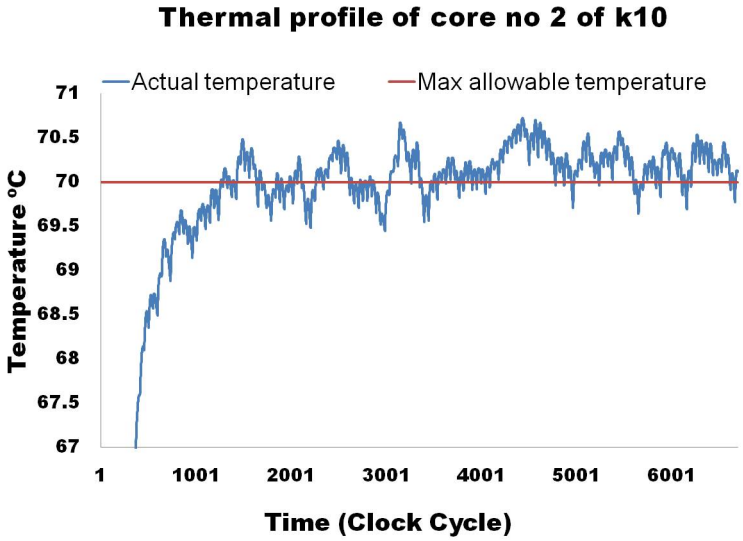


Figure 4.1: Temperature profile and maximum temperature limit of core 2 of $k10$ using the thermal model [137, 136]

of thermal-aware SoC test scheduling also includes this part of the problem. In addition to that, for thermal-safe test scheduling, the test engineers have to take care of an extra thermal constraint throughout the testing of the SoC. At any time, the peak temperature of any core must not violate a certain limit. To check the thermal constraint, information regarding the temperature of the cores are needed, which requires online thermal simulation or at-least a thermal model that can estimate the temperature of the cores without any time consuming online thermal simulation.

Moreover, information regarding test data and floorplan are important to get the exact thermal behaviour of the cores. As the traditional ITC'02 SoC benchmark suite does not have all these details, most of the works in the literature consider some arbitrary values of these parameters, which introduce inaccuracy in the thermal behaviour of the cores. SoC with detail information regarding exact test data and area of the cores can avoid this problem.

The major contributions of this chapter are as follows,

1. A superposition principle based thermal model, which takes help of some offline thermal simulation data to estimate the temperature of a core. Our thermal model considers more detailed information relative to the work presented in [137, 136]. Experimental results show that our model estimates temperature better than [137, 136], hence assures thermal safety of the test schedule more efficiently.

2. A new set of SoCs have been introduced with every detail regarding test vectors, area, floorplan, dynamic and leakage power information to observe

exact thermal behaviour of the chip. Two such new SoC benchmarks have been introduced.

We have used the window-based peak power model mentioned in Section 3.4 to generate the power profiles of the cores of the newly formed SoCs. Superposition principle based thermal model has been used to estimate the temperature of the cores using these power profiles and the floorplan information of the SoCs. Finally a PSO guided test scheduling heuristic has generated a power and thermal-safe test schedule with minimal test application time.

4.4. Superposition Principle Based Thermal Model

As test scheduling problem is NP-Hard, finding an optimal schedule for SoCs with large number of embedded cores using any meta-search technique is time consuming. Most of the meta-search techniques are based on iterative methods, which try to find better solution over several iterations. Temperature of cores in the scheduling interval is required to get a thermal safe test schedule. It requires thermal simulation of the cores at the time of scheduling. However, one major drawback of the thermal simulators like HotSpot is their execution time, which restricts us to invoke it inside the scheduling procedure. An alternate solution is to incorporate some kind of thermal model in the scheduling procedure, which can predict the temperature of the cores during scheduling without incorporating thermal simulations.

We have used a superposition principle based thermal model, which takes the help of linearity of thermal model of HotSpot [136]. Our thermal model uses offline HotSpot [118] thermal simulations for each core in different possible conditions and creates thermal databases for all those conditions. These database information are used for the scheduling purpose. This superposition principle based thermal model, although do not use online Hotspot simulation, produces the same temperature as generated by Hotspot. It helps us to generate thermal safe test schedule much faster than the techniques that use online Hotspot simulations. This thermal superposition model has been suggested in [137, 136], however, our model is more general as it includes both heating and cooling of cores, subject to the conditions of its neighbors.

To get the exact thermal behavior of a core, it is very much important to observe different conditions, which can change the temperature of a core. The temperature of a core C_i can increase due to the following activities.

- Before C_i starts its testing, its initial temperature increases due to its leakage power consumption (Figure 4.2).

- During the period of testing of the core C_i, its temperature increases (Figure 4.3).
- If any number of other cores $C_j(1 \leq j \leq N)(j \neq i)$ are tested in parallel with C_i, due to the lateral spreading of heat from C_j, the temperature of C_i increases (Figure 4.3).
- Initial temperature of C_i increases due to the effect of all the cores which have already been scheduled before C_i starts its testing (Figure 4.4)

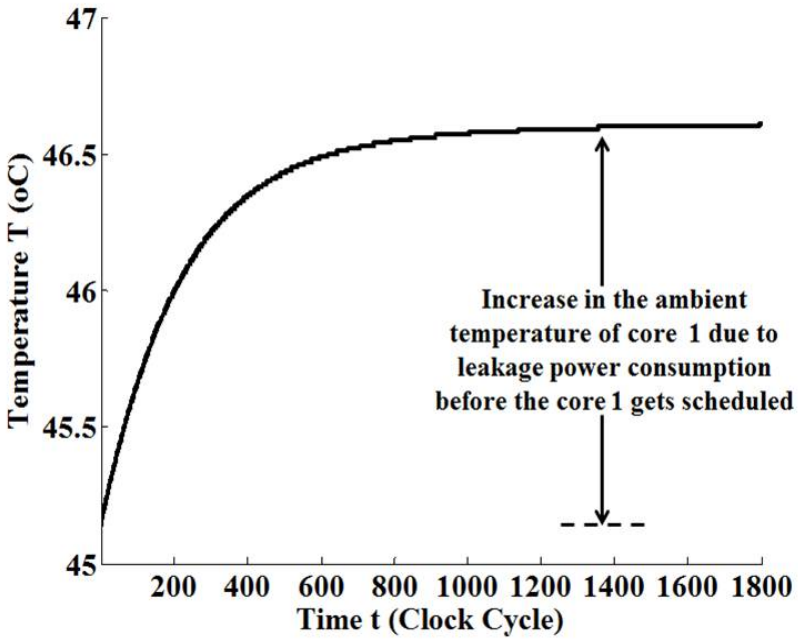


Figure 4.2: Pre-scheduling increase of the initial temperature of core 1 of SoC $k10$ due to leakage power

Figures 4.2, 4.3 and 4.4 are some simulation results on SoC $k10$ (details of the SoC has been mentioned in Section 4.6.1) to depict the exact thermal behavior of core no. 1 in different conditions.

Figure 4.2 shows the increase in the initial temperature due to leakage power consumption of core 1 before it starts its testing in SoC $k10$. It may be noted from the simulation result that, after certain clock cycles, there is no further increase in the initial temperature due to leakage power consumption.

To check the exact heating effect on a core during its test, different conditions need to be considered. First we have considered the situation in which only core 1 is active and all other cores are idle and measured the temperature increase of core 1, due to its own heating. After that we have shown the heating effects due to lateral heat spread from neighboring cores, which are tested in parallel. Figure 4.3 shows the extra heating effect on core 1, if it is active in parallel with

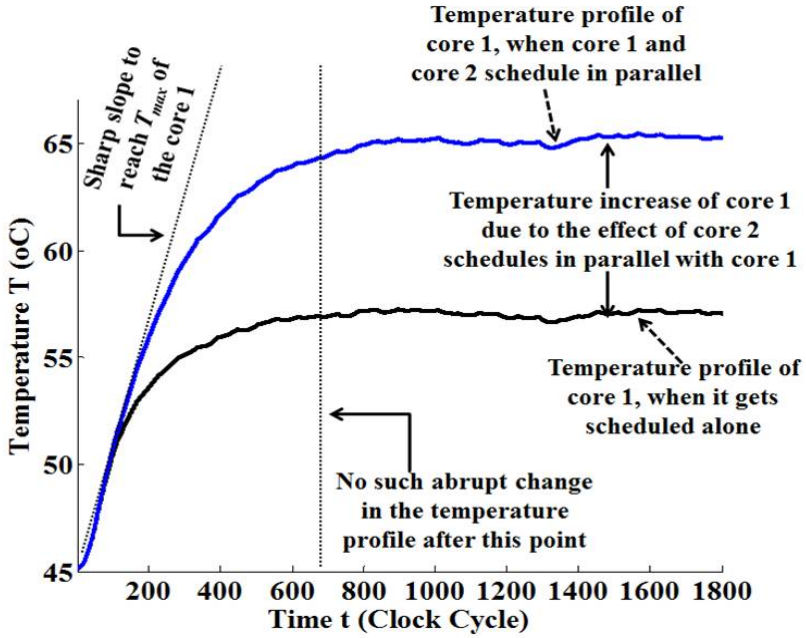


Figure 4.3: Temperature profile of core 1 of SoC $k10$ during its scheduling

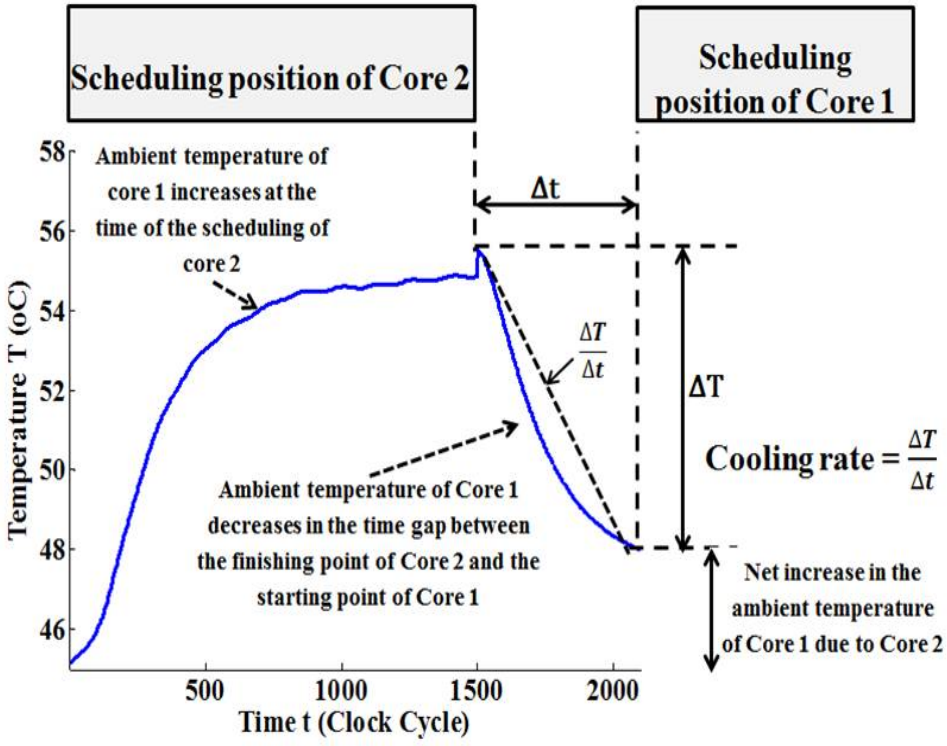


Figure 4.4: Temperature profile of core 1 as a effect of core 2 schedules before core1 of SoC $k10$

core 2. It may be noted that, lateral heat spreading of core 2 causes an extra temperature increase of up to 8°C of core 1, when it is tested in parallel with core 2. An important observation from Figure 4.3 is that the temperature rises to its peak value quickly, and after that the temperature profile is almost flat and no abrupt change can be noticed. It is quite justified from the fact that there are

not that many fluctuations in the middle of the power profile as the number of switching activities does not vary much from one cycle to the next cycle, causing little variations in the dynamic power consumption in the middle.

Figure 4.4 shows the effect of already scheduled cores on the initial temperature of a core, considered to be scheduled next. When core 1 is idle and core 2 is being tested, the initial temperature of core 1 increases by 10°C, because of lateral heat spreading from core 2. Now, if core 1 wants to start its test just after core 2 has finished, the initial temperature of core 1 will be 10°C more than its normal initial temperature. This will increase the maximum temperature of core 1 and if it is more than the temperature budget of the SoC, it may cause a permanent damage in the chip due to burning. However, if core 1 starts its test after a certain time interval from the completion of the test of core 2, in that time gap, its initial temperature reduces towards its normal initial temperature, which leads to more thermal safe schedule than the previous one. Thermal model should be able to handle all these conditions and generate a test schedule that does not violate the thermal constraint.

The objective of the superposition principle based thermal model is to sum up the individual effects of other cores on the temperature of a particular core to get their cumulative effect on the temperature of the core. The operation of our superposition principle based thermal model can be described by the following steps.

1. Calculate the increase in the initial temperature due to leakage power consumption for each core. Create a thermal database of it.

2. For each core $C_i(1 \leq i \leq N)$ calculate its temperature rise when it is tested alone. Then calculate extra increase in temperature of C_i due to core $C_j(1 \leq j \leq N)(i \neq j)$, when C_j is tested in parallel with C_i. This extra increase in temperature of C_i is calculated for each C_j separately. Create a thermal database of it. The database is basically a matrix of size $N \times N$. For each row i in the matrix, the element (i, i) stores the value of temperature increase of the i^{th} core, while it is tested alone. All the other elements (i, j) of i^{th} row store the values of extra increase of temperature of core C_i because of parallel testing of core C_j with core C_i. Superposition principle is applied to calculate the actual temperature increase of core C_i considering the cores, which are active in parallel with C_i during its testing.

3. For each core $C_i(1 \leq i \leq N)$ calculate its initial temperature increase due to the core $C_j(1 \leq j \leq N)(i \neq j)$, which is scheduled prior to C_i. Let us assume, the initial increase in temperature be T_1. Also calculate the cooling

rate for the combination of C_i and C_j. The initial temperature of C_i reduces in the time gap between the end point of C_j and start point of C_i. Cooling rate can be calculated from the slope of the temperature decrease curve as mentioned in figure 4.4. Although the temperature decrease portion of the curve is not exactly linear, still a linear prediction of this portion of the curve can be a good approximation to calculate the simplified cooling rate, as mentioned in Figure 4.4. If the time gap is Δt and the temperature decrease in Δt time is ΔT then,

$$\text{Cooling rate}(CR) = \frac{\Delta T}{\Delta t} \tag{4.1}$$

So the actual increase in the initial temperature of C_i is $(T_1 - \Delta T)$. This is done for each C_j corresponding to each C_i. Two thermal database matrices of size $N \times N$ store all these information. Superposition principle is applied to calculate actual increase in temperature of C_i, considering all C_j that are scheduled prior to C_i.

The heating effect of core C_j on core C_i has been summarized in Figure 4.5. If we consider the start time of testing of core C_i to be $StartC_i$ and end time to be $EndC_i$, the condition of parallelism between C_i and C_j can be described as follows. If $StartC_i \geq StartC_j$ and $EndC_j \geq StartC_i$ then C_i and C_j are active in parallel, while condition of nonparallelism is $StartC_i \geq StartC_j$ and $EndC_j \leq StartC_i$. Figure 4.6 describes the test paralellism between two cores C_i and C_j.

4.5. Test Scheduling Strategy

The thermal-aware SoC test scheduling problem can be described as follows.

4.5.1. Problem Formulation

Suppose a SoC with N cores $C_1, C_2 \cdots C_N$ is to be tested with a maximum of W_{max} TAM resources, a maximum power limit P_{max}, and a maximum temperature limit T_{max}. The test scheduling problem is to allocate TAM resources and test times to the cores so that, the total test application time (TAT) is minimized, while the power consumption during testing remains under P_{max} and the maximum temperature of any core does not cross the temperature upper bound T_{max}.

4.5.2. Test Schedule Generation

The thermal-aware test scheduling problem can be divided into three sub-problems;

1. Basic test scheduling

2. Power-aware test scheduling

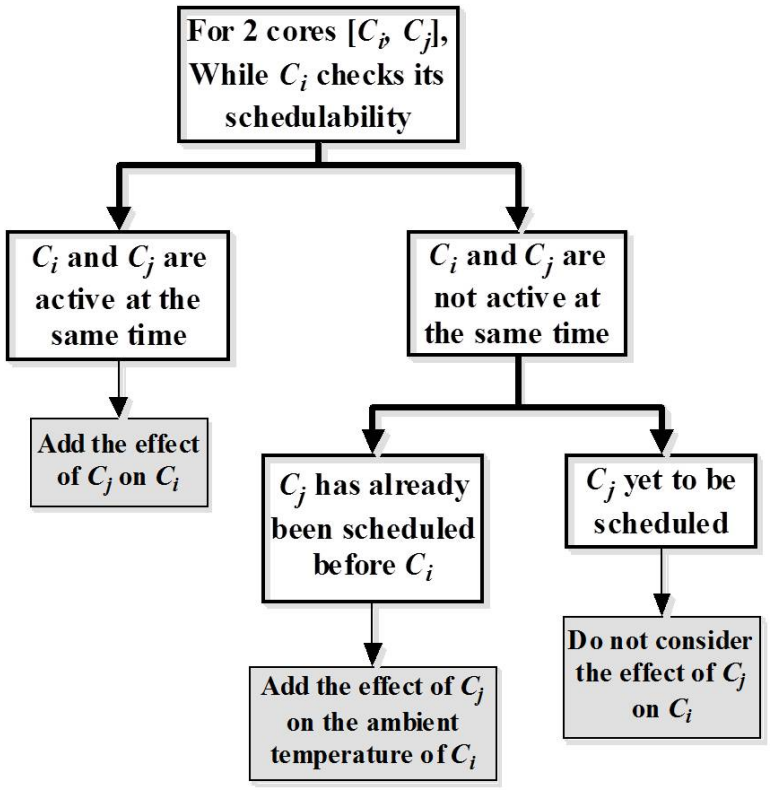


Figure 4.5: Different possibilities of temperature effect of core C_j on core C_i

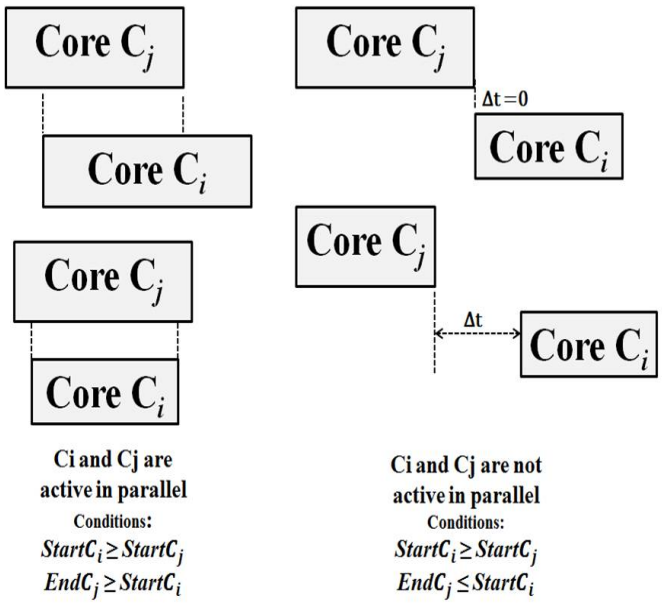


Figure 4.6: Different conditions of test parallelism between two cores C_j and C_i

3. Thermal-aware test scheduling

The solution of first two sub-problems have been mentioned in the Chapter 3. It converts the power-aware test scheduling problem into a 3D-rectangular bin-packing problem and solves it using a PSO based meta-search technique. In addition to that, a superposition principle based thermal model has been used in this

chapter to estimate the temperature of the cores during testing. This superposition principle based thermal model assures thermal safe test schedule with the help of some pre-calculated thermal databases. A thermal constraint controller is used for this purpose to check the thermal safety of the schedule at each scheduling point, where an unscheduled core is expected to get scheduled.

The overall scheduling strategy can be described by the following steps.

Step I:

This step is a stand-alone one, while the next stages need to be solved in an integrated fashion. The stage consists of the following.

- **Wrapper Design:**

We have used the $Design\_Wrapper$ algorithm [57] to generate different wrapper configurations for each core. For a core, the wrapper configurations corresponding to only the pareto-optimal TAM width [57] are noted. The functionality of $Design\_Wrapper$ algorithm [57] has been mentioned in Section 3.3.

- **Power Profile Generation:**

Window-based power profile corresponding to each pareto-optimal point of each core is determined using the method mentioned in Section 3.4. In Section 3.4, we have considered the total switching activities in wrapper chains in a particular clock cycle as the power consumption of that cycle. However, exact power values are required for accurate thermal estimation. To calculate exact power value of each core, we have first calculated the Average transition (ATR) for the total test time of the core. Dynamic power (DP) consumption of a core can be obtained using *Synopsys Design Vision Report Power* tool [119] as mentioned in the Table 4.1 in Section 4.6.1. If the transition in j^{th} cycle is TRN_j, the cycle-accurate power of the j^{th} cycle (CAP_j) can be estimated as

$$CAP_j = \frac{TRN_j}{ATR} \times DP \tag{4.2}$$

The window-based power profile of each core has been formed by dividing this new cycle-accurate power profile (CAP) into several window-intervals and considering peak power value in those window-intervals.

- **Thermal database creation:**

As mentioned in the thermal model in Section 4.4, our scheduling strategy does not invoke any thermal simulator during scheduling. Instead of that, we create four thermal databases to store different thermal values using offline HotSpot [118] thermal simulations for each core. $Thermal\_Database\_1(TD1)$ stores the values

of increase in the initial temperature due to leakage power consumption of each core. $Thermal\_Database\_2(TD2)$ is used to store the temperature values of self testing as well as parallel testing. The increase in the initial temperature of a core because of previously scheduled cores has been stored using two thermal databases $Thermal\_Database\_3(TD3)$ and $Thermal\_Database\_4(TD4)$. Cooling rates of each core can be calculated using $TD3$ and $TD4$. The procedures for generation of thermal databases are mentioned in $Procedure\_TD1$, $Procedure\_TD2$, $Procedure\_TD3$ and $Procedure\_TD4$ noted next.

Algorithm 4.1: *Procedure_ TD1*

```
1  begin
2      for each core C_i(1 ≤ i ≤ N) do
3          Create power profile P_leakage_i like this;
4              Leakage power of C_i for 2000 clock cycles;
5              No power for all other cores C_j(j ≠ i);
6          Feed P_leakage_i and floorplan of the SoC to HotSpot;
7          Calculate the maximum temperature value T_leakage_i from the
           transient and steady state responses of C_i;
8          Enter T_leakage_i in TD1;
```

Algorithm 4.2: *Procedure_ TD2*

```
1  begin
2      for each core C_i(1 ≤ i ≤ N) do
3          Create power profile P_self_i like this;
4              Dynamic power of C_i for the clock cycles required to test C_i;
5              Leakage power for all other cores C_j(j ≠ i) for the same number
               of clock cycles;
6          Feed P_self_i and floorplan of the SoC to HotSpot;
7          Calculate the maximum temperature value T_self_i from the transient
           and steady state responses of C_i;
8          Enter T_self_i in the (i, i) position of the matrix TD2;
9          for each core C_j(1 ≤ j ≤ N)(j ≠ i) do
10             Create power profile P_parallel_ij like this;
11                 Dynamic power of C_i and C_j for the clock cycles required to
                   test C_i;
12                 Leakage power for all other cores C_k(k ≠ j ≠ i) for the same
                   number of clock cycles;
13             Feed P_parallel_ij and floorplan of the SoC to HotSpot;
14             Calculate the maximum temperature value T_parallel_ij from the
               transient and steady state responses of C_i;
15             T_extra_ij = T_parallel_ij − T_self_i;
16             Enter T_extra_ij in the (i, j) position of the matrix TD2;
```

Algorithm 4.3: ***Procedure_ TD3***

```
1 begin
2   for each core C_i(1 ≤ i ≤ N) do
3     for each core C_j(1 ≤ j ≤ N)(j ≠ i) do
4       Create power profile P_preschedule1_ij like this;
5         Dynamic power of C_j for the clock cycles required to test C_j;
6         Leakage power for all other cores C_k(k ≠ j) including C_i for
          the same number of clock cycles;
7       Feed P_preschedule1_ij and floorplan of the SoC to HotSpot;
8       Calculate the maximum temperature value T_preschedule1_ij from the
        transient and steady state responses of C_i;
9       Enter T_preschedule1_ij in the (i, j) position of the matrix TD3;
```

Algorithm 4.4: ***Procedure_ TD4***

```
1 begin
2   for each core C_i(1 ≤ i ≤ N) do
3     for each core C_j(1 ≤ j ≤ N)(j ≠ i) do
4       Create power profile P_preschedule2_ij like this;
5         Dynamic power of C_j for the clock cycles required to test C_j;
6         Leakage power of C_j for next 2000 number of clock cycles;
7         Leakage power for all other cores C_k(k ≠ j) including Ci for
          the total number of clock cycles;
8       Feed P_preschedule2_ij and floorplan of the SoC to HotSpot;
9       Calculate the final temperature value T_preschedule2_ij of C_i;
10      Enter T_preschedule2_ij in the (i, j) position of the matrix TD4;
```

In $Procedure\_TD1$, $P_{leakage_i}$ considers the leakage power of core C_i for 2000 clock cycles and no power for other cores. It may be noted that for all the cores we have worked with, there is no variation in the thermal profile after 2000 clock cycles, due to only leakage power consumption. So, considering 2000 clock cycles is sufficient to get the effect of leakage power on temperature. $T_{leakage_i}$ is the maximum increase in initial temperature of C_i due to leakage power consumption. This value is stored in the i^{th} position of an array of size N, named $TD1$.

In $Procedure\_TD2$, T_{self_i} is the temperature increase of core C_i, when no other cores are being tested in parallel with C_i. $T_{parallel_{ij}}$ is the maximum increase in temperature of C_i, when it is tested in parallel with C_j. $T_{extra_{ij}}$ is the extra increase in temperature of C_i as an effect of parallel testing of C_j with C_i. For example, suppose cores C_1, C_3 and C_7 are active in parallel. So, the temperature of core C_1 can be calculated by adding the (1, 1), (1, 3) and (1, 7) elements of the matrix $TD2$.

In $Procedure\_TD3$, $P_{preschedule1_{ij}}$ considers core C_j as active and C_i as the idle core. $T_{preschedule1_{ij}}$ calculates the maximum increase in the initial temperature of C_i as an effect of earlier scheduling of C_j. In $Procedure\_TD4$, we allow C_i to cool down towards its normal initial temperature for 2000 clock cycles. $T_{preschedule2_{ij}}$ is the initial temperature of C_i after 2000 clock cycles. The cooling rate of C_i for the combination of C_i and C_j can be calculated as

$$\text{Cooling Rate}(CR_{ij}) = \frac{(T_{preschedule1_{ij}} - T_{preschedule2_{ij}})}{2000} \tag{4.3}$$

This is basically the difference between $(i,j)^{th}$ elements of $TD3$ and $TD4$ divided by 2000. We can calculate the actual reduction in the initial temperature of C_i, by multiplying CR_{ij} with the time gap (TG) between end time of C_j and start time of C_i. Net increase in initial temperature of C_i because of earlier scheduling of C_j can be calculated as

$$\text{Net Initial Increase}(NAI_{ij}) = T_{preschedule1_{ij}} - (CR_{ij} \times TG) \tag{4.4}$$

Step II:

In this step selection of one test rectangle per core has been performed using PSO. Each particle gives a set of rectangles, one for each core.

- **Particle Swarm Optimization**

The same particle structure and evolution procedure used for single-frequency power-aware testing (Section 3.6.3) has been used for thermal-aware test scheduling. Fitness of the particle has been evaluated by performing a scheduling of these rectangles.

Step III:

In this step fitness of each particle generated by PSO is calculated by scheduling them using a heuristic method. This fitness value (here TAT) of each particle is again fed back to PSO for further evolution of the particle to get better solution.

- **Scheduling of test rectangles**

The algorithm takes as input the rectangle set corresponding to the particle, the maximum TAM width W_{max}, the maximum power limit P_{max} and maximum allowable temperature T_{max}. It performs a scheduling of the rectangles, honouring the constraints that at no instant of time, the total TAM width requirement exceeds W_{max}, and the instantaneous power value does not exceed P_{max}. Also the maximum temperature of individual cores does not exceed T_{max}. The resulting total test time (TAT) is the fitness of the particle. At any point of time, the

algorithm maintains the following data structures to arrive at a decision about scheduling the next core.

Break_ Point_ List (BP):

A set of time instants at which the power requirement of the schedule has changed from its value in the previous instant. The next core can be scheduled at any of the breakpoints, $bp_k \in BP$. Similar data-structure has also been used for power-aware test schedule generation in Section 3.6.3.

Available_ TAM_ Width_ Info (ATW):

A set with cardinality same as BP. The value atw_k is equal to the total free TAM width available at breakpoint instant bp_k. Similar data-structure has also been used for power-aware test schedule generation in Section 3.6.3.

Power_ Tracker (PT):

This is also a set with cardinality same as BP and holds the total power consumed by the already scheduled cores at the corresponding breakpoint instant. Similar data-structure has also been used for power-aware test schedule generation in Section 3.6.3.

Temperature_ Tracker (TT):

It keeps track of the temperature of each core during scheduling.

A $Power\_Violation\_Checker(PVC)$ checks whether there is any power violation in the schedule. If at any particular point in the schedule, a core does not respect PVC, it does not get the permission to get scheduled at that point. Similarly a $Thermal\_Violation\_Checker(TVC)$ checks whether any core is violating thermal budget or not. TVC checks the scheduling position of a core and uses superposition principle to add different thermal values from $TD1$, $TD2$, $TD3$ and $TD4$ corresponding to that core to calculate the exact temperature of the core. Suppose a core starts its testing from the 0^{th} clock cycle, there will be no leakage power consumption of the core before it starts testing. Naturally, the increase in the initial temperature should not be considered for that particular core, while the same effect has to be considered for the cores which remain idle initially and consume leakage power. Similarly, when a new core gets scheduled, the temperature profiles of the already scheduled cores also get modified because of the newly scheduled core. TVC checks all these possible conditions and decides whether the new core can be scheduled at that breakpoint or not. The temperature calculation procedure of TVC has been mentioned in $Procedure\_TVC$ (Algorithm 4.5).

Algorithm 4.5: ***Procedure_ TVC***

```
1  begin
2      if (a new core C_i wants to schedule at any bp_k) then
3          Calculate temperature T_i of C_i like this;
4          if (idle time of C_i ≥ 1000 clock cycles) then
5              Add T_leakage_i from TD1 to T_i;
6              Check thermal violation C_i;
7              if (thermal violation) then
8                  Do not schedule C_i at bp_k;
9          Add T_self_i from TD2 to T_i;
10         Check thermal violation of C_i;
11         if (thermal violation) then
12             Do not schedule C_i at bp_k;
13         for each core C_j(1 ≤ j ≤ N)(j ≠ i) do
14             Check parallelism between C_i and C_j;
15             if (parallel) then
16                 Add T_extra_ij to T_i;
17                 Add T_extra_ji to T_j;
18                 Check thermal violation C_i and C_j;
19                 if (thermal violation) then
20                     Do not schedule C_i at bp_k;
21             else
22                 Calculate CR_ij;
23                 Find scheduling time gap (TG) between C_j and C_i;
24                 Net increase in initial temperature of C_i is
                   (T_preschedule1_ij − (CR_ij × TG));
25                 Add it to T_i;
26                 Check thermal violation of C_i;
27                 if (thermal violation) then
28                     Do not schedule C_i at bp_k;
29         if (no thermal violation) then
30             Schedule C_i at bp_k;
```

To make the schedule compact, we try to utilize any available TAM resource, power budget and temperature budget at every break-point. Hence, for the break-point bp_k, the algorithm scans the unscheduled rectangle list to check for the largest rectangle that can be scheduled at bp_k. If none are feasible, the algorithm advances to the next break-point. Power requirements and temperature validations are also checked to ensure satisfaction of power limit and temperature limit at every break-point till the end of schedule for the current core. When rectangles corresponding to all cores have been scheduled, the maximum end time of testing of all cores

gives the total test application time. The algorithm to produce the schedule has been presented in Algorithm 4.6.

Algorithm 4.6: ***Schedule_ Rectangles***

input : $LIST_C$, W_{max}, P_{max}, T_{max}
output: A schedule of all the cores respecting all the constraints with minimum TAT

```
1  begin
2      while (all cores are not scheduled) do
3          bp_k ← Break point with minimum time value;
4          atw_k ← Available tam resource at bp_k;
5          while (atw_k > 0) do
6              Set Area_max = 0;
7              forall the unscheduled cores C_i do
8                  if w_ij ≤ atw_k then
9                      if P_ij respects PVC then
10                         if C_i and all earlier scheduled cores respect TVC
                           then
11                             Calculate Area_ij = w_ij × T(w_ij);
12                     if Area_ij > Area_max then
13                         Set Area_max = Area_ij;
14                         Set C_imax = C_i;
15             Select C_imax for scheduling at bp_k;
16             Update BP;
17             Update ATW;
18             Update PT with P_ij;
19             Update TT with current temperature values of all scheduled
               cores;
20             Start_Ci = bp_k;
21             End_Ci = bp_k + T(w_ij);
22             Mark C_i as scheduled;
23         Remove bp_k and atw_k from BP and ATW respectively;
```

4.6. Experimental Results And Discussions

We have divided this section into two parts. Fewer details of the traditional ITC'02 SoC benchmarks [40] have motivated us to form new SoCs with sufficiently detailed information to observe exact thermal behaviour of them. The first sub-section describes the details of our newly formed SoCs, while the next sub-section presents the results of our experimentation on those SoCs.

4.6.1. Experimental Setup

To get exact thermal behaviour of cores in a chip, the input power profiles of the cores and the floorplan of the chip must be exact and specific. Thermal profile of a chip may vary by a large extent with the variation in the power profiles as well as the floorplan of the chip. The power profile of each core depends largely on its dynamic and leakage power consumptions as well as transitions in the scan cells during test pattern shifting at each clock cycle. Details of the test patterns are also very important to get the transition counts in the scan cells. Exact floorplan of a chip may not be obtained without the information regarding the areas of the cores. However, ITC'02 benchmarks [40] contain none of the above mentioned details of the SoCs. In the absence of all these information, most of the works reported in the literature assume a single fixed peak power [28, 53, 98, 18, 76, 38, 77, 29] throughout the test session of a core. Few other works although consider cycle power values [110, 109] of a core, they calculate each cycle power as the transition counts in the scan cells using random test patterns. As different cores have different dynamic power consumptions and also different test patterns show different values of transition counts, these assumptions may not produce exact power profiles of the cores. Moreover, the assumption of hand-crafted floorplans [84] of the chips may again show erroneous thermal behaviours of the cores in a chip as lateral temperature dependency between cores plays a major role in the creation of thermal profile. To overcome all these drawbacks regarding insufficient information of the SoCs, we have developed two SoCs named $k10$ and $k25$ having 10 and 25 cores respectively using ITC'99 [5], ISCAS'89 [27] and ISCAS'85 circuits.

From the benchmark suites mentioned above, a number of cores have been designed as follows.

1. Each Circuit description in *Verilog* format (.v) has been taken as input and mapped to Faraday 90nm standard cell library [6]. The circuits are synthesized using *Synopsys Design Vision* Compiler [119] to generate gate-level netlist from the *Verilog* design description.

2. All the flip-flops are replaced by scan flip-flops using *Synopsys DFT* Compiler for the testability purpose.

3. Multiple scan chains are inserted to reduce the test application time.

4. Dynamic and leakage powers are extracted from *Synopsys Design Vision Report Power* tool [119].

5. Area of each core is calculated using *Cadence Encounter* Compiler [17].

6. Exact floorplan of the SoCs are calculated using Floorplanning tool *Blobb* [19, 3] and *Plottool*. As different cores have different sizes, there are some empty spaces in the floorplan of the SoCs. These empty spaces are packed with zero power boxes to support temperature simulation using HotSpot tool. The packing of empty spaces with zero power boxes has ben performed using a C program.

7. For most of the circuits, test vectors are generated using Synopsys $TetraMax$ $ATPG$ [120]. For larger circuits like $b22$, $b17$ from ITC'99, *Atalanta*-M-2.0 [1] has been used to generate test vectors. Test pattern compression is not performed.

Figure 4.7 presents the total flow of the design set-up that we have followed to form two new SoCs.

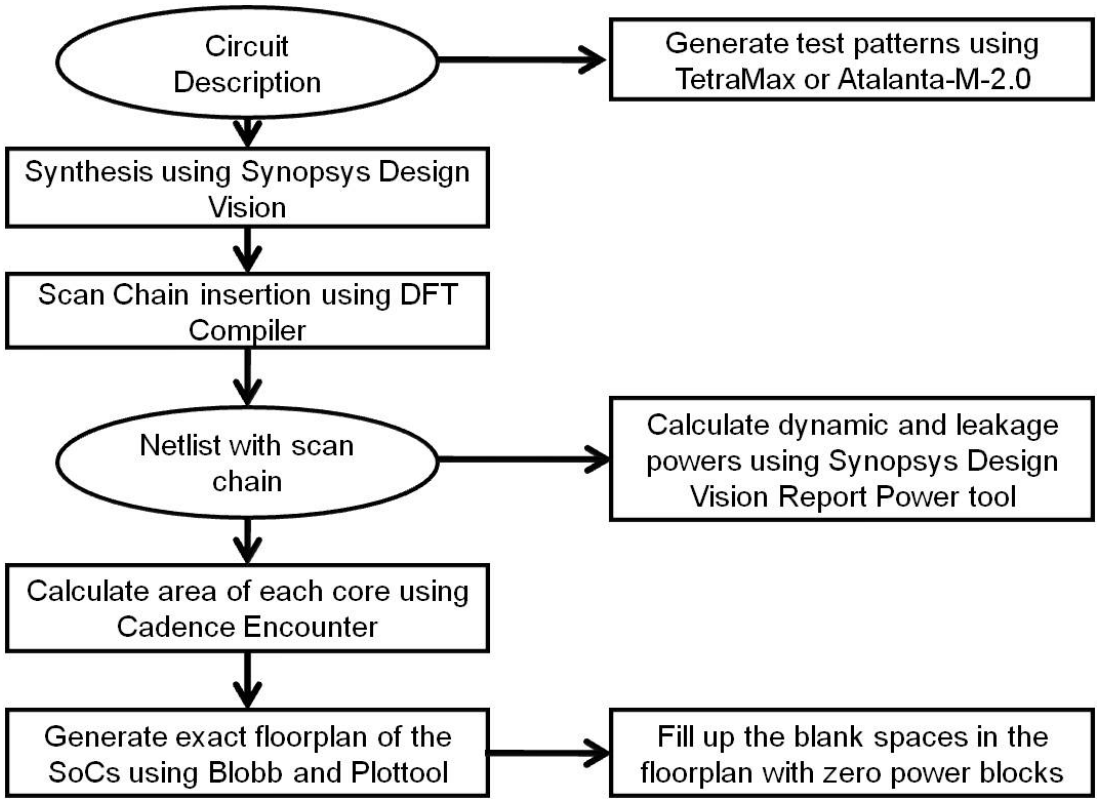
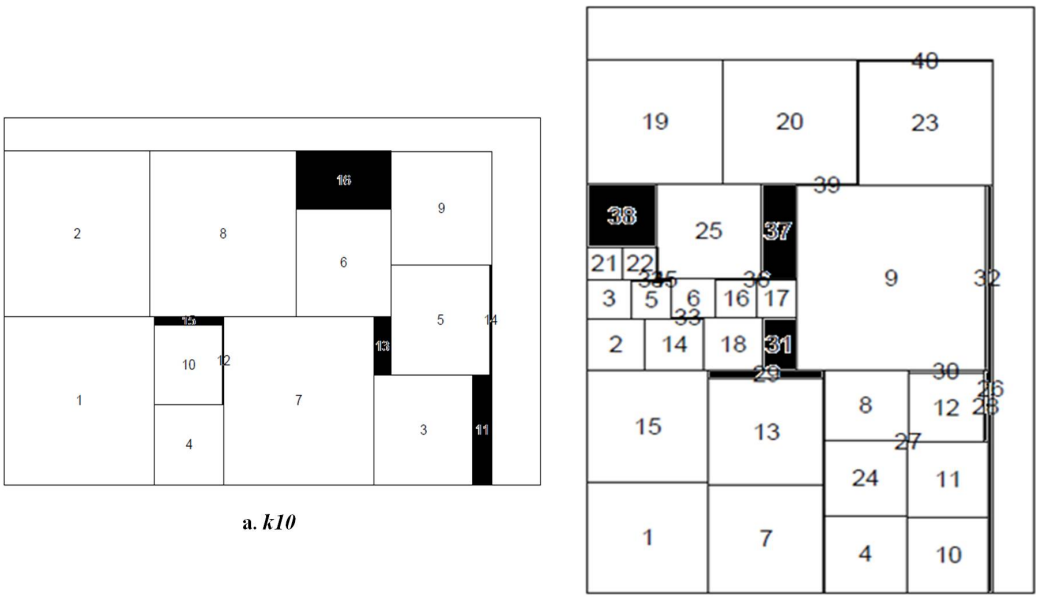


Figure 4.7: SoC design set-up flow

All data that we have obtained from different tools mentioned above, for different circuits and the components of two newly formed SoCs $k10$ and $k25$ are mentioned in Table 4.1. The exact floorplan of both the SoCs generated from *Blobb*[19, 3] and *Plottool* are shown in Figure 4.8. The black portions in the floorplan are the empty spaces which are properly packed using zero power consuming boxes.

Table 4.1: Detail Information Of The Cores Of Newly Formed SoCs

Circuit name	Dynamic power (DP) (mW)	Leakage power (LP) (uW)	Area (A)(um^2)	No of test vectors	No of scan chains	Max length of scan chain	No of copies in SoC	
							$k10$	$k25$
$s38584$	9.1529	315.5739	0.129	146	32	45	2	2
$s38417$	4.6092	311.4597	0.134	100	32	55	2	0
$s35932$	10.2743	282.4460	0.118	64	32	54	0	2
$s15850$	3.0535	128.6255	0.054	131	16	34	1	0
$s13207$	1.9165	116.4896	0.049	273	16	41	0	1
$s9234$	1.2388	71.0824	0.029	147	4	54	1	2
$s5378$	0.7293	35.0989	0.016	124	4	46	0	2
$b22$	4.3550	218.5332	0.093	1501	46	15	0	1
$b21$	2.5584	143.3848	0.061	711	16	31	1	2
$b17$	3.6796	346.3996	0.161	1477	46	31	0	3
$b15$	1.1875	120.6018	0.058	457	16	29	2	2
$b14$	1.3280	68.1279	0.029	737	4	62	1	1
$c7552$	3.5920	38.4599	0.015	219	0	0	0	3
$c5315$	2.2925	27.8943	0.011	123	0	0	0	2
$c1908$	0.9554	7.5269	0.003	119	0	0	0	1
$c499$	0.0040	1.1222	0.0006	52	0	0	0	1



a. *k10*

b. *k25*

Figure 4.8: Exact floorplan of $k10$ and $k25$ obtained using $Blobb$ and $Plottool$

4.6.2. Experimental Results

In this part, simulation results of our experimentation are shown with newly formed SoCs. Window-based peak power model and superposition principle based thermal model are used along with PSO guided test scheduling algorithm to obtain thermal-aware test schedule. The simulation results of our experimentation are shown with newly formed SoCs. However, direct comparison of our work with other related thermal-aware test scheduling approaches is not possible, as they assume their own power profiles and different floorplans. We have tried to show a

comparison of our thermal model with the thermal model presented in [137, 136]. For this purpose we have considered same SoC information (mentioned in experimental setup) and compared the efficiency of the two thermal models. Different test schedules have been generated considering the two thermal models. Post-schedule HotSpot thermal simulations of the schedules have been carried out to finally check whether there are any thermal violations in the schedule or not.

4.6.2.1. Thermal-Aware Test Scheduling

This part shows the results of our experimentation under resource, power and thermal constraints. Table 4.2 shows the results for two newly formed SoCs $k10$ and $k25$. In our case studies we have taken the population size of PSO as 2000 and 3000 particles for $k10$ and $k25$ respectively. Simulation terminates if there is no improvement in solution over 1000, 2000 generations respectively for $k10$ and $k25$. We have shown simulation results for $W_{max} = 64$ and 56 for both the SoCs and made variations in the P_{max} and T_{max} values. A graphical representation of the variation in test application time (TAT) for the SoCs with the variation in P_{max} and T_{max} has been presented in Figure 4.9. It may be observed that, if we increase the P_{max} and T_{max} values, TAT is reduced. Another important observation is, for a particular P_{max} value, if we increase T_{max} value, TAT gradually reduces. This happens as with the relaxation of power and thermal constraints, TAT can be minimized further. We can choose the P_{max} and T_{max} values, according to our requirements, without violating maximum tolerance limit of the SoC, to obtain the best solution. It may be noted that in some cases, increase in the T_{max} value does not really help to reduce the TAT value. It is because of the fact that the SoC has already reached its saturation TAT value in that temperature range. Variation in the temperature does not have any impact on TAT, in that temperature range.

4.6.2.2. Comparison Between Thermal Models

In this part, we focus on the comparison between our proposed thermal model and the thermal model proposed in [137, 136]. It may be noted that, here we are not comparing between the test scheduling strategies as the test scheduling constraints proposed in [137, 136] are different from ours. In [137, 136], testing of a module $M1$ has been mentioned as $T1$ with a specific test length $L1$. Information regarding the number of test lines required to test a module and the utilization of the limited test resources have not been mentioned in [137, 136]. Rather, it depends on a test compatibility graph to select test concurrency between cores. On the other hand, our proposed test scheduling strategy partitions and allocates the limited test resources (TAM lines) to the cores and obviously test length depends on the number of allocated test lines to a particular core. To check the quality of the two thermal models, we have tried to fit the thermal model proposed in [137, 136] in our

Table 4.2: Variation In Test Application Time (TAT) For Different SoCs With The Variation Of P_{max} And T_{max}

k10	P_{max} = 1.3 Watt		P_{max} = 1.5 Watt		P_{max} = 2 Watt		P_{max} = 5 Watt		$P_{max} = \infty$ $T_{max} = \infty$
	T_{max} (°C)	TAT (Clk Cycle)	T_{max} (°C)	TAT (Clk Cycle)	T_{max} (°C)	TAT (Clk Cycle)	T_{max} (°C)	TAT (Clk Cycle)	TAT (Clk Cycle)
W_{max} = 64	72	43154	72	42494	72	42365	72	42365	34021
	80	41424	80	41388	80	41347	80	40924	
	85	39349	85	37875	85	37875	85	37875	
	90	39349	90	37875	90	37875	90	37875	
	95	39349	95	36984	95	36110	95	35256	
	100	39349	100	35880	100	34810	100	34459	
	105	39349	105	35865	105	34810	105	34072	
	110	39349	110	35732	110	34268	110	34021	
W_{max} = 56	72	50285	72	50285	72	50285	72	49010	38933
	80	43530	80	42424	80	42424	80	42424	
	85	43530	85	39672	85	39672	85	39672	
	90	43530	90	39672	90	39672	90	39672	
	95	42839	95	39672	95	39672	95	39672	
	100	42300	100	39672	100	39672	100	39672	
	105	42300	105	39672	105	39672	105	38936	
	110	42300	110	39672	110	38936	110	38936	

k25	P_{max} = 2.7 Watt		P_{max} = 3 Watt		P_{max} = 3.5 Watt		P_{max} = 4.5 Watt		$P_{max} = \infty$ $T_{max} = \infty$
	T_{max} (°C)	TAT (Clk Cycle)	T_{max} (°C)	TAT (Clk Cycle)	T_{max} (°C)	TAT (Clk Cycle)	T_{max} (°C)	TAT (Clk Cycle)	TAT (Clk Cycle)
W_{max} = 64	82	171142	82	169876	82	169869	82	169684	167971
	90	170578	90	169562	90	169146	90	169101	
	95	169740	95	169101	95	169101	95	168908	
	100	169513	100	169101	100	168982	100	168778	
	105	169090	105	168982	105	168904	105	168698	
	110	168698	110	168602	110	168602	110	168602	
W_{max} = 56	82	196567	82	196567	82	193613	82	192163	190231
	90	195710	90	193202	90	192494	90	192050	
	95	193074	95	192400	95	192163	95	191276	
	100	192494	100	191894	100	191276	100	191276	
	105	191814	105	191814	105	191276	105	191276	
	110	191814	110	191644	110	191276	110	191276	

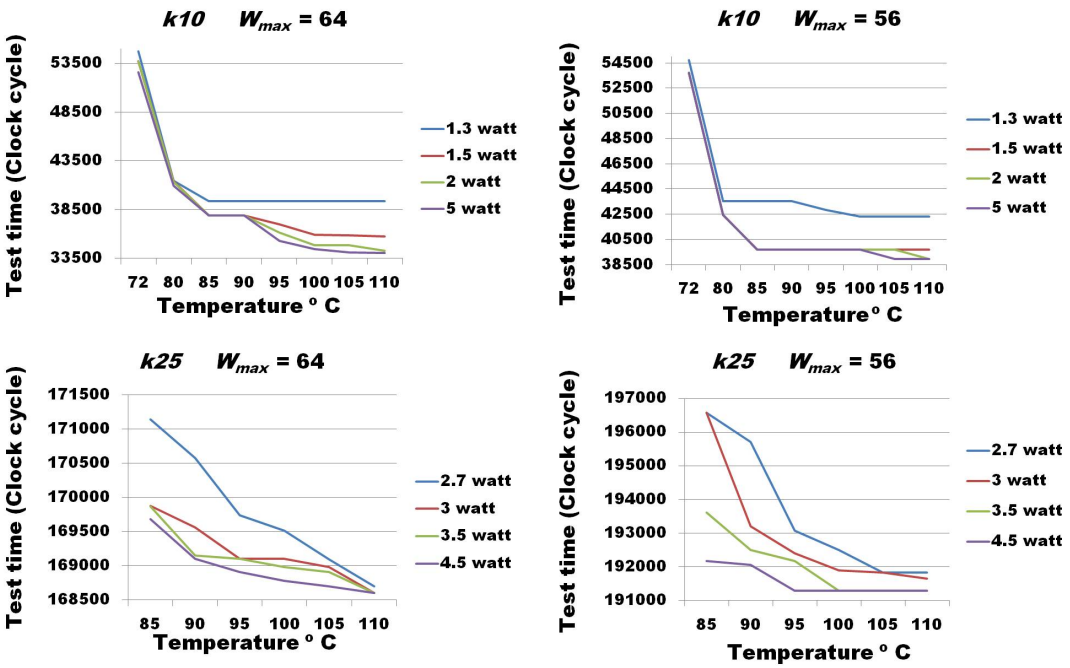


Figure 4.9: Graphical representation of the variation in test application time (TAT) for different SoCs with the variation of P_{max} and T_{max}

TAM constrained test scheduling strategy and then compare with our proposed thermal model. Different test schedules have been generated considering both the

thermal models. Post-schedule Hotspot thermal simulations of the schedules have been carried out to finally check whether there are any thermal violations in the schedules or not.

Tables 4.3 and 4.4 present the minimum allowable temperature values to get a valid test schedule for both the thermal models and their corresponding maximum temperature values obtained from post-schedule Hotspot simulations. Corresponding TAT of each schedule has been reported in this table, however, it is not a comparison between the TATs, as in both of the cases, the test scheduling strategies are the same (i.e. our proposed TAM constrained test scheduling strategy), and the only difference is in the thermal models. Naturally, it is hard to find any significant difference in TAT between the two results. Our main objective in comparison between the two thermal models is to find the minimum temperature budget at which the thermal models can produce test schedules and then checking the thermal validation of the obtained schedules using Hotspot thermal simulations. In that process, we have found that, using thermal model proposed in [137, 136] we are able to get test schedules at 70°C and 78°C for the SoCs $k10$ and $k25$ respectively. However, while using our proposed thermal model, we can get test schedules at minimum temperature levels of 72°C and 82°C for the SoCs $k10$ and $k25$ respectively. Our thermal model is unable to produce test schedules for any thermal budget lower than that. It may be noted that, although the thermal model presented in [137, 136] produces valid test schedules with a lower temperature limit than our thermal model, it does not always ensure thermal safe test schedule. It violates thermal limits in most of the cases. On the other hand, our thermal model violates the thermal limit for none of the cases. This is because our proposed thermal model takes care of the prescheduled heating effects of the cores more efficiently by considering both the temperature increase due to leakage power and doing proper estimation of heating and cooling effects of the already scheduled cores with more accuracy. This shows the efficiency of our thermal model. Figure 4.10 shows the thermal profile of the hottest core (core 2) of $k10$, after its scheduling in the final schedule using our thermal model. It may be noted that maximum temperature of the core is well below the maximum allowable temperature of the chip, while the thermal model presented in [137, 136] violates the specified maximum tolerable temperature limit (Figure 4.1).

Final thermal simulations show that, it is possible to get valid thermal-aware test schedule at 72°C and 82°C for the SoCs $k10$ and $k25$ respectively. Hence, the test schedules generated using the thermal model [137, 136] at these temperature limits, do not show thermal violations. Our tool can produce schedules only with a higher limit. For example, when fed with 70°C as T_{max} value, our tool does not produce any schedule for $k10$. It could work successfully only with a limit

Table 4.3: Comparison Of Validation Of Test Schedule Of Our Thermal Model With The Thermal Model Presented In [137, 136]

SoC name	W_{max}	Thermal Model[137, 136]				Our Thermal Model			
		P_{max} (Watt)	T_{max} (°C)	TAT (Clock Cycle)	Max temp (°C)	Thermal Violation	P_{max} (Watt)	T_{max} (°C)	TAT (Clock Cycle)	Max temp (°C)	Thermal Violation
$k10$	24	1.3	70	93935	71.11	**Yes**	1.3	72	99428	70.92	No
		1.5	70	93935	71.12	**Yes**	1.5	72	99428	70.92	No
		2	70	92835	71.05	**Yes**	2	72	97752	71.54	No
		5	70	90468	71.17	**Yes**	5	72	96925	70.92	No
	32	1.3	70	70161	70.74	**Yes**	1.3	72	73571	70.74	No
		1.5	70	69727	71.56	**Yes**	1.5	72	72238	70.74	No
		2	70	68870	70.74	**Yes**	2	72	72109	70.74	No
		5	70	68126	71.3	**Yes**	5	72	71423	71.81	No
	40	1.3	70	58435	70.66	**Yes**	1.3	72	61804	70.66	No
		1.5	70	57858	70.66	**Yes**	1.5	72	58270	70.66	No
		2	70	57539	70.66	**Yes**	2	72	57355	70.66	No
		5	70	56053	71.42	**Yes**	5	72	57355	71.77	No
	48	1.3	70	49802	70.79	**Yes**	1.3	72	54666	70.66	No
		1.5	70	49802	70.8	**Yes**	1.5	72	54534	70.66	No
		2	70	49802	70.86	**Yes**	2	72	54516	70.66	No
		5	70	49802	70.86	**Yes**	5	72	52288	70.66	No
	56	1.3	70	49802	70.74	**Yes**	1.3	72	50285	70.91	No
		1.5	70	46475	70.91	**Yes**	1.5	72	50285	70.91	No
		2	70	44698	70.93	**Yes**	2	72	50285	70.91	No
		5	70	44591	70.91	**Yes**	5	72	49010	70.66	No
	64	1.3	70	44043	71.2	**Yes**	1.3	72	43154	70.66	No
		1.5	70	43530	70.91	**Yes**	1.5	72	42494	70.91	No
		2	70	44043	71.32	**Yes**	2	72	42365	71.1	No
		5	70	43018	70.91	**Yes**	5	72	42365	70.66	No

Table 4.4: Comparison Of Validation Of Test Schedule Of Our Thermal Model With The Thermal Model Presented In [137, 136] (Contd.)

SoC name	W_{max}	Thermal Model[137, 136]					Our Thermal Model				
		P_{max} (Watt)	T_{max} (°C)	TAT (Clock Cycle)	Max temp (°C)	Thermal Violation	P_{max} (Watt)	T_{max} (°C)	TAT (Clock Cycle)	Max temp (°C)	Thermal Violation
$k25$	24	2.7	78	453745	77.77	No	2.7	82	453745	78.59	No
		3	78	453745	77.41	No	3	82	447883	78.13	No
		3.5	78	448857	78.73	**Yes**	3.5	82	447883	78.27	No
		4.5	78	443495	79.05	**Yes**	4.5	82	445785	79.49	No
	32	2.7	78	333338	77.94	No	2.7	82	338745	78.27	No
		3	78	334027	79.05	**Yes**	3	82	337809	78.13	No
		3.5	78	341010	79.88	**Yes**	3.5	82	337243	79.2	No
		4.5	78	334340	78.09	**Yes**	4.5	82	333807	77.5	No
	40	2.7	78	270111	77.36	No	2.7	82	272670	78.13	No
		3	78	268709	78.54	**Yes**	3	82	271907	79.04	No
		3.5	78	268794	78.95	**Yes**	3.5	82	270419	78.29	No
		4.5	78	237493	78.2	**Yes**	4.5	82	269664	79.25	No
	48	2.7	78	227611	78.14	**Yes**	2.7	82	227611	78.72	No
		3	78	227611	78.14	**Yes**	3	82	227611	77.15	No
		3.5	78	227651	77.8	No	3.5	82	227611	77.52	No
		4.5	78	227590	77.84	No	4.5	82	227539	77.77	No
	56	2.7	78	194950	76.81	No	2.7	82	196567	77.42	No
		3	78	195439	76.46	No	3	82	196567	77.42	No
		3.5	78	194464	77.87	No	3.5	82	193613	78.59	No
		4.5	78	193003	78.31	**Yes**	4.5	82	192163	79.13	No
	64	2.7	78	169718	76.77	No	2.7	82	171142	77.12	No
		3	78	171748	76.42	No	3	82	169876	77.94	No
		3.5	78	169418	76.72	No	3.5	82	169869	77.34	No
		4.5	78	169877	76.95	No	4.5	82	169684	77.18	No

of 72°C. The actual thermal simulations show this limit to be a bit pessimistic as the Hotspot results range between 70.66°C and 71.92°C. This, we believe, has occurred due to the simplified time overlapping model that has been used in our model. In our model, if there is a small overlap in the scheduled test times of two cores, it has been taken as a complete overlap of their test durations, which is probably not accurate for most of the cases. However, this pessimistic assumption could result in thermally valid test schedules.

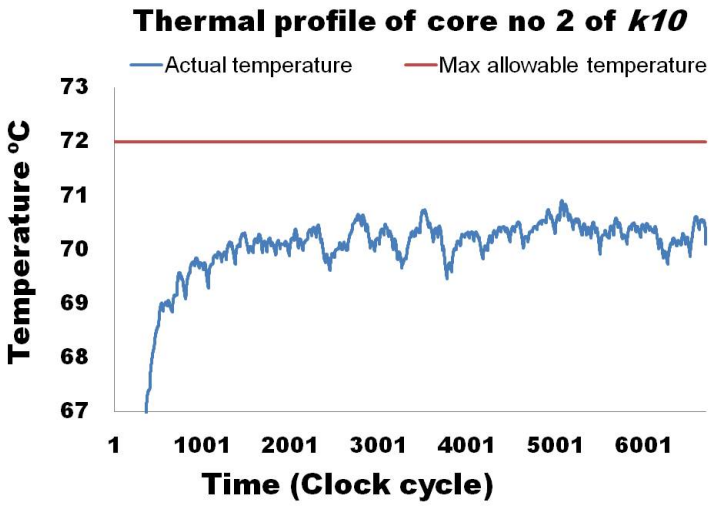


Figure 4.10: Temperature profile and maximum temperature limit of core 2 of $k10$ using our thermal model

4.7. Conclusion

In this chapter a new superposition principle based fast thermal model, capable of estimating temperature with accuracy, has been presented. This thermal model along with the window-based peak power model proposed in Chapter 3 have been incorporated into a PSO based formulation to select test rectangles for cores and scheduling them using a 3-D bin-packing approach. A new set of SoC benchmarks with detailed information has been used to get exact thermal effects of the cores at the time of generation of thermal-aware test schedule using the proposed method.

Chapter 5
Thermal-Aware Test Data Compression Using Dictionary Based Coding

5.1. Introduction

With increasing complexity of present day integrated circuits (ICs), the amount of test data needed to test the ICs has increased dramatically. More test patterns are required to improve fault coverage targeting delay faults, stuck-at-faults and some other subtle faults. Automatic Test Equipment (ATE) has to accommodate a large amount of test data, increasing memory size as well as memory cost. Increasing test data volume also results in longer test application time (TAT), which significantly increases total test cost. Built-in-self-test (BIST) is an alternative solution which avoids the usage of external storage to store test data. However, it suffers from random-resistant faults and bus contention during test application leading to inadequate fault coverage [81]. The other alternative is test data compression, which stores huge amount of test data (T_D) in ATE in a compressed form (T_E), helps to reduce the total memory requirement. The compressed data (T_E) has to pass through a decompressor to get back original uncompressed (T_D) test patterns before being applied to the circuit under test (CUT).

It is worth mentioning that, more than 95% bits of the test data are don't-cares [16]. To get better compression, most of the test compression techniques take the advantage of don't-care bits, which can be flipped either to '0' or '1' to get more number of matching sub-vectors from the test vectors without compromising fault coverage.

Another major concern during testing is the temperature of the IC. Several works of literature have tried to minimize the temperature of the IC during testing by efficiently filling the don't-care bits present in test patterns [31, 142]. 0-fill, 1-fill, minimum-transition-fill and random-fill are some of the popular don't-care bit

filling techniques. Other thermal-aware don't-care bit filling works [31] attempt to reduce temperature using their own cost function to fill up the don't-care bits.

It is interesting to note that, both test data compression and thermal-aware don't-care filling concentrate on filling the don't-care bits efficiently to get better compression and lower temperature during test, respectively. In the literature, these two problems have been solved separately. Test data compression works try to fill up the don't-care bits to get better compression ratio ignoring the thermal effect, while thermal-aware don't-care filling works try to minimize temperature by efficient don't-care bit filling without taking care of test data compression. It may so happen that, for a particular don't-care bit, better compression can be achieved if it is filled with a '0' value, while a '1' value for that bit may produce lower temperature during testing. A trade-off between these two don't-care bit filling techniques is required to obtain a good compression ratio with reasonably low temperature.

In recent times, some works have addressed the problem of high power consumption in test data compression schemes [26, 85]. However, it may be noted that, power minimization may reduce overall temperature of a circuit, but, does not necessarily minimize peak temperature, which causes local hotspot due to non-uniform spatial power distribution in the circuit and hence permanent damage of it. The peak temperature of a block depends not only on power consumption, but also on heat exchange between adjacent blocks. So, the temperature of a circuit requires special attention.

In this chapter, we have first shown the existence of variation in peak temperature of a circuit for a different selection of don't-care bit filling, targeted towards compression. A trade-off between temperature and compression ratio, which can fill up the don't-care bits smartly taking care of compression and temperature simultaneously, has also been presented. This technique efficiently bridges the gap between test data compression and thermal-aware don't-care filling techniques. Finally we have proposed a thermal-aware test data compression technique that reduces the peak temperature of the circuit without compromising on the compression ratio. To the best of our knowledge, this is the first work, which addresses temperature reduction in test data compression schemes.

5.2. Background Of Dictionary Based Test Data Compression

We have used a dictionary-based coding technique for compression [81]. It is a popular test data compression technique in which the total test set is partitioned into several smaller equal-sized test slices. Suppose n test patterns, each of length

L, are required to test a circuit. Total test data can be calculated as $T_D = n \times L$. For a circuit with m number of scan chains, m bits of test data have to be transferred simultaneously from ATE to the circuit. Each such m bit data is called a test slice. If the length of a scan chain is l, total number of the slices formed is $n \times l$. The scan cells are divided into scan chains in a manner to keep the scan chains as balanced as possible. However, some don't-care bits may be required to pad the shorter length test slices to make all of them equal sized. Figure 5.1 shows an example of test data transfer in multiple scan chains and corresponding test slices.

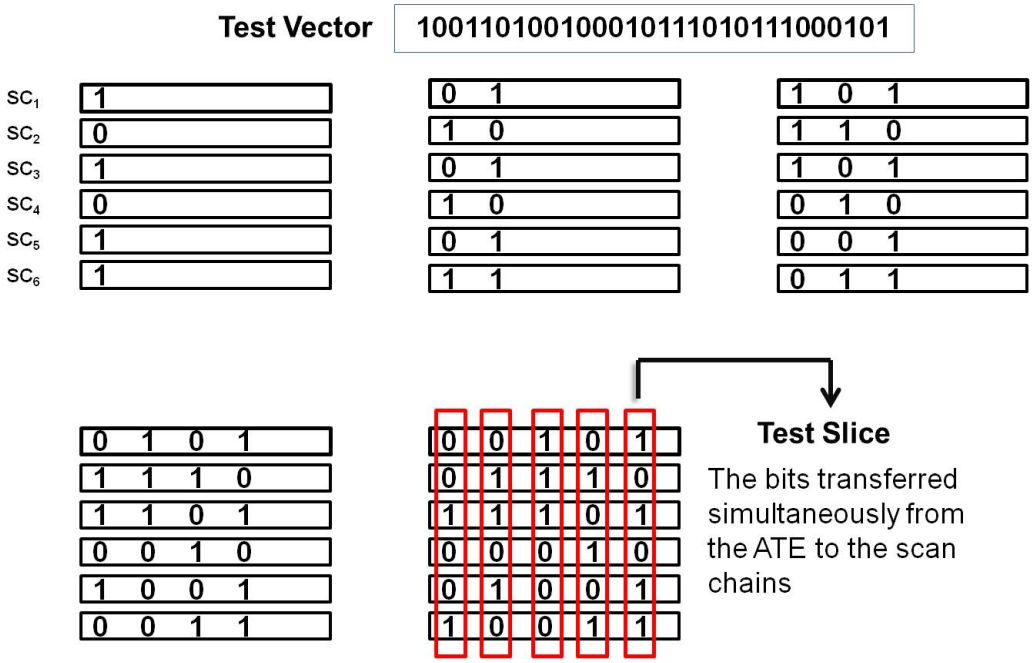


Figure 5.1: An example of test slices

Two slices A and B can be treated as compatible if for any bit position i, either A_i and B_i ($1 \leq i \leq m, m$ is the size of the slice) are equal, or at least one of them is a don't-care. A representative slice of the compatible slices may be stored in a dictionary instead of storing all of them. This helps to reduce the memory size required to store the test data. However, it may be noted that, dictionary size is kept relatively small to reduce hardware overhead. For a dictionary of size D, the representative slices from D largest compatible set of slices are stored in the dictionary. The rest of the slices are stored uncompressed in the memory. In dictionary-based coding, a single bit prefix is used to identify whether a slice belongs to the dictionary or not. The slices, whose representatives are stored in the dictionary, are denoted by a code of length $\lceil log_2|D| \rceil + 1$, while the code length for the rest of the slices is $m + 1$. Selection of D largest compatible set of slices is carried out via a clique partitioning heuristic [81]. The slices are represented by an undirected graph $G = (V, E)$, where each slice represents a vertex and the

compatibility between two slices is denoted by an edge between them. The clique partitioning heuristic tries to find D largest possible cliques from the graph. The representative slices of these cliques are stored in the dictionary. A decompressor is used before CUT to get back the original patterns from the compressed codewords. Figure 5.2 shows an architecture of dictionary coding based test data compression scheme.

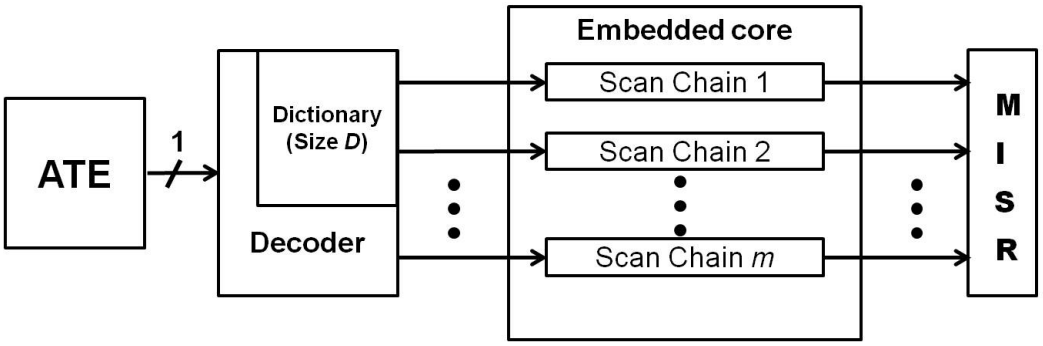


Figure 5.2: Architecture of dictionary coding based test data compression scheme[81]

5.3. Motivation

Any clique partitioning technique uses the flexibility of don't-care bits to get the maximum number of matching slices. It may be noted that, most of the don't-care bits present in original patterns get filled up (either '0' or '1') at the time of generation of final patterns after clique partition, where all the compatible slices belonging to a single clique are replaced by the representative slice corresponding to that clique. For example, both of these two compatible slices (10XX01X1) and (X010X1XX) belonging to the same clique, are replaced by the representative slice (101001X1) in the final patterns. An observation from our experimentation is that around 72% of the total don't-care bits present in original test patterns get filled-up (either '0' or '1') in the clique partitioning process. However, as the clique partitioning is NP-Hard [81], different heuristics generate different clique sets. In that case, a slice may not belong to the same clique for different heuristics. The don't-care bits in the final test patterns also change as the don't-care bits present in a slice may fill up differently if the slice belongs to different cliques for different heuristics. It has a direct impact on the temperature profile of the circuit.

The temperature of a circuit largely depends on the dynamic power consumption, which can be estimated from the transition counts in the scan cells during shifting of test patterns through it. Different test pattern sets generate different thermal profiles of the circuit. As temperature is a major concern during testing, it is desirable to generate a test pattern set which produces a low temperature profile of the circuit. Test designer has the flexibility to efficiently fill up the

don't-care bits present in the original test patterns, to minimize the temperature of the circuit. Thermal-aware don't-care filling techniques fill up the don't-care bits efficiently to minimize temperature during testing, but may fail to produce a large number of compatible slices. This results in poor compression and hence increase the memory size and test cost. It is a challenging issue to fill up the don't-care bits efficiently to produce low temperature during testing, as well as a reasonably good compression ratio. Although, lots of research has been carried out to address test data compression and thermal-aware don't-care filling separately, none of the previous works have merged the problems into a single one and solved it. This has motivated us to develop a thermal-aware test data compression technique that brings balance between compression ratio and temperature of the IC.

5.4. Variation Of Temperature And Compression Ratio For Different Clique Partitions

The clique partitioning heuristic presented in [81] starts having the vertex with maximum number of neighbours, retrieves the largest clique associated with it. The process continues with searching for the next vertex with maximum neighbours from the remaining members of the graph, until all the vertices have been retrieved.

It may be noted that instead of starting at a vertex with the highest neighbours, if we start with every vertex and then explore the total graph to find the best possible clique sets from the graph, we get $n \times l$ number of clique sets for $n \times l$ number of vertices. To generate different clique sets from the graph $G = (V, E)$, we have modified the clique partitioning heuristic. Our modified heuristic uses following data-structures.

Clique_ List (CL):

$Clique\_List\ CL_i$ $(1 \leq i \leq (n \times l))$ is a database storing information regarding all generated cliques from the graph considering i as $start\_vertex$.

Representative Slice:

Some of the don't-care bits of the slices belonging to the same clique are filled up (either '0' or '1') to form a common representative slice for all of them.

Slice_ Info (SI):

All the slices from a particular clique are replaced by the corresponding representative slice of that clique. The modified details of all of the slices

are stored in $Slice\_Info\ SI_i$ $(1 \leq i \leq (n \times l))$. These $Slice\_Info$ are used to generate modified test patterns (TP_i) $(1 \leq i \leq (n \times l))$ corresponding to CL_i.

Algorithm 5.1: ***Modified_ Clique_ Partition_ Heuristic***

```
1  begin
2      Calculate cardinality of each vertex;
3      for i ← 1 to no_of_vertices do
4          current_start_vertex = i;
5          while (all vertices are not selected) do
6              Mark all vertices connected to current_start_vertex as visited;
7              Mark all vertices connected to visited vertices as next_to_visit;
8              Select largest possible clique that can be formed including
               current_start_vertex;
9              Mark all the vertices of the selected clique as selected;
10             Update Clique_List(CL_i) with the newly formed clique;
11             Select the vertex with maximum cardinality as the
               current_start_vertex from the vertices which are marked as
               either visited or next_to_visit, but not a member of any
               clique;Go to line 5;
12             if (no visited or next_to_visit vertex is left to be selected)
               then
13                 Select the vertex with maximum cardinality as the
                   current_start_vertex from rest of the vertices;
14         Form Slice_Info(SI_i) using the information of Clique_List (CL_i);
```

Algorithm 5.1 presents the $Modified\_clique\_partition$ heuristic. It first calculates the cardinality of each vertex. The heuristic starts with every vertex i and tries to find the maximum clique that can be formed including i. Then it progresses finding the next largest clique covering one of the *visited* or *next_to_visit* vertex until all the vertices are included in the corresponding clique list CL_i $(1 \leq i \leq (n \times l))$. CL_i stores the clique information that begins with i as *current_start_vertex* and SI_i stores corresponding modified details of the slices.

Figure 5.3 shows an example set of test data distribution in 8 scan chains and corresponding compatibility graph of the test data. There is a total of 18 slices, each of which has 8 bits. For this particular example, we can have at most 18 different clique sets. Figure 5.4 shows how different start nodes can produce different clique-sets of the same compatibility graph. It may be noted that the highest cardinality vertex is node 6. So, according to clique partitioning heuristic presented in [81], if we start with vertex 6 then the cliques of the corresponding clique list CL are {5,4,6,8}, {11,12,13,14}, {1,2}, {3,7}, {15,16}, {9}, {10},{17}

Slice index	Scan chain index							
	1	2	3	4	5	6	7	8
1	1	0	X	1	X	X	0	1
2	1	0	0	X	0	0	X	1
3	1	X	X	0	X	0	X	1
4	X	0	1	1	0	X	X	X
5	1	0	X	1	X	X	X	X
6	1	X	X	X	X	0	1	X
7	1	0	X	0	0	X	X	1
8	1	0	1	X	0	X	1	0
9	0	1	X	X	0	0	0	X
10	X	1	0	X	1	1	X	X
11	X	0	X	0	0	1	X	0
12	0	0	X	0	X	1	X	0
13	0	X	1	0	X	1	X	0
14	0	X	1	0	0	1	X	X
15	0	1	0	1	X	0	1	X
16	X	1	X	1	X	0	1	X
17	0	1	0	X	X	X	1	X
18	X	1	X	1	X	0	1	0

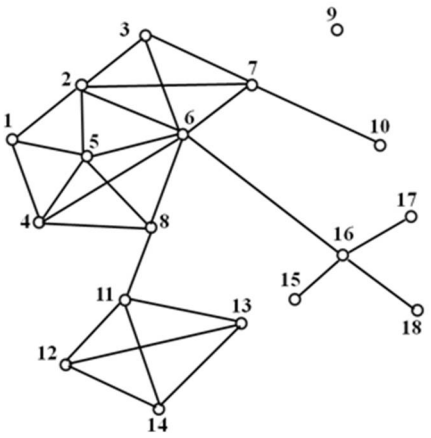


Figure 5.3: An example of test data for multiple scan chain set-up and corresponding compatibility graph.

and {18}. Instead of starting with vertex 6, if we start with vertex 1 then the cliques of the corresponding clique list CL_1 are {1, 2, 5}, {3, 6, 7}, {11, 12, 13, 14}, {4, 8}, {15, 16}, {10}, {17}, {18}, {9}, while the cliques of clique list CL_2 are {2, 3, 6, 7}, {1, 4, 5}, {11, 12, 13, 14}, {15, 16} {8}, {10}, {17}, {18}, {9}. So, starting with different start nodes, clique partitioning can generate different clique sets. It may be noted that, a particular vertex may belong to different cliques for different clique sets. Since for different clique sets, representative slices are also different, the don't-care bits present in the test slice corresponding to that vertex may also get filled up differently. For example, vertex 2 belongs to two different cliques for CL_1 and CL_2. The don't-care bits present in slice 2 also get fill up differently. Original slice detail of slice 2 is (100X00X1), which is modified as (10010001) in SI_1, while the same slice is modified as (10000011) in SI_2. Final test pattern sets TP_1 and TP_2 also differ from each other. These two different test pattern sets may show different thermal behaviour.

To check the variation of the thermal profile for different clique selections, we have to do thermal simulation of the circuits. Thermal profile of a circuit has been calculated using the thermal simulation steps mentioned in the following.

5.4.1. Thermal Simulation Steps

- Each Circuit description in Verilog format (.v) has been taken as input and mapped to Faraday 90nm standard cell library [6]. The circuits have been synthesized using Synopsys Design Vision Compiler [119] to generate a gate-level netlist from the Verilog design description.

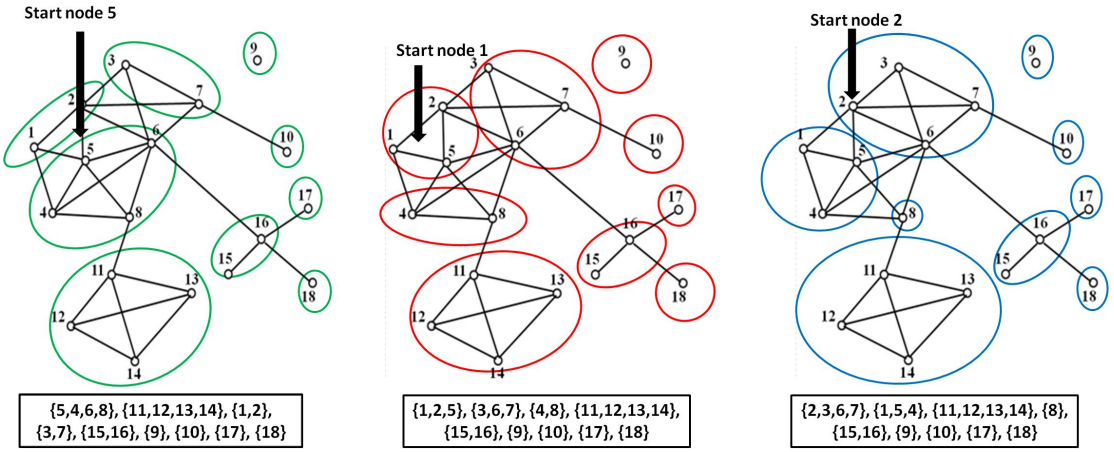


Figure 5.4: Different clique-sets for different start nodes

- All the flip-flops have been replaced by scan flip-flops using Synopsys DFT Compiler for the testability purpose. Multiple scan chains have also been inserted.
- Encounter RTL-to-GDSII system from Cadence [17] has been used to generate floorplan of the given scan inserted design using standard cell library.
- The scan inserted netlist and the test patterns have been fed to a power estimator tool [31]. Power consumption of each of the logic elements (gate, flip-flop) with different types and inputs has been estimated using Synopsys Design Vision tool and stored in a database. This database has been used to convert the transitions in individual logic elements into their corresponding power values during circuit simulation for each scan shift and capture operation.
- The floorplan obtained from Cadence Encounter tool [17] has been divided into a number of blocks of same size for thermal simulation. The power trace has been computed for each of these blocks.
- Thermal simulations have been carried out using thermal simulation tool HotSpot [118] taking power trace and block-level floorplan as input.

It may be noted that the power estimator [31] works for single scan chain operation only. This has been augmented for multiple scan chain operation. Figure 5.5 presents the entire flow of the thermal simulation steps that we have followed to calculate thermal profile of the circuit.

5.5. Observation

We have seen a variation in the thermal profile for different clique selections for several ISCAS'89 and ITC'99 benchmarks. Circuit information regarding the test

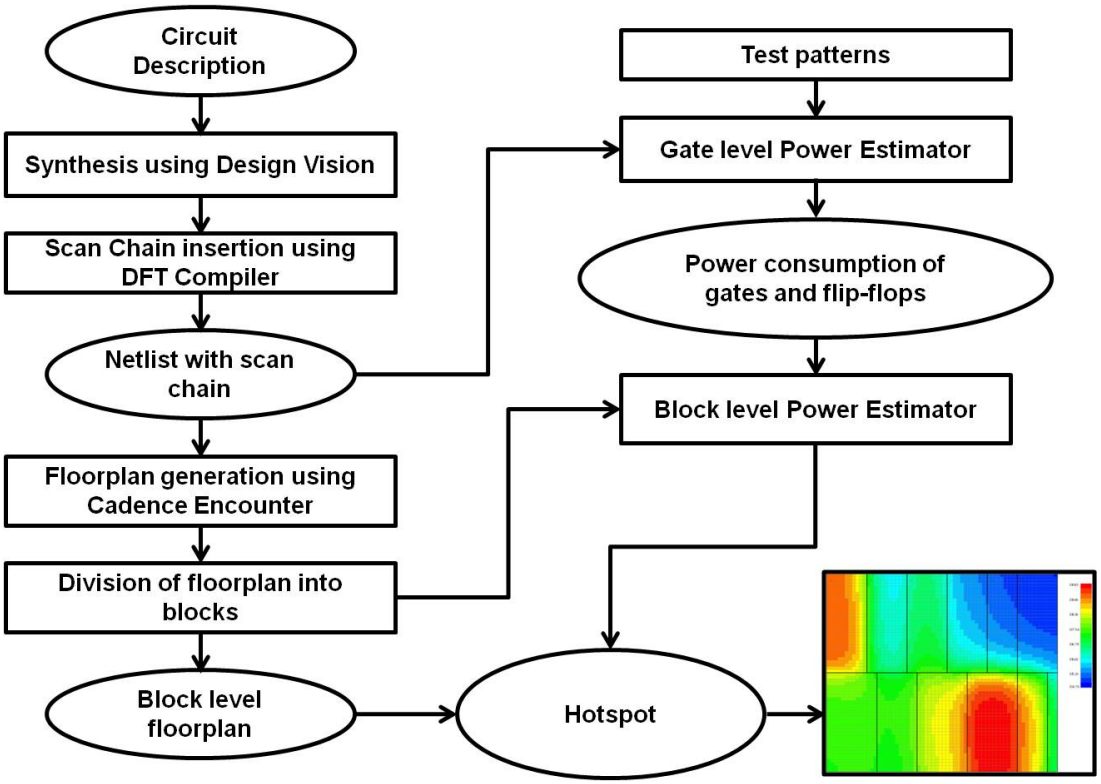


Figure 5.5: Flow of thermal simulation

data volume and the number of clique sets generated for different number of scan chains (N_{SC}) have been mentioned in Table 5.1. Mintest [43] test sets for stuck-at fault are used for ISCAS'89 benchmarks, while test sets for stuck-at fault, generated from TetraMax [120] ATPG tool are used for ITC'99 benchmarks. Table 5.2 shows the variation of peak temperature and compression ratio (CR) for different clique selections. Graphical representations of the variation of temperature and compression ratio for different clique partitions are also presented in Figures 5.6 and 5.7 respectively.

From Table 5.2, a significant variation in peak temperature can be noticed for different clique selections while the corresponding variation of CR is reasonably very less. This shows the scope of obtaining good test data compression with low temperature profile, by efficient don't-care filling.

5.6. A Trade-off Between Temperature And Compression Ratio

In the previous section, we have seen the variation in peak temperature and compression ratio with different clique selections. Although, selecting a solution with lower peak temperature and good compression ratio from all clique lists may be a choice to address both low temperature and good compression issues, this method relies on detailed enumeration to find clique set with low temperature and hence

Table 5.1: Clique Set Information For Different Scan Chains

Circuit name	Test vector length	No of test vectors	T_D	Number of clique sets			
				$N_{SC} = 32$	$N_{SC} = 48$	$N_{SC} = 64$	$N_{SC} = 128$
$s13207$	700	236	165200	5192	3540	2596	1416
$s15850$	611	126	76986	2520	1638	1260	630
$s38417$	1664	99	164736	5148	3465	2574	1287
$s38584$	1464	136	199104	6256	4126	3128	1632
$b14$	277	762	211074	6858	4572	3810	2286
$b15$	485	457	222102	7312	5027	3656	1828

Table 5.2: Variation Of Temperature And Compression Ratio For Different Clique Selections (For D =128)

Circuit name	$N_{SC} = 32$				$N_{SC} = 48$				$N_{SC} = 64$				$N_{SC} = 128$			
	Temp(K)		CR		Temp(K)		CR		Temp(K)		CR		Temp(K)		CR	
	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min
$s13207$	352.1	343.2	73.4	73.2	363.2	353.2	81.3	81.0	382.3	363.1	85.2	84.8	418.3	387.4	91.9	91.4
$s15850$	349.8	338.7	67.9	67.0	366.5	351.2	73.8	72.7	376.7	363.4	76.8	76.0	394.2	379.2	82.3	81.1
$s38417$	351.2	343.5	45.5	43.7	359.2	353.4	54.2	51.8	374.2	368.4	43.1	37.1	416.8	403.2	34.0	32.2
$s38584$	353.0	338.4	63.0	62.3	371.2	351.4	68.0	67.0	384.2	375.8	68.3	67.3	412.1	400.2	70.6	68.9
$b14$	365.2	357.2	56.8	55.0	367.4	361.2	47.4	45.3	385.2	373.2	47.5	45.8	412.7	400.7	52.1	51.2
$b15$	358.7	334.9	73.3	73.0	374.6	345.8	79.1	78.4	380.0	357.2	81.2	80.2	398.5	359	75.4	73.4

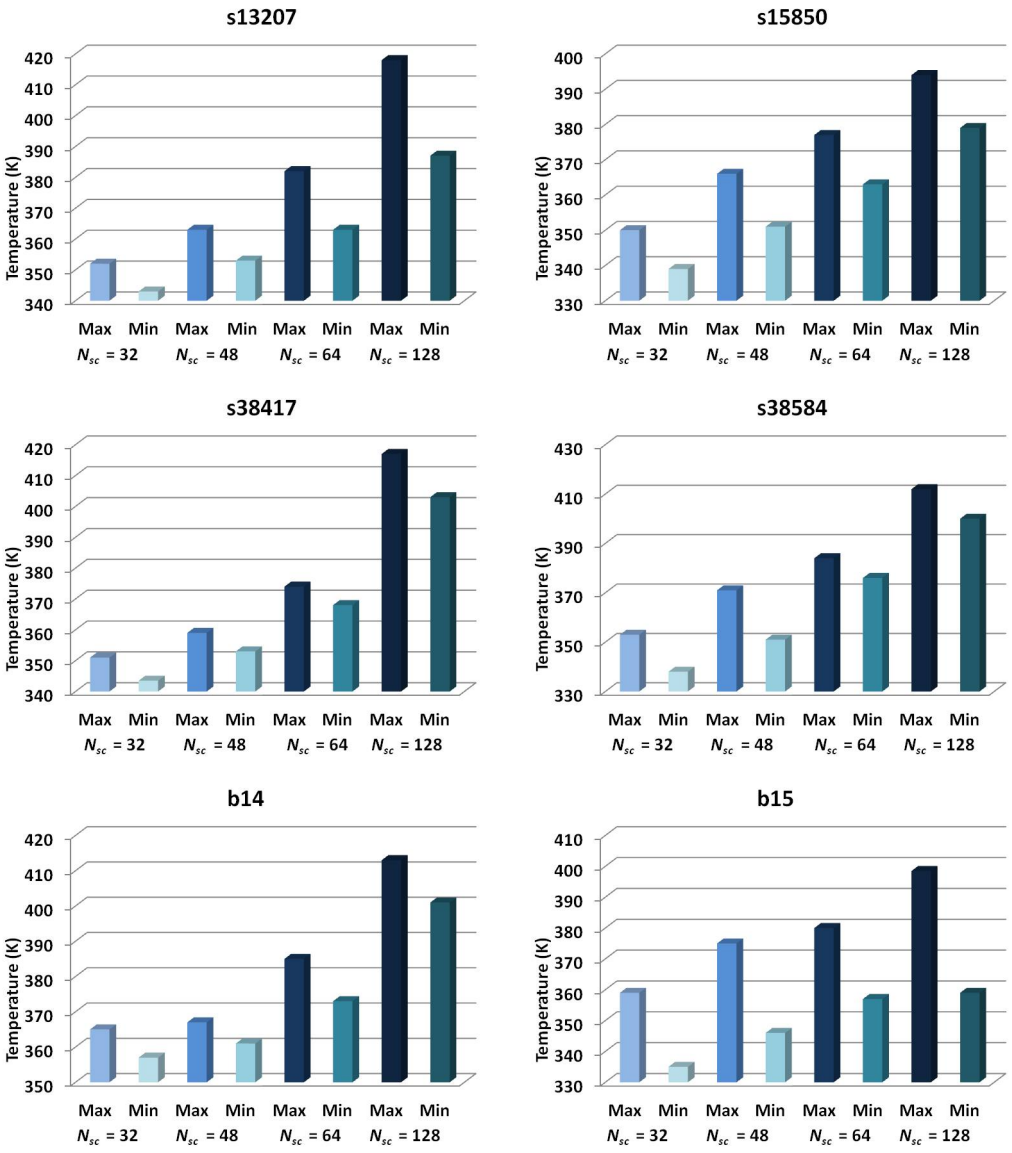


Figure 5.6: Variation of temperature for different clique partitions

may not be applicable for complex circuits with huge amount of test data. Moreover, this method does not fill the don't-care bits to minimize peak temperature. To get a significant reduction in the peak temperature of the chip, some thermal-aware don't-care filling technique has to be incorporated into the clique partitioning heuristic. Compression ratio may have to be compromised to some extent to get a good thermal-aware test pattern set.

High power consumption during shifting of test vectors in the scan chains plays a major role in temperature increase of the circuit. Large number of intermediate patterns are generated during shifting in (out) of a particular test vector (response) through the scan chains. Power consumption increases because of these intermediate patterns.

Minimizing the transition count in scan chains by efficient thermal-aware don't-

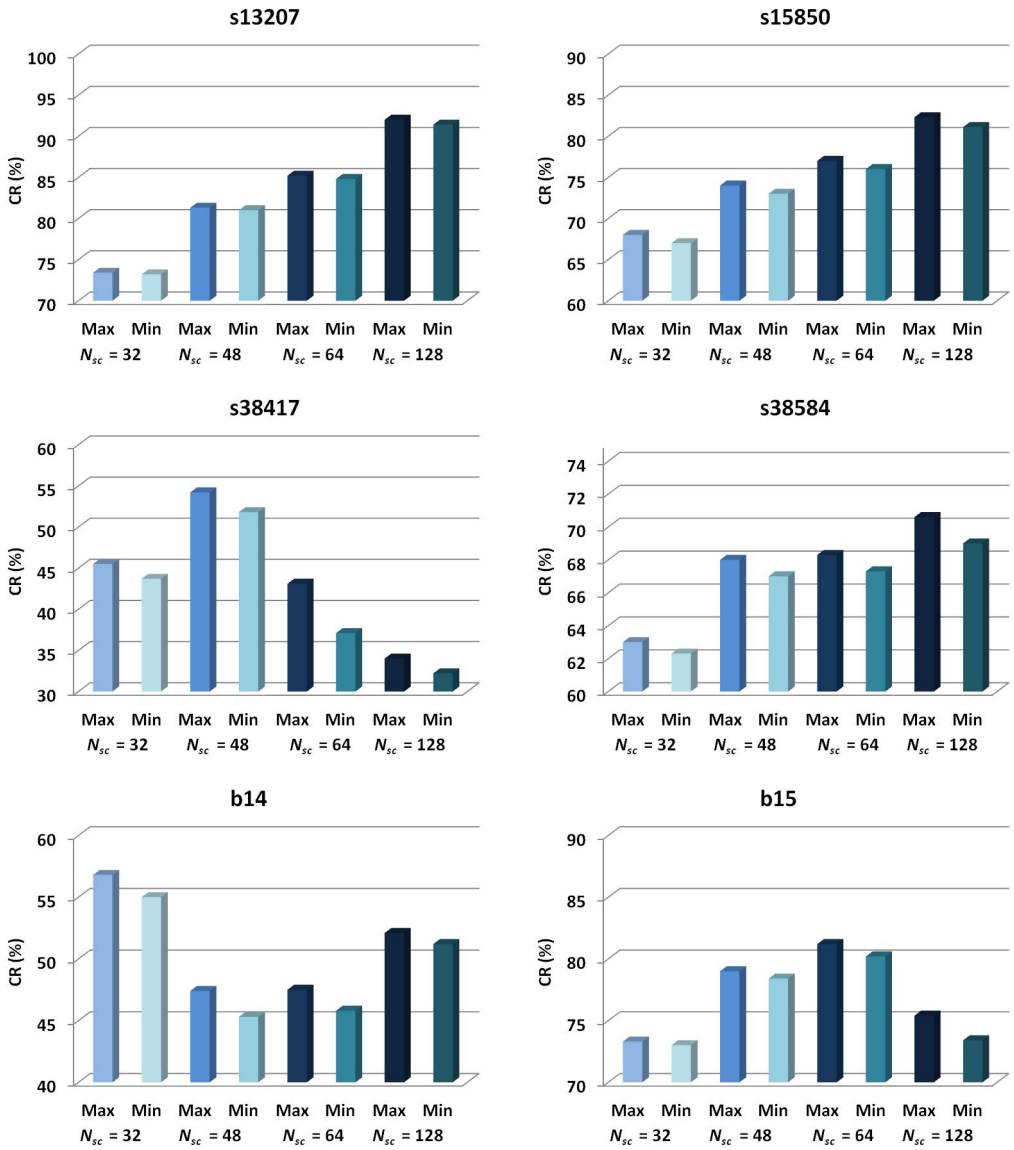


Figure 5.7: Variation of compression ratio for different clique partitions

care filling is a feasible solution to reduce peak temperature, as well as circuit power.

Thermal-aware don't-care filling technique presented in [31] produces reasonably low temperature test pattern set. In [31], the entire circuit has been divided into several blocks consisting of gates and flip-flops. A power estimation tool identifies the flip-flops with high transitions in each block. These flip-flops are called critical flip-flops as they play leading roles in high power consumption of the block. Each block has been given a weight as per the power consumption of the block. However, as the power of neighboring blocks may have a large impact on the temperature of a particular block, floorplan information of the circuit has to be incorporated with the power estimation tool to estimate the temperature of the blocks. The temperature of the hottest block with highest estimated temperature

has been tried to be minimized with an aim to minimize the peak temperature of the circuit.

Algorithm 5.2: ***Temperature_Compression_Trade-off***

input : Original test patterns TP_O with don't-care bits
output: Test patterns TP_{TOF} with a trade-off between temperature and compression ratio

1 **begin**
2 Fill up don't-care bits present in the test patterns using thermal-aware don't-care bit filling technique [31] to get thermal-aware test patterns TP_{TH};
3 Store the thermal-aware test slices obtained from TP_{TH} in thermal-aware $Slice\_Info$ (SI_{TH});
4 Apply clique partitioning heuristic [81] on TP_O; Get corresponding compression-aware $Clique\_List$ (CL_{CM}) and $Slice\_Info$ (SI_{CM}); Compute compression-aware test pattern TP_{CM} from SI_{CM};
5 Sort the compression-aware $Clique\_List$ (CL_{CM}) in descending order on the basis of the number of elements present in a clique;
6 Select a weightage factor Wt within a range of 0 to 1.
7 ($0 \rightarrow$ thermal-aware don't-care filling; $1 \rightarrow$ compression-aware don't-care filling);
8 Calculate total slice conversion $TSC = \lfloor Wt \times n \times l \rfloor$;
9 The slices of SI_{TH} corresponding to top most TSC clique members of sorted $Clique\_List$ (CL_{CM}) are replaced with the slices from SI_{CM}. Modified slice info is SI_{NEW};
10 Compute new test patterns TP_{NEW} from SI_{NEW};
11 Apply clique partitioning heuristic [81] on TP_{NEW}; Get corresponding $Clique\_List$ (CL_{TOF}) and $Slice\_Info$ (SI_{TOF}) ;Compute test patterns TP_{TOF} from SI_{TOF}; Calculate compression ratio of TP_{TOF}; Apply TP_{TOF} to the circuit; calculate peak temperature using temperature calculation steps mentioned in section 5.4.1;

In this section, we present a heuristic named $Temperature\_Compression\_Trade\text{-}off$ (Algorithm 5.2), which does a trade-off between compression ratio and the temperature of the chip. We have used the test pattern set generated using the method described in [31] as the base case of thermal-aware solution. The don't-care filling technique mentioned in [31] can only handle circuits with single scan chain set-up. Our modified thermal-aware don't-care bit filling technique can fill up don't-care bits taking care of the multiple scan chains. We start with two test pattern sets, one with all the don't-care bits filled up using thermal-aware don't-care filling approach and the other one is the original test pattern set TP_O with all the don't-care bits retained. The test pattern set with filled don't-cares can be named as *thermal-aware test pattern set* (TP_{TH}). Our objective is to find the test patterns TP_{TOF} with a trade-off between temperature and compression ratio. As

no don't-care bit is present in TP_{TH}, it cannot use the flexibility of having don't-care bits, which can be filled up suitably to increase the number of compatible slices. Hence, it produces a poor compression ratio. On the other hand, TP_O has the full flexibility to fill up the don't-care bits to achieve a high compression ratio. Clique partitioning heuristic [81] is applied on TP_O to generate a compression-aware *Clique_List* (CL_{CM}) and corresponding *Slice_Info* (SI_{CM}). The slices belonging to SI_{CM} are called *compression-aware slices* as the test pattern set TP_{CM} formed from SI_{CM} produces a good compression ratio. The slices obtained from TP_{TH} are called *thermal-aware slices* due to their low-temperature effect. However, these *compression-aware slices* and *thermal-aware slices* differ in the process of their don't-care bit filling. A new *Slice_Info* (SI_{NEW}) can be formed taking some of the *thermal-aware slices* from SI_{TH} and some of the *compression-aware slices* from SI_{CM} to have the advantage of both good compression as well as low temperature. A new set of test patterns is generated using the slice information of SI_{NEW}. Clique partitioning heuristic [81] is applied on TP_{NEW} to get a clique list CL_{TOF} and corresponding *Slice_Info* (SI_{TOF}). Finally test pattern set TP_{TOF}, computed from SI_{TOF}, produces a balance between compression ratio and temperature. However, the percentage of *compression-aware slices* that will be there in SI_{NEW} (with the rest of the *thermal-aware slices*), can be chosen flexibly using a weight factor Wt with a value in the range of '0' to '1'. A '0' value of Wt chooses all the *thermal-aware slices*, while a '1' value of it refers to all *compression-aware slices* being chosen. Any value between '0' and '1' is a trade-off between temperature and compression ratio.

5.6.1. Experimental Results

In this section, we present the results on several ISCAS'89 and ITC'99 benchmarks. Tables 5.3 and 5.4 show the trade-off between temperature and compression ratio for different weight factors. Temperature has been calculated using the method mentioned in Section 5.4.1. Compression ratio (CR) for the dictionary size D = 128 for different N_{SC} values are shown in the table. Wt is varied from '0' to '1', and for each value of Wt corresponding temperature in Kelvin (T(K)) and CR (in %) are noted for different N_{SC}. It may be noted that, for the value of $Wt = 0$, all don't-cares are filled thermally, hence, the test pattern set produces minimum temperature, but the corresponding compression ratio is very poor, while for $Wt = 1$, although the compression ratio is impressive, it produces a high peak temperature, which can be a serious threat to the thermal safety of the chip. Other values of Wt show a balance between compression ratio and temperature. For $Wt = 0$, all the slices being *thermal-aware*, are expected to produce minimum temperature. With the gradual increase of Wt from '0' to '1', more numbers

of *compression-aware slices* replace *thermal-aware slices*, which increase the compression ratio, but fail to reduce the temperature of the chip. It may be noted that in some cases, the trade-off may not be available. In Table 5.4, one such situation could be observed for circuit $b14$ with Wt larger than 0.5. This we believe has happened due to the heuristic nature of the clique partitioning process.

Figure 5.8 shows a graphical representation of CR and temperature for different values of Wt, and scan chain set-ups for different ISCAS'89 and ITC'99 benchmarks. It may be noted from the figure that for all the benchmarks, temperature increases to a large extent with the increase of the value of N_{SC}. This is due to transfer of more number of test data bits at a time, which increase transition counts in the scan chains and hence more power consumption. Increasing the number of scan chains may transfer test data faster reducing test time of the chip, but at the same time increase in the temperature may cause damage of the chip. However, the variation of CR with N_{SC} varies from circuit to circuit. Although few benchmarks like $s13207$ and $s15850$ show better compression for larger N_{SC}, some other benchmarks like $s38417$, show opposite effect. Also for some other benchmarks CR do not vary much with the increase of N_{SC}.

As none of the previous compression works reported in the literature have addressed the thermal issue in compression, we could not make a direct comparison of our work with those. However, our best case CR is comparable with the results presented in [81]. Although some other works [115, 11] achieve a better compression ratio with more computational complexity and extra hardware overhead, all are based on clique partitioning. Method like dividing the test slices further into smaller sub-slices [115] can also be incorporated with our method to improve compression ratio without hampering the balance between CR and temperature of the chip.

5.7. Temperature Reduction Without Compromising Compression Ratio

In the last section, we have shown the trade-off between temperature and compression ratio. It takes some of the test slices from the set of *compression-aware slices* and others from the set of *thermal-aware slices*, to form the final set of test slices. Obviously we had to sacrifice some compression ratio to reduce the temperature of the chip. Similarly, the chip becomes hotter with a betterment of the compression ratio. However, our main objective in thermal-aware test data compression is to minimize the temperature of the chip without compromising the compression ratio, to get the advantage of both reduced temperature and good compression ratio.

Table 5.3: Variation Of Temperature And Compression Ratio For Different Values Of Wt For Different Scan Chains (For $D = 128$)

			Weightage Factor Wt												
Circuit Name	N_{SC}		**0**	**0.05**	**0.1**	**0.15**	**0.2**	**0.25**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**1**	**[81]**
$s38584$	**32**	**T(K)**	333.83	335.72	336.89	337.86	339.03	339.64	346.77	350.33	352.42	353.95	355.37	356.67	-
		CR (%)	28.51	28.92	29.06	29.28	30.26	31.57	36.69	41.66	47.30	53.32	60.01	62.41	-
	48	**T(K)**	342.34	345.27	347.76	349.66	350.8	353.14	361.63	365.34	368.14	370.46	372.3	374.03	-
		CR(%)	23.03	23.69	24.37	25.51	27.04	30.06	38.43	45.01	52.81	60.74	67.47	67.47	67.9
	64	**T(K)**	347.85	354.45	358	360.57	363.61	366.66	373.1	378.05	381.26	383.56	385.36	386.91	-
		CR(%)	18.82	19.82	20.87	22.78	25.06	27.82	37.95	46.35	54.53	62.35	67.37	70.13	70.8
	128	**T(K)**	376.87	381.16	385.61	390.67	395.19	398.36	405.74	410.09	413.84	416.39	418.44	418.77	-
		CR(%)	9.39	11.76	15.83	19.6	23.55	28.11	40.75	49.68	58.07	65.43	67.97	70.87	70.7
$s38417$	**32**	**T(K)**	333.98	337.19	339.9	342.15	344.32	345.53	348.58	349.55	350.66	351.8	352.75	353.64	-
		CR(%)	9.53	11.11	13.67	16.47	19.35	22.02	31.46	37.79	41.30	42.02	42.20	45.01	-
	48	**T(K)**	342.7	346.69	349.28	352.24	354.53	355.88	358.56	359.68	360.87	361.91	362.76	363.61	-
		CR(%)	11.12	13.08	16.17	19.3	22.42	25.43	35.16	42.38	49.47	52.43	55.43	57.92	61.8
	64	**T(K)**	352.05	358.84	362.83	366.3	368.22	369.89	373.08	374.36	375.6	377.06	377.58	378.46	-
		CR(%)	10.89	12.91	14.98	17.68	20.79	23.63	35.21	37.19	37.93	38.49	41.73	43.3	42.7
	128	**T(K)**	390.49	396.17	399.25	402.55	406.06	408.21	414.93	417.52	417.88	418.59	419.6	420.88	-
		CR(%)	9.28	13.39	17.28	20.96	24.33	27.49	32.85	33.58	34.95	36.57	36.57	36.57	36.7
$s13207$	**32**	**T(K)**	328.41	330.7	332.8	335.05	337.38	339.58	345.81	346.91	346.73	351.48	354.55	357.28	-
		CR(%)	62.03	62.03	62.04	62.06	62.06	62.07	62.41	62.56	63.08	63.96	65.13	73.25	-
	48	**T(K)**	331.37	334.49	337.33	340.18	342.83	344.63	349.67	349.69	353.94	359.4	362.1	365.09	-
		CR(%)	63.81	63.81	63.86	63.86	64.05	64.05	65.57	65.79	67.55	69.63	71.92	81.01	81.1
	64	**T(K)**	338.7	345.46	351.76	357.24	362.27	367.21	379.03	379.18	385.79	388.81	391.45	394.73	-
		CR(%)	63.28	63.4	63.4	63.58	63.83	64.34	65.61	66.95	69.29	71.68	75.8	85.01	85.4
	128	**T(K)**	352.33	363.79	371.03	379.73	385.72	390.62	405.4	417.02	424.4	429.56	432.72	436.18	-
		CR(%)	63.5	63.64	63.9	64.77	66.38	66.77	69.05	72.05	74.66	78.19	83.4	91.3	91.5

Table 5.4: Variation Of Temperature And Compression Ratio For Different Values Of Wt For Different Scan Chains (For $D = 128$) (Contd.)

			Weightage Factor Wt												
Circuit Name	N_{SC}		**0**	**0.05**	**0.1**	**0.15**	**0.2**	**0.25**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**1**	**[81]**
$s15850$	**32**	**T(K)**	332.32	333.78	335.27	336.56	337.84	338.7	341.76	345.85	348.96	351.37	353.08	354.38	-
		CR(%)	45.46	45.46	45.46	45.58	45.80	46.01	47.94	48.96	51.84	56.27	62.32	67.25	-
	48	**T(K)**	336.68	340.9	344.98	348.99	350.51	352.92	360.62	363.23	365.74	368.04	369.66	370.44	-
		CR (%)	42.18	42.24	42.34	42.76	43.23	43.96	47.3	50.74	56.47	63.1	69.25	74.2	74.4
	64	**T(K)**	341.93	347.68	352.43	356.04	359.48	361.92	369.8	372.71	375.37	377.91	379.61	379.32	-
		CR(%)	38.79	38.84	39.58	41.55	41.76	43.46	48.76	50.95	54.2	67.62	73.43	75.84	75.8
	128	**T(K)**	365.53	371.58	375.86	379.18	381.57	382.25	387.94	390.85	393.56	395.33	396.35	394.37	-
		CR(%)	35.25	35.8	38.38	41.53	43.94	45.88	57.13	64.64	72.44	78.59	80.84	81.59	81.9
$b14$	**32**	**T(K)**	346.56	348.55	350	351.18	354.7	357.83	362.05	364.27	366.03	367.48	368.63	369.67	-
		CR (%)	31.48	31.43	31.79	31.98	32.67	33.91	37.73	40.98	45.38	50.12	53.30	53.30	-
	48	**T(K)**	341.36	343.62	345.71	348.44	351.25	353.53	358.9	361.76	364.38	366.64	368.49	370.07	-
		CR(%)	22.74	23.58	24.57	25.6	27.56	29.95	37.15	42.96	46.47	46.47	46.47	45.61	-
	64	**T(K)**	350.67	350.77	350.75	353.82	357.78	360.44	370.32	375.98	380.91	384.97	387.97	389.43	-
		CR(%)	26.8	26.91	26.97	27.58	29.01	30.6	37.94	44.32	46.31	45.96	44.74	43.78	-
	128	**T(K)**	379.7	379.94	380.06	380.18	380.37	380.41	390.16	401.17	408.89	414.34	416.69	416.96	-
		CR(%)	38.1	38.1	38.1	38.1	39.4	39.4	42.4	45.57	51.69	53.79	55.76	61.72	-
$b15$	**32**	**T(K)**	326.11	328.45	331.79	335.59	339.97	344.82	354.71	357.15	360.55	363.67	366.73	370.23	-
		CR(%)	55.62	55.55	55.55	55.70	56.62	57.03	59.10	59.84	60.92	62.64	65.49	73.09	-
	48	**T(K)**	330.69	331.69	334.34	343.07	349.85	356.71	361.58	365.29	368.95	373.22	376	379.67	-
		CR(%)	45.44	45.51	45.68	47	48.27	50.74	53.78	56.92	60.03	65.01	69.82	78.95	-
	64	**T(K)**	332.41	338.23	343.08	348.32	352.09	355.72	365.03	370.87	373.81	377.67	381.08	384.21	-
		CR(%)	33.98	35.25	36.93	39.73	41.6	43.62	49.57	54.58	59.75	65.67	72.12	80.65	-
	128	**T(K)**	349.24	358.32	367.33	373.99	379.31	384.5	391.78	398.11	402.66	406.28	408.85	409.99	-
		CR(%)	18.71	21.4	24.55	28.23	31.59	34.95	44.82	52.63	60.08	67.63	73.53	73.53	-

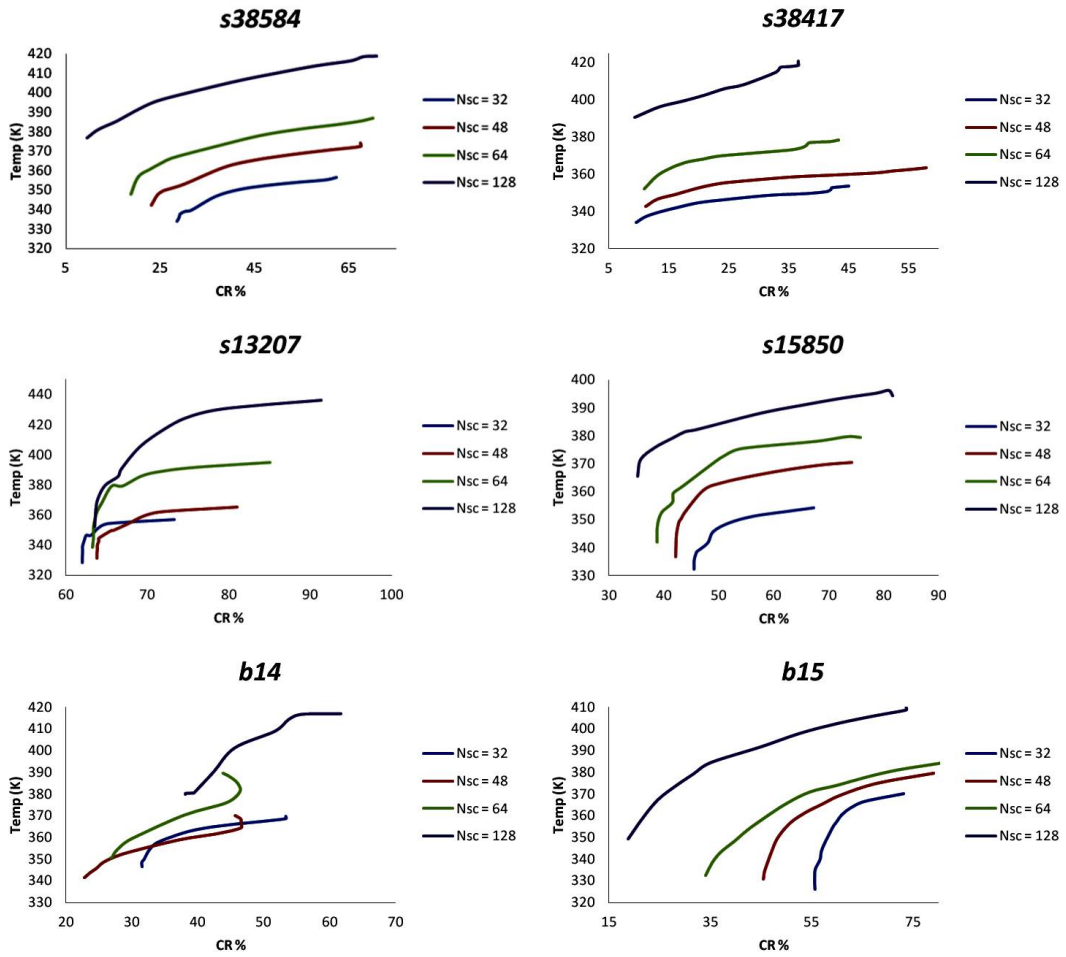


Figure 5.8: Variation of temperature and CR with Wt for different scan chain set-up for different ISCAS'89 and ITC'99 benchmarks.

It may be noted that, in dictionary based coding, compression ratio totally depends on the number of dictionary elements and the number of test slices that can be represented by any of the dictionary entries. So, our main objective in dictionary based coding is to fill the don't-care bits present in the test patterns in such a manner that more number of test slices can be represented by any dictionary entry. Due to the dictionary size constraint, only few of the representative test slices can be accommodated in the dictionary. The don't-care bit filling of the rest of the test slices do not affect the compression ratio.

Instead of selecting few slices from the *compression-aware slices* and rest from the *thermal-aware slices* depending upon the Wt value, if we select all the *thermal-aware slices* for the test slices whose representatives are not stored in the dictionary, temperature can be minimized without hampering the compression ratio. Moreover, the dictionary elements may also have don't-care bits. Efficient filling of these don't-care bits can also be helpful in further reduction of peak temperature.

Let us call the test slices, which are represented by any of the dictionary element, *represented test slices*. It may be noted that, a don't-care bit at any position of a dictionary element suggests a don't-care bit at the same position in all the represented test slices. It may so happen that the thermal-aware don't-care bit filling suggests different values of these don't-care bits in the represented test slices. If we fill all the don't-cares in the represented test slices, as suggested by the thermal-aware don't-care bit filling, that may hamper the compression ratio. To keep compression ratio unchanged, those don't-care bits in the represented test slices have to be filled with the same value, either '0' or '1'. To keep maximum number of thermal-aware bits intact in the set of final test slices, we have first consulted the *thermal-aware slices*. If for a particular don't-care bit, *thermal-aware slices* of the represented test slices suggests a '0' value in more number of cases than a '1' value, we have filled that don't-care with a '0' value and vice-verse. Algorithm 5.3 presents the procedure of temperature reduction without compromising the compression ratio.

The total flow of the algorithm can be explained with the Figure 5.9. We have considered the same example compatibility graph mentioned in Figure 5.3. From this graph, we have obtained the representative test slices after performing clique partitioning on it. *compression-aware slices* have been formed using these representative test slices. We have considered a dictionary of size 4. So only 4 of the largest representative test slices (i.e. {5,4,6,8}, {11,12,13,14}, {1,2} and {3,7}) can be accommodated in the dictionary. On the other hand *thermal-aware slices* have been formed by filling the don't-care bits of the test slices to reduce the peak temperature. However, it may be noted from the dictionary elements that the 7^{th} bit of the 2^{nd} dictionary element (i.e. corresponding to representative test slice of test slices{11,12,13,14}) is don't-care. If we look into the 7^{th} bit of the represented test slices (i.e. 11^{th}, 12^{th}, 13^{th} and 14^{th}) in *thermal-aware slices*, we can see that, in 3 out of 4 cases (i.e. 11^{th}, 12^{th} and 13^{th}), thermal-aware bit filling suggests a '0' value for that don't-care bit. So, we have filled that don't-care bit (i.e. 7^{th} bit) in all the four test slices with a '0' value to keep more number of thermal-aware bits in the final test pattern without compromising the compression ratio. Similarly, other don't-care bits corresponding to other dictionary elements, have been filled up in the final test pattern set.

5.7.1. Experimental Results

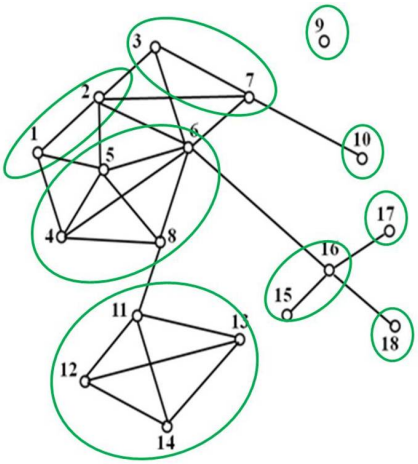
In this section we have shown the results of temperature reduction without compromising the compression ratio. Table 5.5 shows the results of our experimentation for different ISCAS'89 and ITC'99 benchmarks. In this table we have shown two comparisons between the methods $Temperature\_Compression\_Trade\text{-}off$

Algorithm 5.3: *Temperature_Reduction*

input : Compression-aware *Clique_List* (CL_{CM}) and *Slice_Info* (SI_{CM}); Thermal-aware *Slice_Info* (SI_{TH}); Dictioary Size D;
output: Final test patterns TP_{FINAL} with reduced temperature and high compression ratio;

```
1  begin
2      Select D largest cliques from Clique_List (CL_CM);
3      Form the dictionary with the representative test slices of D largest
       cliques;
4      Select the test slices from Slice_Info (SI_CM) corresponding to the
       members of D largest cliques to form a new set of test slices
       Slice_Info (SI_FINAL) with rest of the slices from Slice_Info (SI_TH);
5      for All the D dictionary entries do
6          for j ← 1 to m do
7              if j is a don't-care bit then
8                  Mark the represented test slices of the dictionary entry;
9                  Count_Zero ← 0;
10                 Count_One ← 0;
11                 for All the represented test slices do
12                     if j^th bit is '0' in
13                     thermal-aware slices then
14                         Count_Zero ++;
15                     else
16                         Count_One ++;
17                 if (Count_Zero > Count_One) then
18                     Fill the j^th bit of all the marked slices with '0' in
                       (SI_FINAL);
19                 else
20                     Fill the j^th bit of all the marked slices with '1' in
                       (SI_FINAL);
21     Form the TP_FINAL from SI_FINAL;
```

(TOFF) and *Temperature_Reduction* (TR). First we have shown the reduction in temperature for same compression ratio (CR) value and then the improvement in CR for same temperature value. The 4^{th}, 5^{th} and 6^{th} columns of the table show the reduction in temperature that can be achieved using our proposed *Temperature_Reduction* method, without compromising the best case CR (i.e. the CR achieved for $wt = 1$ using *Temperature_Compression_Trade-off* procedure). Column 6 reports the % reduction in temperature. We have calculated the % reduction using the equation 5.1, where $Temp_{TOFF}(wt = 1)$ suggests the temperature value obtained using the method *Temperature_Compression_Trade-off*, considering all the test slices filled up using *compression-aware slices* and



Representative Test Slices | **Slice Index**

1	0	0	1	0	0	0	1	{1,2}
1	0	x	0	0	0	x	1	{3,7}
1	0	1	1	0	0	1	0	{4,5,6,8}
0	1	x	x	0	0	0	x	{9}
x	1	0	x	1	1	x	x	{10}
0	0	1	0	0	1	x	0	{11,12,13,14}
0	1	0	1	x	0	1	x	{15,16}
0	1	0	x	x	x	1	x	{17}
x	1	x	1	x	0	1	0	{18}

Compression-Aware Test Slices

Slice index	Scan chain index							
	1	2	3	4	5	6	7	8
1	1	0	0	1	0	0	0	1
2	1	0	0	1	0	0	0	1
3	1	0	x	0	0	0	x	1
4	1	0	1	1	0	0	1	0
5	1	0	1	1	0	0	1	0
6	1	0	1	1	0	0	1	0
7	1	0	x	0	0	0	x	1
8	1	0	1	1	0	0	1	0
9	0	1	x	x	0	0	0	x
10	x	1	0	x	1	1	x	x
11	0	0	1	0	0	1	x	0
12	0	0	1	0	0	1	x	0
13	0	0	1	0	0	1	x	0
14	0	0	1	0	0	1	x	0
15	0	1	0	1	x	0	1	x
16	0	1	0	1	x	0	1	x
17	0	1	0	x	x	x	1	x
18	x	1	x	1	x	0	1	0

4 Elements Dictionary

1	0	1	1	0	0	1	0
0	0	1	0	0	1	x	0
1	0	0	1	0	0	0	1
1	0	x	0	0	0	x	1

Final Test Slices

Slice index	Scan chain index							
	1	2	3	4	5	6	7	8
1	1	0	0	1	0	0	0	1
2	1	0	0	1	0	0	0	1
3	1	0	0	0	0	0	1	1
4	1	0	1	1	0	0	1	0
5	1	0	1	1	0	0	1	0
6	1	0	1	1	0	0	1	0
7	1	0	0	0	0	0	1	1
8	1	0	1	1	0	0	1	0
9	0	1	0	0	0	0	0	0
10	1	1	0	1	1	1	1	1
11	0	0	1	0	0	1	0	0
12	0	0	1	0	0	1	0	0
13	0	0	1	0	0	1	0	0
14	0	0	1	0	0	1	0	0
15	0	1	0	1	1	0	1	1
16	1	1	1	1	1	0	1	1
17	0	1	0	1	1	1	1	1
18	1	1	0	1	0	0	1	0

Thermal-Aware Test Slices

Slice index	Scan chain index							
	1	2	3	4	5	6	7	8
1	1	0	1	1	0	1	0	1
2	1	0	0	0	0	0	0	1
3	1	0	0	0	0	0	1	1
4	1	0	1	1	0	0	0	0
5	1	0	1	1	0	0	0	0
6	1	0	0	0	0	0	1	0
7	1	0	0	0	0	1	1	1
8	1	0	1	1	0	0	1	0
9	0	1	0	0	0	0	0	0
10	1	1	0	1	1	1	1	1
11	1	0	0	0	0	1	0	0
12	0	0	0	0	0	1	0	0
13	0	0	1	0	0	1	0	0
14	0	0	1	0	0	1	1	1
15	0	1	0	1	1	0	1	1
16	1	1	1	1	1	0	1	1
17	0	1	0	1	1	1	1	1
18	1	1	0	1	0	0	1	0

Figure 5.9: An example flow of thermal-aware test data compression

$Temp_{TOFF}(wt = 0)$ suggests the temperature value obtained using the method $Temperature\_Compression\_Trade\text{-}off$, considering all the test slices filled up using *thermal-aware slices*. $Temp_{TR}$ represents the temperature value obtained using $Temperature\_Reduction$ method.

$$\%\text{Temp. Reduction} = \frac{(Temp_{TOFF}(wt = 1) - Temp_{TR})}{(Temp_{TOFF}(wt = 1) - Temp_{TOFF}(wt = 0))} \times 100 \quad (5.1)$$

It may be noted from column 6 that upto 64% of reduction in temperature can be achieved without compromising the compression ratio, using the proposed $Temperature\_Reduction$ method.

We have also noted the compression ratio that would have been achieved using the $Temperature\_Compression\_Trade\text{-}off$ method for the same temperature that we have achieved considering the best case CR for $Temperature\_Reduction$ method. Column 8 reports the temperatures and compression ratios that we have obtained using $Temperature\_Reduction$ method, while column 7 reports the compression ratio obtained from $Temperature\_Compression\_Trade\text{-}off$ method, for the same temperature value. We have also calculated the % improvement in compression ratio using the equation 5.2, where $CR_{TR}(Temp_{TR})$ notes the compression ratio obtained considering best case CR, using $Temperature\_Reduction$ method and $CR_{TOFF}(Temp_{TR})$ notes compression ratio for $Temperature\_Compression\_Trade\text{-}off$ method, obtained at the same temperature. $CR_{TOFF}(wt = 1)$ and $CR_{TOFF}(wt = 0)$ are the compression ratio obtained using $Temperature\_Compression\_Trade\text{-}off$ method considering all test slices *thermal-aware slices* and *compression-aware slices* respectively.

$$\%\text{CR improvement} = \frac{(CR_{TR}(Temp_{TR}) - CR_{TOFF}(Temp_{TR}))}{(CR_{TOFF}(wt = 1) - CR_{TOFF}(wt = 0))} \times 100 \quad (5.2)$$

It may be noted from the table that an improvement of upto 95% in CR can be achieved for same temperature value using the $Temperature\_Reduction$ method. However, CR does not improve for $N_{sc} = 64$ of benchmark $b14$. This can be explained using the Table 5.4, which shows CR does not improve for a higher value of wt for some cases of benchmark $b14$. A lesser value of wt shows better compression than a higher value of it. Although, the $Temperature\_Reduction$ method reduces the temperature by 34.91% from the maximum reported temperature (for $wt = 1$), it is unable to produce an improvement in CR, as for this case, $wt = 1$ does not report best CR.

Table 5.5: Reduction In Temperature For Same Compression Ratio And Improvement Of CR For Same Temperature For Different Benchmarks

Circuit Name			For same CR			For same Temperature		
	N_{SC}		TOFF	TR	% Reduction in Temperature	TOFF	TR	% Impv. in CR
$s38584$	**32**	**T(K)**	356.67	351.06	**24.56**	351.06	351.06	-
		CR (%)	62.41	62.41	-	44.47	62.41	**52.91**
	48	**T(K)**	374.03	367.58	**20.35**	367.58	367.58	-
		CR(%)	67.47	67.47	-	50.12	67.47	**39.04**
	64	**T(K)**	386.91	376.76	**25.99**	376.76	376.76	-
		CR(%)	70.13	70.13	-	44.38	70.13	**50.19**
	128	**T(K)**	418.77	411.11	**18.28**	411.11	411.11	-
		CR(%)	70.87	70.87	-	50.89	70.87	**32.50**
$s38417$	**32**	**T(K)**	353.64	351.5	**10.89**	351.5	351.5	-
		CR(%)	45.01	45.01	-	41.8	45.01	**9.05**
	48	**T(K)**	363.61	359.5	**19.66**	359.5	359.5	-
		CR(%)	57.92	57.92	-	35.1	57.92	**48.76**
	64	**T(K)**	378.46	373.3	**19.54**	373.3	373.3	-
		CR(%)	43.3	43.3	-	35.9	43.3	**22.83**
	128	**T(K)**	420.88	412.1	**28.89**	412.1	412.1	-
		CR(%)	36.57	36.57	-	30.57	36.57	**21.99**
$s13207$	**32**	**T(K)**	357.28	345.9	**39.42**	345.9	345.9	-
		CR(%)	73.25	73.25	-	62.5	73.25	**95.81**
	48	**T(K)**	365.09	354.71	**30.78**	354.71	354.71	-
		CR(%)	81.01	81.01	-	68.01	81.01	**75.58**
	64	**T(K)**	394.73	371	**42.35**	371	371	-
		CR(%)	85.01	85.01	-	70.12	85.01	**68.52**
	128	**T(K)**	436.18	407.02	**34.78**	407.02	407.02	-
		CR(%)	91.3	91.3	-	69.18	91.3	**79.57**
$s15850$	**32**	**T(K)**	354.38	340.2	**64.28**	340.2	340.2	-
		CR(%)	67.25	67.25	-	47.03	67.25	**92.77**
	48	**T(K)**	370.44	358.98	**33.96**	358.98	358.98	-
		CR (%)	74.2	74.2	-	46.2	74.2	**87.45**
	64	**T(K)**	379.32	358.93	**54.53**	358.93	358.93	-
		CR(%)	75.84	75.84	-	42.9	75.84	**88.91**
	128	**T(K)**	394.37	387.99	**22.12**	387.99	387.99	-
		CR(%)	81.59	81.59	-	57.3	81.59	**52.42**
$b14$	**32**	**T(K)**	369.67	360.59	**39.29**	360.59	360.59	-
		CR (%)	53.30	53.30	-	35.7	53.30	**80.67**
	48	**T(K)**	370.07	361.01	**31.56**	361.01	361.01	-
		CR(%)	45.61	45.61	-	42.3	45.61	**14.47**
	64	**T(K)**	389.43	375.9	**34.91**	375.9	375.9	-
		CR(%)	43.78	43.78	-	44.58	43.78	**-4.71**
	128	**T(K)**	416.96	406.1	**29.15**	406.1	406.1	-
		CR(%)	61.72	61.72	-	50.47	61.72	**47.63**
$b15$	**32**	**T(K)**	370.23	357.58	**28.67**	357.58	357.58	-
		CR(%)	73.09	73.09	-	60.16	73.09	**74.00**
	48	**T(K)**	379.67	363.12	**33.79**	363.12	363.12	-
		CR(%)	78.95	78.95	-	54.7	78.95	**72.37**
	64	**T(K)**	384.21	368.14	**31.02**	368.14	368.14	-
		CR(%)	80.65	80.65	-	52.89	80.65	**59.48**
	128	**T(K)**	409.99	394.24	**25.93**	394.24	394.24	-
		CR(%)	73.53	73.53	-	50.18	73.53	**42.59**

5.8. Conclusion

In this chapter we have presented a trade-off between compression ratio and temperature of the chip by efficiently filling the don't-care bits present in the test patterns. Clique partitioning guided dictionary based coding technique has been adopted for test data compression. The flexibility of our technique helps to choose a suitable balance between thermal safety and test cost to store huge amount of test data. Finally, we have proposed a thermal-aware test data compression technique that can reduce the temperature of the chip without compromising the compression ratio, hence provides the advantages of good compression and low temperature test.

Chapter 6
Conclusions and Future Works

6.1. Conclusions

The primary focus of this work has been to develop optimal test architecture and test scheduling strategy for System-on-Chip (SoC) under different constraints like resources, power and temperature. In the process of this exercise, we came up with a particle swarm optimization (PSO) based test scheduling strategy which outperforms most of the contemporary scheduling strategies proposed in the literature. PSO has its own advantage of faster convergence towards near optimal solution and its ability of better search space exploration has made it an extremely useful evolutionary technique. PSO guided bin-packing heuristic has been used in this work to solve the test scheduling problem. Although, the main objective of test scheduling is to assign test resources to the cores and selection of core ordering in test procedure to minimize the overall test application time, several other parameters like test power and temperature during testing have been major concerns for the test engineers now a days.

This particular research work has addressed the following test parameters at the time of test schedule generation for SoC.

1. Partitioning of test resources and resource allocation to the cores.

2. High power consumption issues during testing.

3. High temperature issues during testing.

High values of test mode power or temperature may cause a serious threat, even a permanent damage or burning of the chip. In this research work, test mode power and temperatures have been controlled under a certain safe limit at the time of generation of test schedule.

Chapter 3 mainly concentrates on power-aware test scheduling strategy, which can generate a test schedule with minimal TAT and without violating system level power limit. A new window-based peak power model has been proposed to

estimate the test mode power of the cores. This power model is simpler than cycle-accurate power model in terms of number of power values to be considered during the schedule generation process, hence, able to generate test schedule much faster. It also makes less power over-estimation, compared to the conventional global peak power model, which leads to more test parallelism. This reduces TAT.

Thermal issues during testing has been taken care of in Chapter 4. A superposition principle based fast and accurate thermal model has been used to estimate temperature of a core without online thermal simulations. This thermal model along with window-based peak power model has been integrated with PSO guided 3D-bin packing heuristic to generate a final power- and thermal-aware test schedule for SoC.

The increasing amount of test data of the cores increases the memory requirements and the test cost. An well known practice is to store the test data in a compressed manner in the memory. The huge amount of don't-care bits present in the test patterns are generally manipulated to increase the compression ratio. However, different don't-care bit filling strategies have direct impacts on switching activities during scan-shifts, hence, variation in the thermal profile. As high temperature during testing is not desirable, care must be taken at the time of don't-care filling. A trade-off between temperature and compression has been made in Chapter 5. As, high compression and low temperature is expected from the testing point of view, this chapter has also proposed a don't-care bit filling technique that can reduce temperature without compromising test compression.

6.2. Future Works

In the following we enumerate some probable direction for future works.

1. **Testing of 3D-Stacked ICs**

   Recently, 3D-IC has emerged to be a potential solution to the problem of long global interconnects of 2D-ICs. Instead of designing 2D-IC with long global interconnects, lengths can be reduced significantly by designing circuit components into several layers and bonding them together. Although 3D-Stacked ICs (SICs) provide several advantages over 2D-IC, testing of 3D-SIC has become more challenging because of its high complexity. Development of test infrastructure for 3D-SICs not only requires attention on test resource partitioning and resource allocation, but also on the transportation of test data to the different layers of the stack via Through Silicon Vias (TSVs). Power and thermal managements of 3D-SICs become even more difficult to handle. At the same time, these TSVs and the stacking process can also be potential sources of defects. As, the design for testability (DfT) for 3D-SICs

is yet to mature, the future works in test architecture development and test scheduling can be carried on for 3D-SICs.

2. **Multi-frequency Thermal-Aware Test Scheduling**

   Our current thermal-aware test scheduling strategy does not support multi-frequency test environment because of enormous size of thermal databases. The future work in this domain also includes modification of thermal model and database size to facilitate multi-frequency test environment.

3. **Integrated thermal optimization for advanced test compression strategies**

   In this thesis, we have proposed thermal-aware test data compression strategy based on dictionary coding. However, there exist many other test data compression strategies (as mentioned in Chapter 2), most of which manipulate the don't-cares to attain high test compression. Temperature reduction strategies can also be integrated with these test compression techniques to achieve high compression with at a reduced temperature.

4. **Integration of thermal-aware test compression with thermal-aware test scheduling**

   In the thesis, we have reported thermal-aware test data compression and thermal-aware test scheduling as separate problem. However, thermal-aware test data compression can be applied on the ATPG generated test patterns and these thermal-aware compressed test data can be used to test a core at the time of generating thermal-aware test schedule. This may help to reduce the temperature of each core as well as a better thermal-aware test schedule. The integration of thermal-aware test data compression with thermal-aware test scheduling can be a future scope of the work.

Dissemination out of this Research

- **Journals**
  - *Rajit Karmakar* and Santanu Chattopadhyay - "Window Based Peak Power Model and Particle Swarm Optimization Guided 3-D Bin Packing for SoC Test Scheduling", Elsevier Integration, the VLSI Journal (2015), vol- 50, pp. 61-73.
  - *Rajit Karmakar* and Santanu Chattopadhyay - "Particle Swarm Optimization Guided Bin-Packing Approach for Power- and Thermal-Aware Test Scheduling of System-on-Chip", Elsevier Microelectronics Journal. (Submitted after 1^{st} revision)
  - *Rajit Karmakar* and Santanu Chattopadhyay - "An Integrated Approach For Temperature, Test Time and Test Data Size Reduction In Dictionary Based Coding", IEEE Transactions on Very Large Scale Integration (VLSI) systems. (Submitted after 1^{st} revision)
- **Conferences**
  - *Rajit Karmakar* and Santanu Chattopadhyay - "Thermal-Aware Test Data Compression Using Dictionary Based Coding", in 28^{th} International Conference on VLSI Design (VLSID), pp.53-58, 2015.
  - *Rajit Karmakar*, Aditya Agarwal and Santanu Chattopadhyay - "Particle Swarm Optimization guided multi-frequency power-aware System-on-Chip test scheduling using window-based peak power model", in 18^{th} International Symposium on VLSI Design and Test (VDAT), pp.1,6, 16-18 July 2014.

Bibliography

[1] http://ddd.fit.cvut.cz/prj/atalanta-m/man.html.

[2] *http://en.wikipedia.org/wiki/Moore's-law.*

[3] http://vlsicad.eecs.umich.edu/bk/blobb/.

[4] *http://www.britannica.com/EBchecked/topic/705881/Moores-law.*

[5] *http://www.cad.polito.it/downloads/tools/itc99.html.*

[6] Faraday.http://www.faraday.com.tw/aip/ips/90library.html, January 2011.

[7] Ahn, J.-H., and Kang, S. Soc test scheduling algorithm using aco-based rectangle packing. In *Computational Intelligence*. Springer, 2006, pp. 655–660.

[8] Balakrishnan, K. J., and Touba, N. A. Matrix-based test vector decompression using an embedded processor. In *Defect and Fault Tolerance in VLSI Systems, 2002. DFT 2002. Proceedings. 17th IEEE International Symposium on* (2002), IEEE, pp. 159–165.

[9] Balakrishnan, K. J., and Touba, N. A. Deterministic test vector decompression in software using linear operations [soc testing]. In *VLSI Test Symposium, 2003. Proceedings. 21st* (2003), IEEE, pp. 225–231.

[10] Balakrishnan, K. J., and Touba, N. A. Relationship between entropy and test data compression. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 26*, 2 (2007), 386–395.

[11] Basu, K., and Mishra, P. Test data compression using efficient bitmask and dictionary selection methods. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 18*, 9 (2010), 1277–1286.

[12] Bayraktaroglu, I., and Orailoglu, A. Test volume and application time reduction through scan chain concealment. In *Proceedings of the 38th annual Design Automation Conference* (2001), ACM, pp. 151–155.

[13] Bayraktaroglu, I., and Orailoglu, A. Decompression hardware determination for test volume and time reduction through unified test pattern compaction and compression. In *VLSI Test Symposium, 2003. Proceedings. 21st* (2003), IEEE, pp. 113–118.

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

[15] Bonhomme, Y., Girard, P., Guiller, L., Landrault, C., and Pravossoudovitch, S. A gated clock scheme for low power scan testing of logic ics or embedded cores. In *Test Symposium, 2001. Proceedings. 10th Asian* (2001), IEEE, pp. 253–258.

[16] Butler, K., Saxena, J., Jain, A., Fryars, T., Lewis, J., and Hetherington, G. Minimizing power consumption in scan testing: pattern generation and dft techniques. In *Test Conference, 2004. Proceedings. ITC 2004. International* (Oct 2004), pp. 355–364.

[17] Cadence Inc. *Encounter. ,"Encounter user guide," Product Version 7.1.2, year = 2008,.*

[18] Chakrabarty, K. Design of system-on-a-chip test access architectures under place-and-route and power constraints. In *Proceedings of the 37th Annual Design Automation Conference* (2000), ACM, pp. 432–437.

[19] Chan, H. H., and Markov, I. L. Practical slicing and non-slicing block-packing without simulated annealing. In *Proceedings of the 14th ACM Great Lakes symposium on VLSI* (2004), ACM, pp. 282–287.

[20] Chandra, A., and Chakrabarty, K. Frequency-directed run-length (fdr) codes with application to system-on-a-chip test data compression. In *VLSI Test Symposium, 19th IEEE Proceedings on. VTS 2001* (2001), IEEE, pp. 42–47.

[21] Chandra, A., and Chakrabarty, K. System-on-a-chip test-data compression and decompression architectures based on golomb codes. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 20*, 3 (2001), 355–368.

[22] Chandra, A., and Chakrabarty, K. A unified approach to reduce soc test data volume, scan power and testing time. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 22*, 3 (2003), 352–363.

[23] Chang, X., Zhang, M., Zhang, G., Zhang, Z., and Wang, J. Adaptive clock gating technique for low power ip core in soc design. In *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on* (May 2007), pp. 2120–2123.

[24] Chen, M., and Orailoglu, A. Scan power reduction in linear test data compression scheme. In *Computer-Aided Design-Digest of Technical Papers, 2009. ICCAD 2009. IEEE/ACM International Conference on* (2009), IEEE, pp. 78–82.

[25] Chen, M., and Orailoglu, A. Scan power reduction for linear test compression schemes through seed selection. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 20*, 12 (2012), 2170–2183.

[26] Chen, M., and Orailoglu, A. Scan power reduction for linear test compression schemes through seed selection. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 20*, 12 (Dec 2012), 2170–2183.

[27] Cheng, W.-T., and Davidson, S. Sequential circuit test generator (stg) benchmark results. In *Circuits and Systems, 1989., IEEE International Symposium on* (May 1989), pp. 1939–1941 vol.3.

[28] Chou, R. M., Saluja, K. K., and Agrawal, V. D. Scheduling tests for vlsi systems under power constraints. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 5*, 2 (1997), 175–185.

[29] CUI, X.-L., CHENG, W., AND LI, C.-R. Effective soc test scheduling under peak power constraints based on the ant colony algorithm. *Microelectronics & Computer 7* (2011), 001.

[30] DABHOLKAR, V., CHAKRAVARTY, S., POMERANZ, I., AND REDDY, S. Techniques for minimizing power dissipation in scan and combinational circuits during test application. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 17*, 12 (Dec 1998), 1325–1333.

[31] DUTTA, A., KUNDU, S., AND CHATTOPADHYAY, S. Thermal aware don't care filling to reduce peak temperature and thermal variance during testing. In *Test Symposium (ATS), 2013 22nd Asian* (Nov 2013), pp. 25–30.

[32] EL-MALEH, A. H., AND AL-ABAJI, R. H. Extended frequency-directed run-length code with improved application to system-on-a-chip test data compression. In *Electronics, Circuits and Systems, 2002. 9th International Conference on* (2002), vol. 2, IEEE, pp. 449–452.

[33] FANG, H., TONG, C., AND CHENG, X. Runbasedreordering: a novel approach for test data compression and scan power. In *Proceedings of the 2007 Asia and South Pacific Design Automation Conference* (2007), IEEE Computer Society, pp. 732–737.

[34] FLORES, P., COSTA, J., NETO, H., MONTEIRO, J., AND MARQUES-SILVA, J. Assignment and reordering of incompletely specified pattern sequences targetting minimum power dissipation. In *VLSI Design, 1999. Proceedings. Twelfth International Conference On* (Jan 1999), pp. 37–41.

[35] GEKAS, G., NIKOLOS, D., KALLIGEROS, E., AND KAVOUSIANOS, X. Power aware test-data compression for scan-based testing. In *Electronics, Circuits and Systems, 2005. ICECS 2005. 12th IEEE International Conference on* (2005), IEEE, pp. 1–4.

[36] GHOSH, S., BASU, S., AND TOUBA, N. A. Joint minimization of power and area in scan testing by scan cell reordering. In *VLSI, 2003. Proceedings. IEEE Computer Society Annual Symposium on* (2003), IEEE, pp. 246–249.

[37] GIRARD, P., LANDRAULT, C., PRAVOSSOUDOVITCH, S., AND SEVERAC, D. Reducing power consumption during test application by test vector ordering. In *Circuits and Systems, 1998. ISCAS '98. Proceedings of the 1998 IEEE International Symposium on* (May 1998), vol. 2, pp. 296–299 vol.2.

[38] GIRI, C., SARKAR, S., AND CHATTOPADHYAY, S. A genetic algorithm based heuristic technique for power constrained test scheduling in core-based socs. In *Very Large Scale Integration, 2007. VLSI-SoC 2007. IFIP International Conference on* (2007), IEEE, pp. 320–323.

[39] GIRI, C., SARKAR, S., AND CHATTOPADHYAY, S. Test scheduling for core-based socs using genetic algorithm based heuristic approach. In *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence.* Springer, 2007, pp. 1032–1041.

[40] GOEL, S. K., AND MARINISSEN, E. J. Effective and efficient test architecture design for socs. In *Test Conference, 2002. Proceedings. International* (2002), pp. 529–538.

[41] Gonciari, P. T., Al-Hashimi, B. M., and Nicolici, N. Variable-length input huffman coding for system-on-a-chip test. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 22*, 6 (2003), 783–796.

[42] Gupta, R. K., and Zorian, Y. Introducing core-based system design. *Design & Test of Computers, IEEE 14*, 4 (1997), 15–25.

[43] Hamzaoglu, I., and Patel, J. Test set compaction algorithms for combinational circuits. In *Computer-Aided Design, 1998. ICCAD 98. Digest of Technical Papers. 1998 IEEE/ACM International Conference on* (Nov 1998), pp. 283–289.

[44] Hamzaoglu, I., and Patel, J. H. Reducing test application time for full scan embedded cores. In *Fault-Tolerant Computing, 1999. Digest of Papers. Twenty-Ninth Annual International Symposium on* (1999), IEEE, pp. 260–267.

[45] He, Z., Peng, Z., and Eles, P. A heuristic for thermal-safe soc test scheduling. In *Test Conference, 2007. ITC 2007. IEEE International* (Oct 2007), pp. 1–10.

[46] He, Z., Peng, Z., and Eles, P. Simulation-driven thermal-safe test time minimization for system-on-chip. In *Asian Test Symposium, 2008. ATS '08. 17th* (Nov 2008), pp. 283–288.

[47] He, Z., Peng, Z., and Eles, P. Thermal-aware test scheduling for core-based soc in an abort-on-first-fail test environment. In *Digital System Design, Architectures, Methods and Tools, 2009. DSD '09. 12th Euromicro Conference on* (Aug 2009), pp. 239–246.

[48] He, Z., Peng, Z., and Eles, P. Multi-temperature testing for core-based system-on-chip. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010* (March 2010), pp. 208–213.

[49] He, Z., Peng, Z., Eles, P., Rosinger, P., and Al-Hashimi, B. Thermal-aware soc test scheduling with test set partitioning and interleaving. *Journal of Electronic Testing 24*, 1-3 (2008), 247–257.

[50] Hetherington, G., Fryars, T., Tamarapalli, N., Kassab, M., Hassan, A., and Rajski, J. Logic bist for large industrial designs: real issues and case studies. In *Test Conference, 1999. Proceedings. International* (1999), IEEE, pp. 358–367.

[51] Hsu, F. F., Butler, K. M., and Patel, J. H. A case study on the implementation of the illinois scan architecture. In *Test Conference, 2001. Proceedings. International* (2001), IEEE, pp. 538–547.

[52] Huang, Y., Cheng, W.-T., Tsai, C.-C., Mukherjee, N., Samman, O., Zaidan, Y., and Reddy, S. M. Resource allocation and test scheduling for concurrent test of core-based soc design. In *Test Symposium, 2001. Proceedings. 10th Asian* (2001), pp. 265–270.

[53] Huang, Y., Reddy, S. M., Cheng, W.-T., Reuter, P., Mukherjee, N., Tsai, C.-C., Samman, O., and Zaidan, Y. Optimal core wrapper width selection and soc test scheduling based on 3-d bin packing algorithm. In *Test Conference, 2002. Proceedings. International* (2002), IEEE, pp. 74–82.

[54] Ichihara, H., Setohara, Y., Nakashima, Y., and Inoue, T. Test compression/decompression based on jpeg vlc algorithm. In *Asian Test Symposium, 2007. ATS'07. 16th* (2007), IEEE, pp. 87–90.

[55] Iyengar, V., Chakrabarty, K., and Marinissen, E. J. Efficient wrapper/tam co-optimization for large socs. In *Design, Automation and Test in Europe Conference and Exhibition, 2002. Proceedings* (2002), pp. 491–498.

[56] Iyengar, V., Chakrabarty, K., and Marinissen, E. J. On using rectangle packing for soc wrapper/tam co-optimization. In *VLSI Test Symposium, 2002. (VTS 2002). Proceedings 20th IEEE* (2002), pp. 253–258.

[57] Iyengar, V., Chakrabarty, K., and Marinissen, E. J. Test wrapper and test access mechanism co-optimization for system-on-chip. *Journal of Electronic Testing 18*, 2 (2002), 213–230.

[58] Janfaza, V., Behnam, P., Forouzandeh, B., and Alizadeh, B. A low-power enhanced bitmask-dictionary scheme for test data compression. In *VLSI (ISVLSI), 2014 IEEE Computer Society Annual Symposium on* (2014), IEEE, pp. 220–225.

[59] Jas, A., Ghosh-Dastidar, J., Ng, M.-E., and Touba, N. A. An efficient test vector compression scheme using selective huffman coding. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 22*, 6 (2003), 797–806.

[60] Jas, A., Ghosh-Dastidar, J., and Touba, N. A. Scan vector compression/decompression using statistical coding. In *VLSI Test Symposium, 1999. Proceedings. 17th IEEE* (1999), IEEE, pp. 114–120.

[61] Jas, A., Pouya, B., and Touba, N. A. Test data compression technique for embedded cores using virtual scan chains. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 12*, 7 (2004), 775–781.

[62] Jas, A., and Touba, N. A. Test vector decompression via cyclical scan chains and its application to testing core-based designs. In *Test Conference, 1998. Proceedings., International* (1998), IEEE, pp. 458–464.

[63] Kavousianos, X., Chakrabarty, K., Jain, A., and Parekhji, R. Test schedule optimization for multicore socs: handling dynamic voltage scaling and multiple voltage islands. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 31*, 11 (2012), 1754–1766.

[64] Kavousianos, X., Kalligeros, E., and Nikolos, D. Multilevel huffman coding: an efficient test-data compression method for ip cores. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 26*, 6 (2007), 1070–1083.

[65] Kavousianos, X., Kalligeros, E., and Nikolos, D. Optimal selective huffman coding for test-data compression. *Computers, IEEE Transactions on 56*, 8 (2007), 1146–1152.

[66] Kennedy, J., and Eberhart, R. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on* (Nov 1995), vol. 4, pp. 1942–1948 vol.4.

[67] Kim, K. S. Xmax: A practical and efficient compression architecture. In *Test Conference, 2005. Proceedings. ITC 2005. IEEE International* (2005), IEEE, pp. 2–pp.

[68] Knieser, M. J., Wolff, F. G., Papachristou, C. A., Weyer, D. J., and McIntyre, D. R. A technique for high ratio lzw compression. In *Proceedings of the conference on Design, Automation and Test in Europe-Volume 1* (2003), IEEE Computer Society, p. 10116.

[69] Koenemann, B., Barnhart, C., Keller, B., Snethen, T., Farnsworth, O., and Wheater, D. A smartbist variant with guaranteed encoding. In *Test Symposium, 2001. Proceedings. 10th Asian* (2001), IEEE, pp. 325–330.

[70] Koranne, S. Formulating soc test scheduling as a network transportation problem. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 21*, 12 (2002), 1517–1525.

[71] Koranne, S. On test scheduling for core-based socs. In *Design Automation Conference, 2002. Proceedings of ASP-DAC 2002. 7th Asia and South Pacific and the 15th International Conference on VLSI Design. Proceedings.* (2002), IEEE, pp. 505–510.

[72] Koranne, S., and Chakrabarty, K. A novel reconfigurable wrapper for testing of embedded core-based socs and its associated scheduling algorithm. In *SOC (System-on-a-Chip) Testing for Plug and Play Test Automation*. Springer, 2002, pp. 51–70.

[73] Krishna, C., Jas, A., and Touba, N. A. Test vector encoding using partial lfsr reseeding. In *Test Conference, 2001. Proceedings. International* (2001), IEEE, pp. 885–893.

[74] Krishna, C., and Touba, N. A. Adjustable width linear combinational scan vector decompression. In *Proceedings of the 2003 IEEE/ACM international conference on Computer-aided design* (2003), IEEE Computer Society, p. 863.

[75] Larsson, E., and Fujiwara, H. System-on-chip test scheduling with reconfigurable core wrappers. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 14*, 3 (2006), 305–309.

[76] Larsson, E., and Peng, Z. Test scheduling and scan-chain division under power constraint. In *Test Symposium, 2001. Proceedings. 10th Asian* (2001), IEEE, pp. 259–264.

[77] Le Cui, X., Shi, X. M., Li, H., and Lee, C. L. A shuffle frog-leaping algorithm for test scheduling of 2d/3d soc. In *Solid-State and Integrated Circuit Technology (ICSICT), 2012 IEEE 11th International Conference on* (2012), IEEE, pp. 1–3.

[78] Lee, J., and Touba, N. A. Lfsr-reseeding scheme achieving low-power dissipation during test. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 26*, 2 (2007), 396–401.

[79] Lee, K.-J., Chen, J.-J., and Huang, C.-H. Using a single input to support multiple scan chains. In *Proceedings of the 1998 IEEE/ACM international conference on Computer-aided design* (1998), ACM, pp. 74–78.

[80] Li, J., Liu, X., Zhang, Y., Hu, Y., Li, X., and Xu, Q. On capture power-aware test data compression for scan-based testing. In *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design* (2008), IEEE Press, pp. 67–72.

[81] Li, L., and Chakrabarty, K. Test data compression using dictionaries with fixed-length indices [soc testing]. In *VLSI Test Symposium, 2003. Proceedings. 21st* (April 2003), pp. 219–224.

[82] Li, L., Chakrabarty, K., and Touba, N. A. Test data compression using dictionaries with selective entries and fixed-length indices. *ACM Transactions on Design Automation of Electronic Systems (TODAES) 8*, 4 (2003), 470–490.

[83] Li, L., Choi, K., and Nan, H. Effective algorithm for integrating clock gating and power gating to reduce dynamic and active leakage power simultaneously. In *Quality Electronic Design (ISQED), 2011 12th International Symposium on* (March 2011), pp. 1–6.

[84] Liu, C., Veeraraghavan, K., and Iyengar, V. Thermal-aware test scheduling and hot spot temperature minimization for core-based systems. In *Defect and Fault Tolerance in VLSI Systems, 2005. DFT 2005. 20th IEEE International Symposium on* (Oct 2005), pp. 552–560.

[85] Liu, X., and Xu, Q. On x-variable filling and flipping for capture-power reduction in linear decompressor-based test compression environment. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 31*, 11 (Nov 2012), 1743–1753.

[86] Lu, S.-K., Chuang, H.-M., Lai, G.-Y., Lai, B.-T., and Huang, Y.-C. Efficient test pattern compression techniques based on complementary huffman coding. In *Testing and Diagnosis, 2009. ICTD 2009. IEEE Circuits and Systems International Conference on* (2009), IEEE, pp. 1–4.

[87] Macii, E., Bolzani, L., Calimera, A., Macii, A., and Poncino, M. Integrating clock gating and power gating for combined dynamic and leakage power optimization in digital cmos circuits. In *Digital System Design Architectures, Methods and Tools, 2008. DSD '08. 11th EUROMICRO Conference on* (Sept 2008), pp. 298–303.

[88] Marinissen, E. J., Goel, S. K., and Lousberg, M. Wrapper design for embedded core test. In *Test Conference, 2000. Proceedings. International* (2000), IEEE, pp. 911–920.

[89] Marinissen, E. J., Iyengar, V., and Chakrabarty, K. A set of benchmarks for modular testing of socs. In *Test Conference, 2002. Proceedings. International* (2002), IEEE, pp. 519–528.

[90] Marinissen, E. J., Iyengar, V., and Chakrabarty, K. Itc'02 soc test benchmarks web site, 2006.

[91] Marinissen, E. J., Kapur, R., and Zorian, Y. On using ieee p1500 sect for test plug-n-play. In *Test Conference, 2000. Proceedings. International* (2000), IEEE, pp. 770–777.

[92] Millican, S. K., and Saluja, K. K. Formulating optimal test scheduling problem with dynamic voltage and frequency scaling. In *Test Symposium (ATS), 2013 22nd Asian* (2013), IEEE, pp. 165–170.

[93] Mitra, S., and Kim, K. S. Xpand: An efficient test stimulus compression technique. *Computers, IEEE Transactions on 55*, 2 (2006), 163–173.

[94] Moghaddam, E., Rajski, J., Reddy, S., and Janicki, J. Low test data volume low power at-speed delay tests using clock-gating. In *Test Symposium (ATS), 2011 20th Asian* (Nov 2011), pp. 267–272.

[95] Muthyala, S. S., and Touba, N. A. Soc test compression scheme using sequential linear decompressors with retained free variables. In *VLSI Test Symposium (VTS), 2013 IEEE 31st* (2013), IEEE, pp. 1–6.

[96] Nicolici, N., and Al-Hashimi, B. M. Scan latch partitioning into multiple scan chains for power minimization in full scan sequential circuits. In *Design, Automation and Test in Europe Conference and Exhibition 2000. Proceedings* (2000), IEEE, pp. 715–722.

[97] Nourani, M., and Tehranipour, M. H. Rl-huffman encoding for test compression and power reduction in scan applications. *ACM Transactions on Design Automation of Electronic Systems (TODAES) 10*, 1 (2005), 91–115.

[98] Pouget, J., Larsson, E., and Peng, Z. Soc test time minimization under multiple constraints. In *Test Symposium, 2003. ATS 2003. 12th Asian* (2003), IEEE, pp. 312–317.

[99] Rajski, J., Tyszer, J., Kassab, M., and Mukherjee, N. Embedded deterministic test. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 23*, 5 (2004), 776–792.

[100] Reda, S., and Orailoglu, A. Reducing test application time through test data mutation encoding. In *Design, Automation and Test in Europe Conference and Exhibition, 2002. Proceedings* (2002), IEEE, pp. 387–393.

[101] Reddy, S. M., Miyase, K., Kajihara, S., and Pomeranz, I. On test data volume reduction for multiple scan chain designs. *ACM Transactions on Design Automation of Electronic Systems (TODAES) 8*, 4 (2003), 460–469.

[102] Rosinger, P., Al-Hashimi, B., and Chakrabarty, K. Rapid generation of thermal-safe test schedules. In *Design, Automation and Test in Europe, 2005. Proceedings* (March 2005), pp. 840–845 Vol. 2.

[103] Rosinger, P., Al-Hashimi, B., and Chakrabarty, K. Thermal-safe test scheduling for core-based system-on-chip integrated circuits. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 25*, 11 (Nov 2006), 2502–2512.

[104] Rosinger, P., Al-Hashimi, B., and Nicolici, N. Power constrained test scheduling using power profile manipulation, iscas 2001.

[105] Rosinger, P., Gonciari, P. T., Al-Hashimi, B., and Nicolici, N. Simultaneous reduction in volume of test data and power dissipation for systems-on-a-chip. *Electronics Letters 37*, 24 (2001), 1434–1436.

[106] Rosinger, P. M., Al-Hashimi, B. M., and Nicolici, N. Power profile manipulation: a new approach for reducing test application time under power constraints. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 21*, 10 (2002), 1217–1225.

[107] Ruan, X., and Katti, R. S. Data-independent pattern run-length compression for testing embedded cores in socs. *Computers, IEEE Transactions on 56*, 4 (2007), 545–556.

[108] Sabne, A., Tiwari, R., Shrivastava, A., Ravi, S., and Parekhji, R. A generic low power scan chain wrapper for designs using scan compression. In *VLSI Test Symposium (VTS), 2010 28th* (2010), IEEE, pp. 135–140.

[109] Samii, S., Larsson, E., Chakrabarty, K., and Peng, Z. Cycle-accurate test power modeling and its application to soc test scheduling. In *Test Conference, 2006. ITC'06. IEEE International* (2006), IEEE, pp. 1–10.

[110] Samii, S., Selkala, M., Larsson, E., Chakrabarty, K., and Peng, Z. Cycle-accurate test power modeling and its application to soc test architecture design and scheduling. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 27*, 5 (2008), 973–977.

[111] Sehgal, A., Goel, S. K., Marinissen, E. J., and Chakrabarty, K. Ieee p1500-compliant test wrapper design for hierarchical cores. In *Test Conference, 2004. Proceedings. ITC 2004. International* (Oct 2004), pp. 1203–1212.

[112] Sheshadri, V., Agrawal, V. D., and Agrawal, P. Optimal power-constrained soc test schedules with customizable clock rates. In *SOC Conference (SOCC), 2012 IEEE International* (2012), IEEE, pp. 271–276.

[113] Sheshadri, V., Agrawal, V. D., and Agrawal, P. Power-aware soc test optimization through dynamic voltage and frequency scaling. In *Very Large Scale Integration (VLSI-SoC), 2013 IFIP/IEEE 21st International Conference on* (2013), IEEE, pp. 102–107.

[114] Sismanoglou, P., and Nikolos, D. Enhancing dictionary based test data compression using the ate repeat instruction. In *Electronics, Circuits, and Systems (ICECS), 2013 IEEE 20th International Conference on* (2013), IEEE, pp. 401–404.

[115] Sismanoglou, P., and Nikolos, D. Input test data compression based on the reuse of parts of dictionary entries: Static and dynamic approaches. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 32*, 11 (Nov 2013), 1762–1775.

[116] Sismanoglou, P., and Nikolos, D. Test data compression based on reuse and bit-flipping of parts of dictionary entries. In *Design and Diagnostics of Electronic Circuits & Systems, 17th International Symposium on* (2014), IEEE, pp. 110–115.

[117] Sivanantham, S., Manuel, J. P., Sarathkumar, K., Mallick, P., and Perinbam, J. R. P. Reduction of test power and test data volume by power aware compression scheme. In *Advances in Computing and Communications (ICACC), 2012 International Conference on* (2012), IEEE, pp. 158–161.

[118] Stan, M. R., Skadron, K., Barcella, M., Huang, W., Sankaranarayanan, K., and Velusamy, S. Hotspot: a dynamic compact thermal model at the processor-architecture level. *Microelectronics Journal 34*, 12 (2003), 1153–1165.

[119] Synopsys Inc. *Design Vision "User Guide", Version 2002.05*, 2002.

[120] Synopsys Inc. *TetraMax ATPG Guide*, 2006.

[121] Tang, H., Reddy, S. M., and Pomeranz, I. On reducing test data volume and test application time for multiple scan chain designs. In *2013 IEEE International Test Conference (ITC)* (2003), IEEE Computer Society, pp. 1079–1079.

[122] Tehranipoor, M., Nourani, M., and Chakrabarty, K. Nine-coded compression technique for testing embedded cores in socs. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 13*, 6 (2005), 719–731.

[123] Tehranipour, M. H., Nourani, M., Arabi, K., and Afzali-Kusha, A. Mixed rl-huffman encoding for power reduction and data compression in scan test. In *Circuits and Systems, 2004. ISCAS'04. Proceedings of the 2004 International Symposium on* (2004), vol. 2, IEEE, pp. II–681.

[124] Touba, N. A. Survey of test vector compression techniques. *Design & Test of Computers, IEEE 23*, 4 (2006), 294–303.

[125] Volkerink, E. H., and Mitra, S. Efficient seed utilization for reseeding based compression [logic testing]. In *VLSI Test Symposium, 2003. Proceedings. 21st* (2003), IEEE, pp. 232–237.

[126] Wang, K., Huang, L., Zhou, C., and Pang, W. Particle Swarm Optimization for Traveling Salesman Problem. In *Proceedings of the Second International Conference on Machine Learning and Cybermetics* (2003), pp. 1583–1585.

[127] Wang, L.-T., Wen, X., Furukawa, H., Hsu, F.-S., Lin, S.-H., Tsai, S.-W., Abdel-Hafez, K. S., and Wu, S. Virtualscan: a new compressed scan technology for test cost reduction. In *Test Conference, 2004. Proceedings. ITC 2004. International* (2004), IEEE, pp. 916–925.

[128] Ward, S. I., Schattauer, C., and Touba, N. A. Using statistical transformations to improve compression for linear decompressors. In *Defect and Fault Tolerance in VLSI Systems, 2005. DFT 2005. 20th IEEE International Symposium on* (2005), IEEE, pp. 42–50.

[129] Wolff, F. G., and Papachristou, C. Multiscan-based test compression and hardware decompression using lz77. In *Test Conference, 2002. Proceedings. International* (2002), IEEE, pp. 331–339.

[130] Wurtenberger, A., Tautermann, C. S., and Hellebrand, S. Data compression for multiple scan chains using dictionaries with corrections. In *Test Conference, 2004. Proceedings. ITC 2004. International* (2004), IEEE, pp. 926–935.

[131] Wuu, J.-Y., Chen, T.-C., and Chang, Y.-W. Soc test scheduling using the b*-tree based floorplanning technique. In *Design Automation Conference, 2005. Proceedings of the ASP-DAC 2005. Asia and South Pacific* (2005), vol. 2, IEEE, pp. 1188–1191.

[132] Xia, Y., Chrzanowska-Jeske, M., Wang, B., and Jeske, M. Using a distributed rectangle bin-packing approach for core-based soc test scheduling with power constraints. In *Computer Aided Design, 2003. ICCAD-2003. International Conference on* (Nov 2003), pp. 100–105.

[133] Xu, Q., and Nicolici, N. Wrapper design for multifrequency ip cores. *Very Large Scale Integration (VLSI) Systems, IEEE Tran. on 13*, 6 (2005), 678–685.

[134] Xu, Q., and Nicolici, N. Multifrequency tam design for hierarchical socs. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 25*, 1 (2006), 181–196.

[135] Yang, B., Sanghani, A., Sarangi, S., and Liu, C. A clock-gating based capture power droop reduction methodology for at-speed scan testing. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2011* (March 2011), pp. 1–7.

[136] Yao, C., Saluja, K., and Ramanathan, P. Partition based soc test scheduling with thermal and power constraints under deep submicron technologies. In *Asian Test Symposium, 2009. ATS '09.* (Nov 2009), pp. 281–286.

[137] Yao, C., Saluja, K., and Ramanathan, P. Power and thermal constrained test scheduling under deep submicron technologies. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 30*, 2 (Feb 2011), 317–322.

[138] Yao, C., Saluja, K., and Ramanathan, P. Temperature dependent test scheduling for multi-core system-on-chip. In *Test Symposium (ATS), 2011 20th Asian* (Nov 2011), pp. 27–32.

[139] Yao, C., Saluja, K., and Ramanathan, P. Thermal-aware test scheduling using on-chip temperature sensors. In *VLSI Design (VLSI Design), 2011 24th International Conference on* (Jan 2011), pp. 376–381.

[140] Yilmaz, M., and Chakrabarty, K. Seed selection in lfsr-reseeding-based test compression for the detection of small-delay defects. In *Design, Automation & Test in Europe Conference & Exhibition, 2009. DATE'09.* (2009), IEEE, pp. 1488–1493.

[141] Yoneda, T., Masuda, K., and Fujiwara, H. Power-constrained test scheduling for multi-clock domain socs. In *Design, Automation and Test in Europe, Proceedings* (2006), vol. 1, pp. 1–6.

[142] Yoneda, T., Nakao, M., Inoue, I., Sato, Y., and Fujiwara, H. Temperature-variation-aware test pattern optimization. In *European Test Symposium (ETS), 2011 16th IEEE* (May 2011), pp. 214–214.

[143] Yu, T., Yoneda, T., Chakrabarty, K., and Fujiwara, H. Thermal-safe test access mechanism and wrapper co-optimization for system-on-chip. In *Asian Test Symposium, 2007. ATS '07. 16th* (Oct 2007), pp. 187–192.

[144] Yu, T., Yoneda, T., Chakrabarty, K., and Fujiwara, H. Test infrastructure design for core-based system-on-chip under cycle-accurate thermal constraints. In *Design Automation Conference, 2009. ASP-DAC 2009. Asia and South Pacific* (Jan 2009), pp. 793–798.

[145] Yu, Y., Peng, X., and Peng, Y. A test scheduling algorithm based on two-stage ga. In *Journal of Physics: Conference Series* (2006), vol. 48, IOP Publishing, p. 658.

[146] Yu, Y., Yang, Z., and Peng, X. Test data compression based on variable prefix dual-run-length code. In *Instrumentation and Measurement Technology Conference (I2MTC), 2012 IEEE International* (2012), IEEE, pp. 2537–2542.

[147] Zhao, D., Chandran, U., and Fujiwara, H. Shelf packing to the design and optimization of a power-aware multi-frequency wrapper architecture for modular ip cores. In *Design Automation Conf., Asia and South Pacific* (2007), pp. 714–719.

[148] Zhao, D., Huang, R., Yoneda, T., and Fujiwara, H. Power-aware multi-frequency heterogeneous soc test framework design with floor-ceiling packing. In *Circuits and Systems, IEEE Int. Symp. on* (2007), pp. 2942–2945.

[149] Zorian, Y., Marinissen, E. J., and Dey, S. Testing embedded-core based system chips. In *Test Conference, 1998. Proceedings., International* (1998), IEEE, pp. 130–143.

[150] Zou, W., Reddy, S. M., Pomeranz, I., and Huang, Y. Soc test scheduling using simulated annealing. In *VLSI Test Symposium, 2003. Proceedings. 21st* (April 2003), pp. 325–330.