Advanced Methods for Testing Multi-Core SoCs

A Dissertation

submitted to the designated
by the General Assembly of Special Composition
of the Department of Computer Science and Engineering
Examination Committee

by

Fotios I. Vartziotis

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

University of Ioannina
December 2016

Dedication

Στη γυναίκα μου, Μαρία και στα παιδιά μου, Γιάννη και Σωτήρη.
Στην μνήμη της αδερφής μου Ολίνας.

Acknowledgements

First of all, I would like to sincerely thank my supervisor, Prof. X. Kavousianos, for his guidance and support throughout this study.

Special thanks to Prof. K. Chakrabarty for providing useful feedback on my research and assisting me many times during my study.

I want to thank A. Jain and R. Parekhji from Texas Instruments, who provided me with data for industrial SoCs and useful comments upon my research work.

I would also like to thank Prof. G. Tsiatouhas and Prof. D. Nikolos for their support as members of the advisory committee.

Many thanks to my colleagues V. Tenentes, G. Sfikas and P. Georgiou for their support and the good times we had together.

I am deeply indebted to the grandparents of my children for their love and support. And finally I would like to thank my wife Maria for being on my side during the ups and downs of this effort.

Table of Contents

List of Figures

List of Tables

Extended Abstract in English

Vartziotis, Fotios, I., PhD
Department of Computer Science & Engineering, University of Ioannina, Greece
December, 2016
Advanced Methods for Testing Multi-Core SoCs
Supervisor: Xrysovalantis Kavousianos

We are in the dawn of a new era of "Internet of Things". Novel home and business "things" appear rapidly, and they promise to improve the quality of our lives and grow the world's economy. Looking into the "brain" of these "things", the design paradigm of the SoC is emerged. The SoC design paradigm is not a static description of guidelines and methodologies. In contrast, it evolves continuously trying to keep in pace with the ever-increased requirements of the new era.

Among the most important challenges facing the SoC design industry today are cost reduction and power management. The need for increased functionality and performance is translated to the design of complex SoCs that use advanced but costly process technologies and power management techniques. Thus, the industry is consistently looking for new, effective cost- and power- aware design solutions to apply at the end-products. Such solutions are the dynamic-voltage and frequency scaling, and the partition of the SoC into multiple voltage islands, which greatly affect the test process and test cost.

This research focuses in the development of an efficient, integrated and computational-friendly methodology able to minimize the test cost of moderate, large and very large multi-core, DVFS-based SoCs with voltage islands while power constraints are met. We introduce Time Division Multiplexing (TDM), a novel method for testing DVFS-based SoCs with multiple voltage islands. We present three power-aware test scheduling approaches able to exploit the advantages offered by TDM method at maximum level. Specifically, an integer-linear programming approach is proposed to deliver optimal test schedules for SoCs of moderate size, while a rectangle-packing/simulated-annealing approach is proposed to offer cost-effective solutions for large and very large SoCs. For early design-phase decisions as well as for cases with strict CPU-time limits, a greedy approach is proposed that offers fairly good results. Experimental results on two industrial SoCs show the superiority of the TDM-based approach against conventional approaches.

We extend the TDM method with Space Division Multiplexing (SDM) creating a new Space and Time Division Multiplexing (STDM) methodology that offers a highly efficient solution for multi-site test applications with a limited number of ATE channels. Space multiplexing permits the use of test access mechanisms (TAMs) that are narrower than the wrappers of the embedded cores in an SoC while time multiplexing exploits the available frequency bandwidth to parallelize test application, thereby minimizing the additional time overhead. Experiments on an industrial SoC show that STDM is very effective for narrow TAMs and it significantly increases the number of sites that can be tested in parallel. Thus, the test cost is minimized.

Furthermore, we propose a branch-&-bound (B-&-B) technique to optimize the test access mechanism for minimizing test-time when Time Division Multiplexing is used to schedule tests. The proposed technique exploits some unique properties of TDM to set a mathematically derived, strict, computationally simple and accurate bounding criterion that enables the B-&-B algorithm to rapidly prune more than 99% of the ineffective TAM configurations. In addition, a fast greedy test-scheduling approach is adopted to evaluate and identify the most effective configurations. The use of the greedy approach is justified by its high correlation (in evaluating TAM designs) with the very effective (but much slower) simulated annealing approach. For very large SoCs, a global TAM distribution approach is proposed, which optimally distributes the TAM lines into multiple SoC areas. To the best of our knowledge, this is the first TAM optimization approach that takes into account the unique characteristics of multi-V_{dd} SoCs and the benefits of the highly effective TDM approach, and offers a complete solution to the test-scheduling problem for multi-V_{dd} SoCs. Experiments on two industrial SoCs, as well as on two very large artificial SoCs show the benefits of the proposed method in both single-site and multi-site test applications.

Finally, we introduce a critical path−oriented thermal aware X−filling methodology for high un−modelled defect coverage. The thermal activity during testing can be considerably reduced by applying power-oriented filling of the unspecified bits of test vectors. However, traditional power-oriented X-fill methods in bibliography do not correlate the thermal activity with delay failures, and they consume all the unspecified bits to reduce the power dissipation at every region of the core. Therefore, they adversely affect the un-modelled defect coverage of the generated test vectors. The proposed method identifies the unspecified bits that are more critical for delay failures, and it fills them in such a way as to create a thermal-safe neighbourhood around the most critical regions of the core. For the rest of the unspecified bits a probabilistic model based on output deviations is adopted to increase the un-modelled defect coverage of the test vectors. Experimental results show that the proposed method offers a fair trade−off between critical paths delay and un−modelled defect coverage. Due to its nature, it may complement the formulation of the problem to be solved in many existing thermal and power aware techniques.

Εκτεταμενη Περιληψη στα Ελληνικα

Φώτιος Βαρτζιώτης του Ιωάννη και της Λαμπρινής, PhD
Τμήμα Μηχανικών Πληροφορικής, Πανεπιστήμιο Ιωαννίνων
Δεκέμβριος, 2016
Προχωρημένες μέθοδοι για τον έλεγχο ορθής λειτουργίας πολυπύρηνων συστημάτων σε ολοκληρωμένα
Επιβλέποντας: Χρυσοβαλάντης Καβουσιανός

Είμαστε στην αυγή της νέας εποχής του “Ιντερνέτ των πραγμάτων”. Νέα, καινοτόμα “πράγματα” για τον ιδιώτη και τις επιχειρήσεις εμφανίζονται με ολοένα αυξανόμενο ρυθμό και υπόσχονται να βελτιώσουν την ποιότητα της ζωής μας αλλά και την εικόνα της παγκόσμιας οικονομίας. Κοιτώντας βαθιά μέσα στον εγκέφαλό των “πραγμάτων” μπορούμε να δούμε επεξεργαστές που ακολουθούν τον σχεδιασμό του συστήματος σε ολοκληρωμένο (ΣσΟ). Το σχεδιαστικό παράδειγμα του ΣσΟ δεν περιλαμβάνει απλά μια σειρά προκαθορισμένων μεθοδολογιών και οδηγιών. Αντίθετα, εξελίσσεται συνεχώς ώστε να ανταποκριθεί με επιτυχία στις ολοένα αυξανόμενες απαιτήσεις της νέας εποχής.

Η μείωση του κόστους παραγωγής και η διαχείριση της καταναλισκόμενης ισχύος των ΣσΟ αποτελούν ίσως τις σημαντικότερες προκλήσεις που αντιμετωπίζει σήμερα η βιομηχανία κατασκευής τους. Η ανάγκη για αυξημένη λειτουργικότητα και απόδοση μεταφράζεται σε ανάγκη για τον σχεδιασμό πολύπλοκων ΣσΟ, τα οποία χρησιμοποιούν προηγμένες όσο και δαπανηρές τεχνολογίες επεξεργασίας και τεχνικές διαχείρισης ισχύος. Υπό το παραπάνω πρίσμα, η βιομηχανία αναζητά συνεχώς νέες, πιο αποτελεσματικές σχεδιαστικές λύσεις ως προς το κόστος και τη διαχείριση ισχύος για τα προϊόντα της. Μια από αυτές τις λύσεις είναι και η τεχνική της δυναμικής κλιμάκωσης της τάσης και της συχνότητας λειτουργίας (ΔΚΤΣ) ενός ΣσΟ, καθώς και η δημιουργία ξεχωριστών νησίδων τάσεων λειτουργίας (ΝΤΛ) εντός του ΣσΟ. Ωστόσο, η χρήση των παραπάνω τεχνικών επηρεάζει σημαντικά την διαδικασία ελέγχου ορθής λειτουργίας του ΣσΟ τόσο ως προς την διαδικασία όσο και ως προς το κόστος.

Η έρευνά μας εστιάζει στην ανάπτυξη ολοκληρωμένων, αποτελεσματικών και υπολογιστικά φιλικών μεθοδολογιών ικανών να ελαχιστοποιήσουν το κόστος ελέγχου ορθής λειτουργίας μικρών, μεγάλων και πολύ μεγάλων πολυπύρηνων ΣσΟ. Θεωρούμε ότι τα ΣσΟ εφαρμόζουν τεχνικές διαχείρισης ισχύος όπως η ΔΚΤΣ και οι ΝΤΛ και ότι ο έλεγχος ορθής λειτουργίας διεξάγεται υπό ένα σύνολο περιορισμών σχετικών με την κατανάλωση ισχύος. Στο παραπάνω πλαίσιο, εισάγουμε μια νέα, καινοτόμο μέθοδο ελέγχου ορθής λειτουργίας, η οποία στηρίζεται

στην τεχνική της χρονικής πολυπλεξίας (ΧΡΠ). Παρουσιάζουμε τρεις νέες τεχνικές για τον χρονοπρογραμματισμό της διαδικασίας ελέγχου ορθής λειτουργίας, οι οποίες αξιοποιούν στο μέγιστο τα οφέλη της ΧΡΠ. Συγκεκριμένα, προτείνουμε μια μέθοδο γραμμικού προγραμματισμού ακεραίων, η οποία είναι ικανή να προσφέρει βέλτιστες λύσεις για τον χρονοπρογραμματισμό του ελέγχου ορθής λειτουργίας σε μικρού και μεσαίου μεγέθους ΣσΟ. Επίσης, προτείνουμε μια νέα μέθοδο, η οποία στηριζόμενη στον αποτελεσματικό συνδυασμό των αλγορίθμων 'Simulated Annealing' και 'Rectangle Packing' (SA-RP), καταφέρνει να παράγει με μικρό υπολογιστικό κόστος υψηλής ποιότητας χρονοπρογραμματισμό για τον έλεγχο μεγάλων και πολύ μεγάλων ΣσΟ. Για την πρώιμη σχεδιαστική φάση του ελέγχου ορθής λειτουργίας καθώς και για περιπτώσεις όπου η διαθέσιμη υπολογιστική ισχύς είναι περιορισμένη, προτείνουμε μια "άπληστη" μέθοδο που οδηγεί σε πολύ καλές λύσεις για τον χρονοπρογραμματισμό της διαδικασίας ελέγχου. Πειραματικά αποτελέσματα αποδεικνύουν την ανωτερότητα των προτεινόμενων μεθόδων ΧΡΠ, έναντι των συμβατικών.

Στα πλαίσια της έρευνάς μας, έχουμε επεκτείνει τη μέθοδο ΧΡΠ με νέους, πρόσθετους μηχανισμούς χωρικής πολυπλεξίας (ΧΩΠ). Δημιουργήσαμε μια νέα μέθοδο χωρικής και χρονικής πολυπλεξίας (ΧΧΠ), ικανή να βελτιώσει σημαντικά την εκτέλεση παράλληλων διαδικασιών ελέγχου (multi-site test), ιδιαίτερα σε περιπτώσεις όπου ο αριθμός των διαθέσιμων καναλιών ελέγχου είναι περιορισμένος, λόγω κόστους. Η ΧΩΠ επιτρέπει να υλοποιούμε, σε ένα μηχανισμό πρόσβασης για έλεγχο (ΜΠγΕ), διαύλους με μικρότερο εύρος από το εύρος που χρησιμοποιεί η θύρα ενός περιβλήματος πυρήνα βασισμένου στο "πρωτόκολλο 1500" για την παράλληλη φόρτωση δεδομένων ελέγχου στους πυρήνες του ΣσΟ. Παράλληλα, η ΧΡΠ αναλαμβάνει να εκμεταλλευτεί στο έπακρο το διαθέσιμο εύρος των καναλιών ελέγχου ώστε να περιορίσει τυχόν χρονικές αυξήσεις στον χρονοπρογραμματισμό λόγω του περιορισμένου αριθμού των διαθέσιμων καναλιών ελέγχου ανά πυρήνα και γενικότερα ανά ΣσΟ. Πειραματικά αποτελέσματα δείχνουν ότι η μέθοδος ΧΧΠ είναι ιδιαίτερα αποδοτική για ΜΠγΕ με μικρό αριθμό καναλιών και αυξάνει σημαντικά τον αριθμό των ΣσΟ που μπορούν να ελεγχθούν παράλληλα. Συνεπώς περιορίζει το κόστος ελέγχου.

Προτείνουμε μια τεχνική "Διαίρει και Βασίλευε" (ΔΒ), η οποία έχει ως στόχο να βελτιστοποιήσει τον ΜΠγΕ σε διαδικασίες ελέγχου ορθής λειτουργίας, οι οποίες χρονοπρογραμματίζονται με την μέθοδο της ΧΡΠ. Η προτεινόμενη τεχνική εκμεταλλεύεται κάποιες μοναδικές ιδιότητες της μεθόδου ΧΡΠ ώστε να υπολογίσει ένα μαθηματικά αποδεδειγμένο, υπολογιστικά απλό αλλά ακριβές κριτήριο αποκλεισμού λύσεων, το οποίο είναι ικανό να απορρίψει άμεσα περισσότερους από το 99% των πιθανών αναποτελεσματικών σχεδιασμών του ΜΠγΕ. Ακόμη, μια γρήγορη "άπληστη" μέθοδος αναλαμβάνει να αξιολογήσει και να αναγνωρίσει του βέλτιστους υπαρκτούς σχεδιασμούς του ΜΠγΕ. Η χρήση της "άπληστης" μεθόδου δικαιολογείται από την υψηλή συσχέτιση που παρουσιάζουν τα αποτελέσματά της με τα αποτελέσματα της πολύ αποδοτικής μεθόδου SA-RP, η οποία δεν μπορεί να χρησιμοποιηθεί για λόγους ταχύτητας. Επιπρόσθετα, για την βελτιστοποίηση του ΜΠγΕ σε πολύ μεγάλα ΣσΟ, προτείνουμε μια νέα, ολοκληρωμένη μέθοδο κατανομής καναλιών ελέγχου, η οποία κατανέμει αποδοτικά τα κανάλια ελέγχου σε ένα σύνολο περιοχών του ΣσΟ. Με βάση τα όσα γνωρίζουμε η προτεινόμενη μέθοδος ΔΒ είναι η πρώτη που προσεγγίζει το θέμα της βελτιστοποίησης του ΜΠγΕ λαμβάνοντας υπόψη τα μοναδικά χαρακτηριστικά των ΣσΟ πολλαπλών τάσεων και αξιοποιώντας τα οφέλη της πολύ αποδοτικής μεθόδου ΧΡΠ προσφέροντας μια συνολική λύση στο πρόβλημα του χρονοπρογραμματισμού των ΣσΟ πολλαπλών τάσεων. Πειραματικά αποτελέσματα σε 2 ΣσΟ της βιομηχανίας καθώς και σε 2 πολύ μεγάλα τεχνητά ΣσΟ αποδεικνύουν τα οφέλη της προτεινόμενης μεθόδου για απλές και παράλληλες διαδικασίες ελέγχου.

Τέλος, εισάγουμε μια νέα μέθοδο για την θερμική προστασία των κρίσιμων μονοπατιών σε ένα πυρήνα ενός ΣσΟ. Η προτεινόμενη μέθοδος επιτυγχάνει το στόχο μέσω της επιλεκτικής συμπλήρωσης των αδιάφορων τιμών 'Χ' των διανυσμάτων ελέγχου (ΔΕ), η οποία ωστόσο πραγματοποιείται με τέτοιο τρόπο ώστε τα παραγόμενα ΔΕ να αποκτούν μεγαλύτερη ικανότητα αναγνώρισης μη μοντελοποιημένων σφαλμάτων υλικού (ΜΜΣΥ). Η αύξηση της θερμοκρασίας σε έναν πυρήνα μπορεί να περιοριστεί συνολικά μέσω της επιλεκτικής συμπλήρωσης των 'Χ' των ΔΕ, με τιμές που περιορίζουν την κατανάλωση της ισχύος. Ωστόσο, οι παραδοσιακές μέθοδοι επιλεκτικής συμπλήρωσης 'Χ' που συναντάμε στη βιβλιογραφία δεν συσχετίζουν την παραγόμενη θερμότητα σε μια περιοχή του πυρήνα με "σφάλματα λόγω καθυστέρησης", και χρησιμοποιούν όλες τις αδιάφορες τιμές ενός ΔΕ για να μειώσουν την κατανάλωση της ισχύος σε κάθε περιοχή του πυρήνα. Έτσι, μειώνουν την ικανότητα ενός ΔΕ να αναγνωρίζει ΜΜΣΥ. Η προτεινόμενη μέθοδος αναγνωρίζει αδιάφορες τιμές των ΔΕ, οι οποίες μπορεί να αποδειχθούν κρίσιμες για την εμφάνιση σφαλμάτων λόγω καθυστέρησης, και τις συμπληρώνει με τιμές, οι οποίες μπορούν να περιορίσουν τον ρυθμό αύξησης της θερμοκρασίας στην περιοχή των κρίσιμων μονοπατιών του πυρήνα αλλά και στις γειτονικές περιοχές. Οι υπόλοιπες αδιάφορες τιμές του ΔΕ λαμβάνουν τιμές με βάση τα αποτελέσματα ενός μοντέλου "αποκλίσεων εξόδου", ώστε να βελτιώνεται η ικανότητα αναγνώρισης ΜΜΣΥ του ΔΕ. Πειραματικά αποτελέσματα δείχνουν ότι η προτεινόμενη μέθοδος επιτυγχάνει μια αποδοτική ανταλλαγή μεταξύ της καθυστέρησης ενός μονοπατιού και της ικανότητας αναγνώρισης ΜΜΣΥ των ΔΕ. Λόγω της φύσης της, η παραπάνω μέθοδος μπορεί να χρησιμοποιηθεί συμπληρωματικά με οποιαδήποτε μέθοδο διαχείρισης ισχύος ή / και παραγόμενης θερμότητας.

Chapter 1

Introduction

1.1 System On Chip

In the modern era, a growing number of physical objects, at an unprecedented rate, are equipped with “edge” devices that enhance them with computational intelligence and Internet connection abilities, realizing the vision of Internet of Things (IoT) [1] - [3]. Nowadays, thermostats that continually look for innovative ways to reduce your energy bill, cars that service and fix themselves whenever there is a problem, shoes that direct you around town, toothbrushes that team up with your dentist to improve your oral hygiene and, umbrellas that check the local weather to ensure that you never get wet are advertised in the media.

Generally speaking, IoT refers to the transformation of everyday objects from being traditional to “smart”. It is the result of technological progress in many parallel and often overlapping fields, including those of embedded systems, ubiquitous and pervasive computing, mobile telephony, telemetry and machine-to-machine communication, wireless sensor networks, mobile computing, and computer networking. IoT is opening

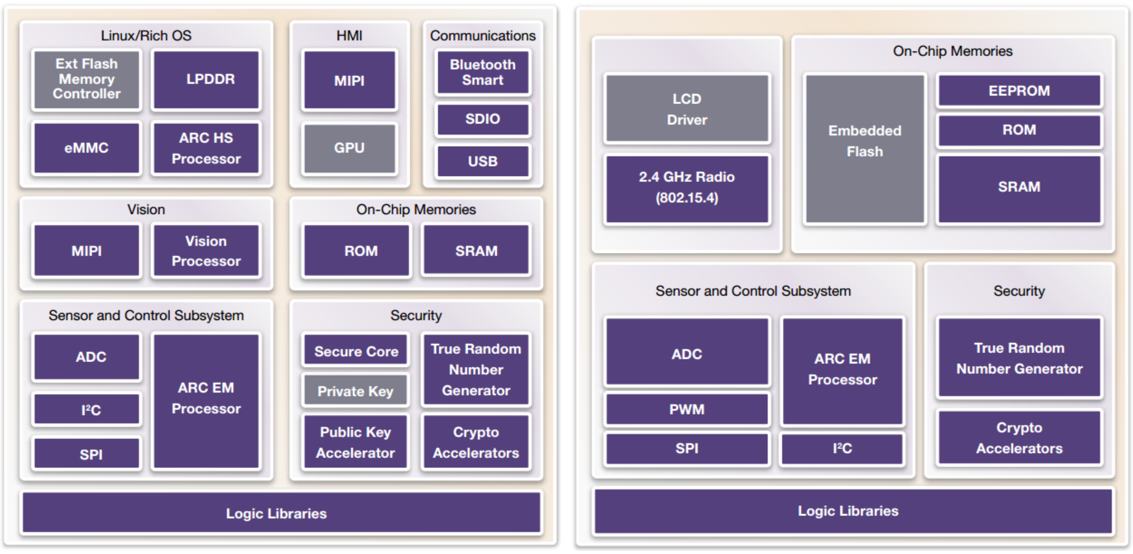
Figure 1.1: Key IoT applications [4].

tremendous opportunities for a large number of novel applications that promise to improve the quality of our lives and grow the world's economy. These applications include home support, healthcare, inventory and product management, workplace, security and surveillance, transportation, industrial automation, emergency response and environmental monitoring.

The so-called "Things" or "edge" devices [4] are a fusion of processing units, sensors, actuators and software applications. They are able to sense physical phenomena, translate them into a stream of information data, communicate them to human beings or other devices and trigger actions on the physical world. According to market reports [5], more than 26 billion of such devices are expected to be "connected" to our world by 2020.

Looking deeply into the hardware of these devices, the design paradigm of the System on Chip (SoC) emerges [6], that through a great variety of embedded cores / Intellectual Property (IP)s tries to serve the heterogeneous needs of the IoT. Application processors for feature rich wearable devices, microcontrollers for low-end applications and smart analog devices with power management abilities constitute a mosaic of IoT system architectures [4].

Conventional gate-based or cell-based design methodologies are no longer sufficient. The shift toward very deep submicron technology has enabled Integrated Circuit (IC) designers to develop SoCs, where an entire system is implemented on a chip. To meet the SoC design economics, increase the productivity and decrease time-to-market, the use of out-of-house and / or previously in-house designed modules has become common practice in SoC design. Such modules are referred to as embedded cores or IPs. It is predicted that in the near future, embedded cores, of which 40 to 60% will be from external sources [8], will populate 90% of a chip. Actually, a few large companies such as Intel, Toshiba, and IBM are able for in-house manufacturing while most of the semiconductor companies are fabless and outsource their manufacturing to a silicon foundry such as TSMC, UMC, or IBM. This is explained by the fact that the annual sales of an IC enterprise need to exceed



a) Example high-end SoC block application

b) Example Low-end MCU SoC Block Diagram

Figure 1.2: Example architectures of SoCs addressed to IoT [4].

$10B to justify the investments required at the 45 nm process, and this figure continues to climb as processes advance [9]. Typical examples of SoCs that are used for High- and Low- end IoT applications are shown in Fig. 1.2 [4]. As it is illustrated, each SoC architecture may consist of a variety of embedded cores / IPs. Such IPs can be microcontroller, microprocessor or Digital Signal Processor (DSP) cores (also, there are multiprocessor SoCs, MPSoCs, that may have more than one processor core), memory blocks including a selection of ROM, RAM, EEPROM and flash memory, timing sources including oscillators and phase-locked loops, peripherals including counter-timers, real-time timers and power-on reset generators, external interfaces, including industry standards such as USB, FireWire, Ethernet, USART, SPI, analog interfaces including ADCs and DACs, voltage regulators and power management circuits, buses, either proprietary or industry-standard, connecting these blocks.

1.2 The demand for low power operation

IoT and mobile devices require sophisticated computational abilities, advanced color displays, wireless technology and increased storage. The levels of integration enabled by SoC design and the continued advances in process and manufacturing technology made possible their realization. However, the continuing trend in applications for ever increasing functionality, performance and integration within SoCs have inadvertently pushed the power profiles of such systems beyond acceptable power density limits [68]. As an indi-

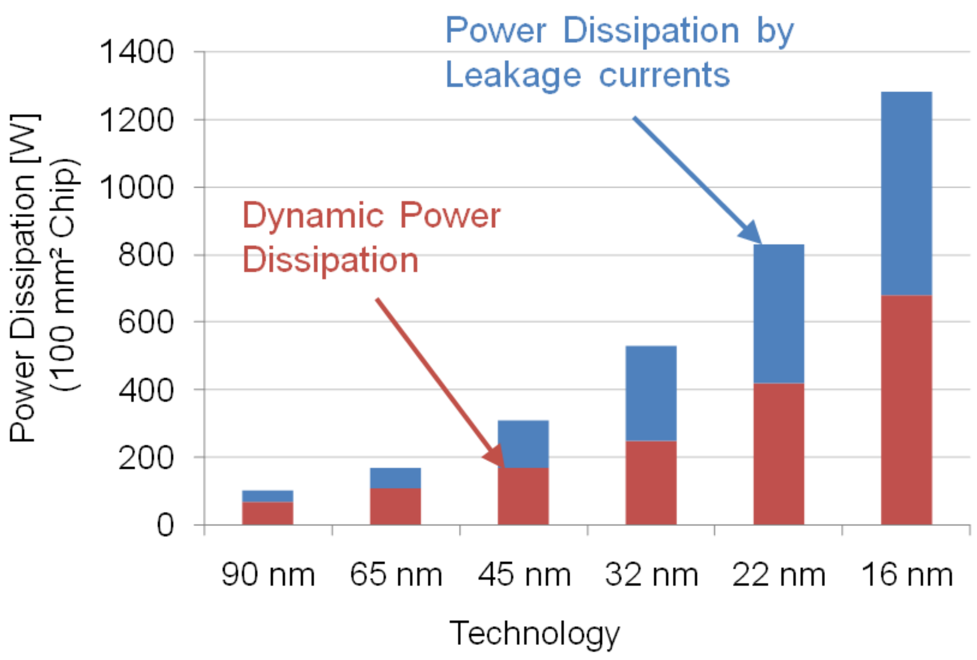


Figure 1.3: Power dissipation per process technology [71].

cation of the severity of the problem, we refer the Itanium2 from Intel, that consumes almost 130 Watts [54]. Fig. 1.3 [71] shows a plot of the power density of microprocessor chips for various technology generations.

The trend depicted in Fig. 1.3 is worrisome [7] and has forced power to become an important challenge for higher levels of integration due to further device scaling. Higher power density has a negative impact on the performance as well as reliability of the chip [72]. Moreover, dissipating more heat has a large impact on the packaging, the heat sinks and the cooling environment used in an SoC, especially in terms of cost [73]. Thus, for almost every system architecture, reducing overall power consumption and localized power density is essential, in order to maintain the feasibility of future applications.

An additional obvious driver for low-power systems is the tremendous increase in the demand for battery-powered IoT and mobile devices. Limited battery life has significant impact in the utility / value of such devices. Unfortunately, battery technology seems unable to keep pace with the requirements for increasing complexity, functionality and performance of the systems they power [74]. While the advances in CMOS technology have seen a doubling in transistor density roughly every 18 months, the equivalent advancement in battery technology is greater than every five years [7]. Fig. 1.4 illustrates the widening gap between the trends of processor's power consumption and the improvement in battery power capacity. As these systems have fixed throughput requirements, a careful trade-off between power and performance is the key to extend battery life.

Today, energy issues have become a major topic in the public debate. As the IoT vision calls for the deployment of billions new devices [5], it is expected that the increase of the electricity consumption of the ICT infrastructure (datacenters and network equipment)

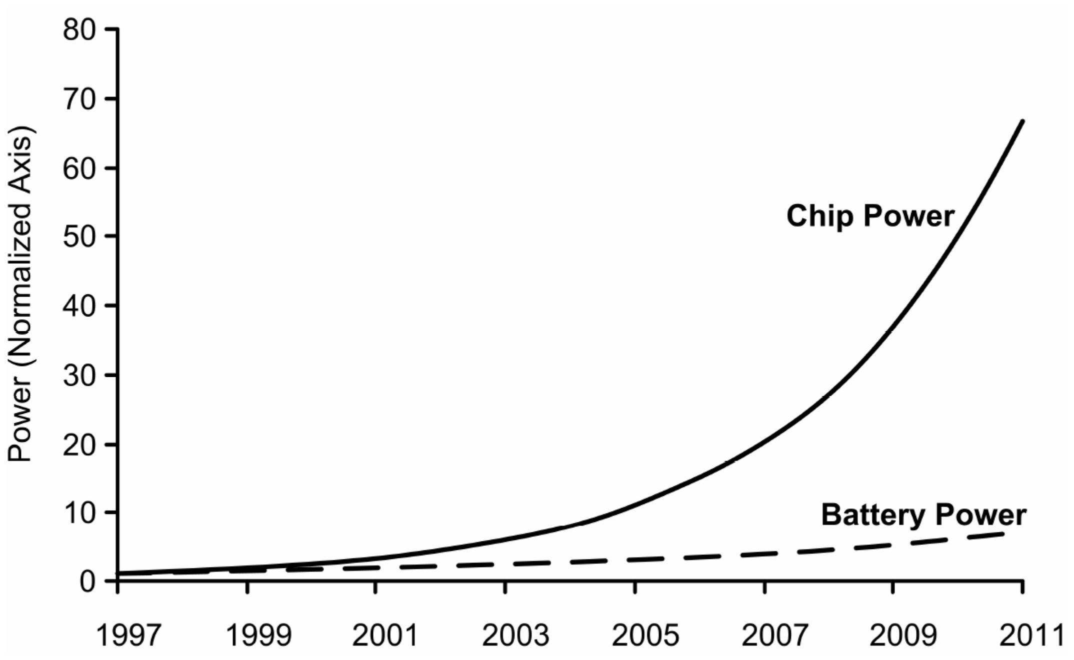


Figure 1.4: Trends in battery maximum power and chip power [74].

to handle the so-called "Big Data", the significant carbon footprint from the high-volume production and the increased level of electronic waste will trigger environmental concerns [6]. The so-called "green" initiative, especially in Europe, has already placed great pressure on manufacturers of SoCs to reduce power consumption as much as possible. So, from large desktop units to mobile and IoT devices, the drive today is towards "green" SoCs and Design-for-the-Environment (DFE) [6].

The above discussion shows that the pursuit of efficient methods for reduction in power consumption has moved to the forefront of the SoC design challenge. Today, satisfying the power budget is one of the most important design goals in SoC design. Hence, designers continuously seek effective ways to "arm" their SoCs with effective power management techniques ready to account for mobility, SoC complexity and process technology. The impact of low-power-oriented design on the SoC architecture is shown in Fig.1.5 [69] and is more than encouraging, since it is predicted that in year 2026, most of the SoCs will consume up to 8 watts.

The need for low power consumption is not confined to the functional operation of devices only. Power issues can greatly affect the Very-Large-Scale Integration (VLSI) testing process too. In fact, test designers face an even tougher challenge than SoC designers. A device under test (DUT) lacks package support. In addition, typical power management schemes are disabled and device activity may be higher during testing. The community of VLSI testing studies for years the impact of power consumption in the test process and has developed numerous techniques able to limit down power-related implications.

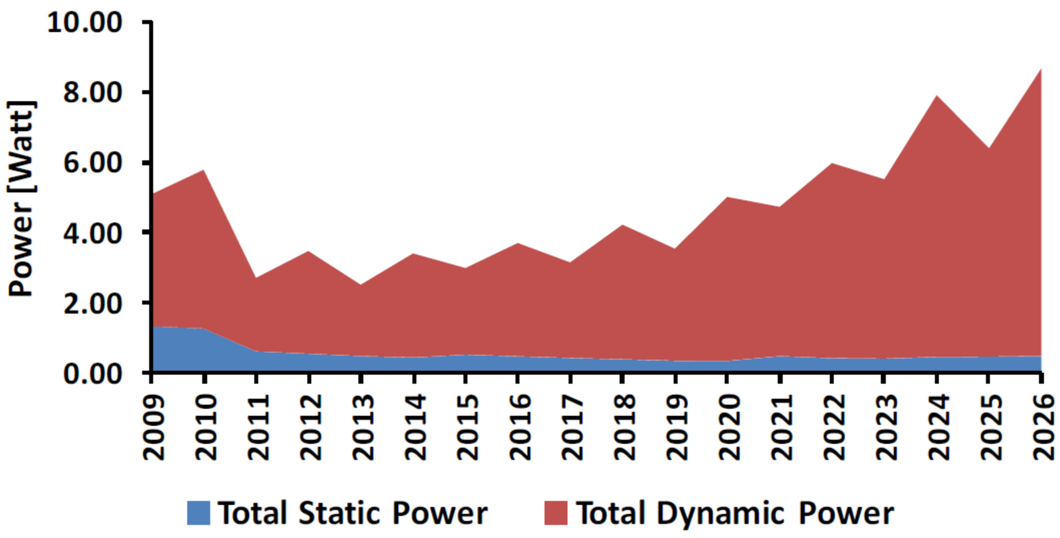


Figure 1.5: Impact of Low-Power Design Technology on SoC Consumer Portable Power Consumption [69].

1.3 Power consumption sources

Power consumption within a device can be analysed into two basic components, dynamic power consumption P_D based on the switching activity and the static power consumption based on leakage P_L, namely

$$P_{Total} = P_D + P_L \tag{1.1}$$

This is also illustrated in Fig. 1.3 and 1.5. The dynamic power consumption, which is traditionally the most important component, can be further analysed into the switch power P_{sw} due to the charging and discharging of capacitive loads driven by the circuit (including net capacitance and input loads), and short circuit power P_{sc} occurring momentarily during switching when pairs of PMOS and NMOS transistors are conducting simultaneously (Fig. 1.6). The leakage power P_L can also be broken down into a number of key contributors. One is the current flowing through the reverse biased diode formed between the diffusion regions and the substrate (I_{diode}). Another is the current flowing through transistors that are not conducting, tunneling through the gate oxide ($I_{subthreshold}$) (Fig. 1.6).

Lately, the static power dissipation has become significant for a number of reasons. First, as clearly shown in Fig. 1.3, one of the negative effects of the advance in deep sub - micron process technology is the relative increase in power due to leakage currents. Moreover, when V_{th} is lowered to succeed better speeds in chip, the static power is increased exponentially due to the exponential relation between subthreshold leakage current and V_{th} [11]. For example, leakage current within a 130nm process with a 0.7V threshold gives approximately 10-20 pA per transistor. When the threshold is reduced to 0.3V in the same process, the leakage current goes to 10-20 nA per transistor [7]. The problem becomes even worse when the temperature of the chip increases. Higher temperature

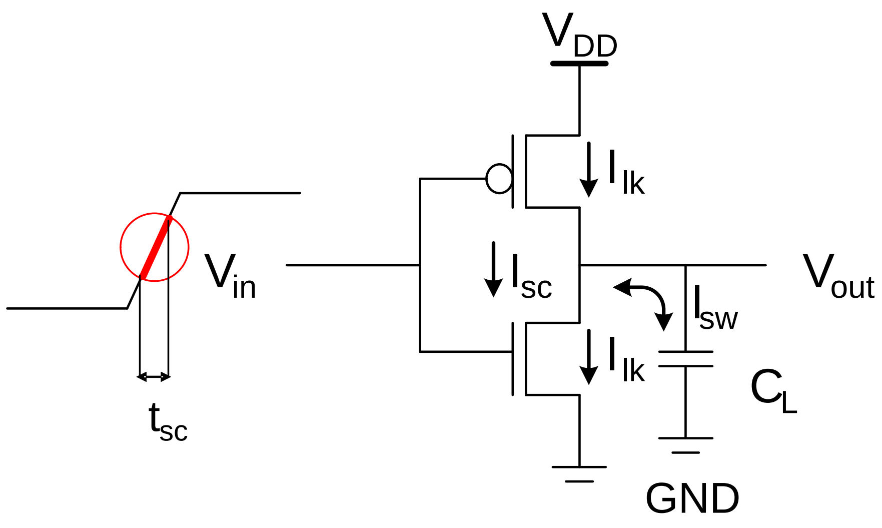


Figure 1.6: Power Consumption in a simple inverter.

increases the subthreshold leakage and, in the worst case, the power budget can be exceeded [7]. Also, due to the reduction in gate-oxide thickness for sub-100nm CMOS, gate leakage is also contributing significantly to the overall leakage power [12]. However, for future technology nodes, it is expected that high-k dielectrics would mitigate the increase of gate leakage as it appears to be the only effective way of reducing gate leakage [13].

1.4 Low power design techniques

Low power design methodologies can be implemented at different stages of the design process. The following subsections briefly describe techniques from dynamic and leakage power reduction perspectives.

1.4.1 Dynamic-power-oriented reduction techniques

The following high-level equations characterize the key factors in the dynamic power dissipation on a chip [9][127][7]:

$$P_{sw} = A \cdot C \cdot V^2 \cdot F \tag{1.2}$$

$$P_{sc} = A \cdot \frac{B}{12} \cdot (V - 2 \cdot V_{th})^3 \cdot F \cdot T \tag{1.3}$$

where A is the switching activity, C is the total load capacitance, V the supply voltage, F the target frequency, B the gain factor, T the rise / fall time of gate inputs and V_{th}

the voltage threshold. Dynamic power can be minimized by reducing any of the terms in Equations 1.2 and 1.3. Thus, designers have devised methods of reducing one or more of these parameters. Most of the efforts have concentrated on power supply and switched capacitance C_{eff} reduction, namely the product of the activity A and the total load capacitance C. The proposed techniques can be implemented at the architecture, logic design and circuit design levels. In the following lines we shortly describe methods that minimize the dynamic power consumption.

A significant portion of the dynamic power in a chip can be due to the distribution network of the clock [10]. The most common way to reduce this power is to turn off the clock when it is not required. This method is called clock gating. It was first proposed in [14] and studied in greater detail in [15] and [16]. The operand isolation method prevents sections of the circuitry from accepting changes in their inputs unless they are expected to respond to those changes [17]. An example technique for automating operand isolation has been proposed in [18]. The gate-level transformation method is a logic optimization technique where a small group of gates are replaced by a logically-equivalent set in order to reduce dynamic power [19]. Path balancing is used to reduce glitch power. Glitch power is dissipated when the inputs to a gate do not arrive at the same time causing unwanted transitions of the output signal before the final value is reached [22]. The path balancing technique manages to avoid the aforementioned transitions by equalizing the delays of paths that converge to the same gate [20][21]. The gate sizing method determines the device widths for each gate. The basic rule is to use smaller transistors that satisfy the delay constraints. Thus, to reduce dynamic power, gates with higher toggle rates must be made as small as possible. A polynomial formulation using Elmore delay models is used in traditional gate sizing approaches. Heuristic-based greedy approaches [24][25] can be used to solve such a polynomial problem. The transistor sizing is similar to gate sizing with the exception that in gate sizing all the transistors are sized together with the same factor while in transistor sizing each transistor is sized individually. Optimizations by sizing individual transistors have been explored in [26][27]. Finally, the Voltage scaling techniques are closely related to our research work and they are discussed in further detail in the following subsection.

1.4.2 Voltage Scaling

Voltage scaling exploits the quadratic relationship between V_{DD} and power consumption due to switching activity (Equation 1.2). Reducing the supply voltage is perhaps the most attractive approach for dynamic power reduction. Supply voltages can be reduced dynamically or in a static manner. Dynamic Voltage Scaling (DVS) reduces supply voltage during circuit operation based on the throughput requirements of the system. Multiple supply voltage (multi-V_{DD}) implementation is a static technique that pre-assigns different supply voltages to different IP blocks or even gates based on the throughput requirement.

Dynamic Voltage and Frequency Scaling (DVFS) is an efficient power management technique that offers an effective trade-off between power consumption and system per-

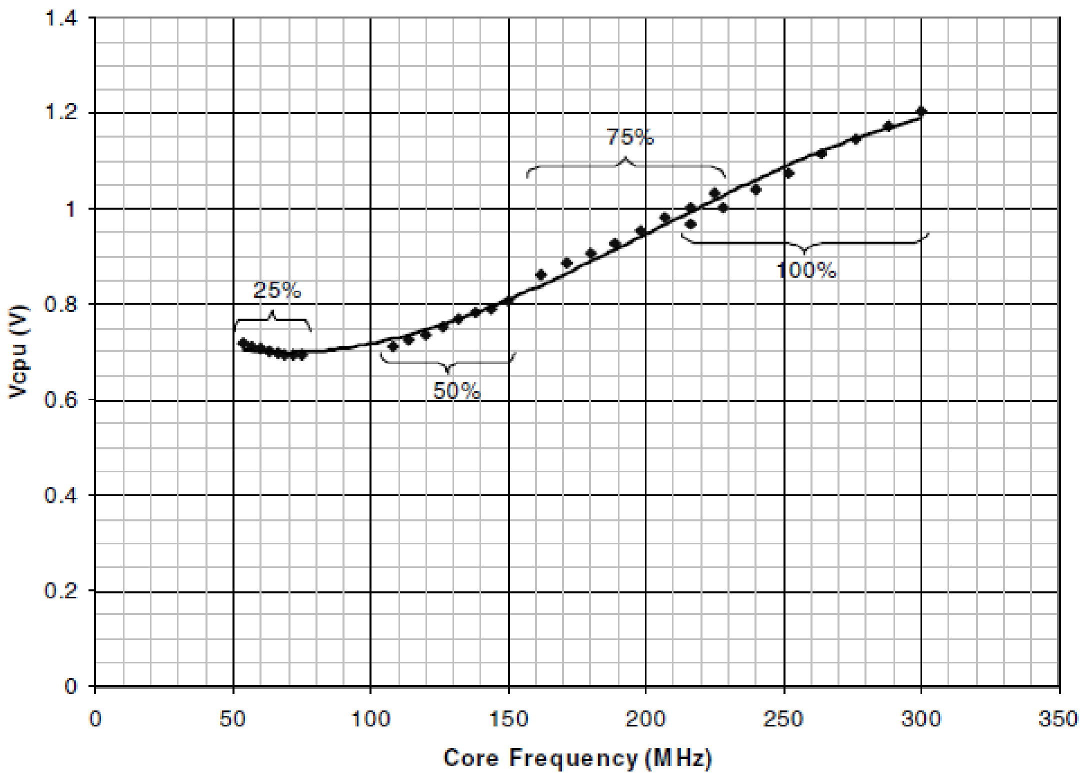


Figure 1.7: Core voltage vs frequency [47].

formance as it adaptively adjusts the power supply-voltage and the frequency depending on the workload of the SoC [44], [45]. To compensate for the lost performance, one can increase the degree of pipelining and parallelism in the SoC, reduce the V_{th} to achieve higher performance, assign lower V_{DD} values only to non-critical paths [46]. Fig. 1.7 shows the affect of the voltage scaling on the speed performance of IEM926 test SoC [47]. As it is illustrated, when the supply voltage has a high value ($V_{dd} = 1.2V$), the throughput of the test SoC becomes maximum (300MHz), while for lower values of supply voltage ($V_{dd} = 0.7V$), the system performance decreases almost by a factor of three (100MHz).

DVFS has been implemented in several state-of-the-art processors [52] - [176] that can be found in a wide range of portable devices. Typically, there are three operating modes: computation-intensive mode, low-speed deadline-driven mode and idle mode [57]. Compute-intensive tasks, such as MPEG video and audio decompression, demand maximum throughput. In contrast, tasks such as text processing and data entry require only a fraction of the system throughput. Finishing these tasks early has no benefits and thus dynamic voltage and frequency scaling can be applied at the cost of reduced speed. DVFS utilizes this slack time by lowering the supply voltage as well as the clock frequency to achieve quadratic savings in energy. Fig. 1.8, combined with Fig. 1.7, depicts the power and energy savings that can be achieved, when voltage scaling techniques are applied to the IEM926 test SoC.

Multi-V_{DD} design is a static voltage scaling approach for reducing dynamic power

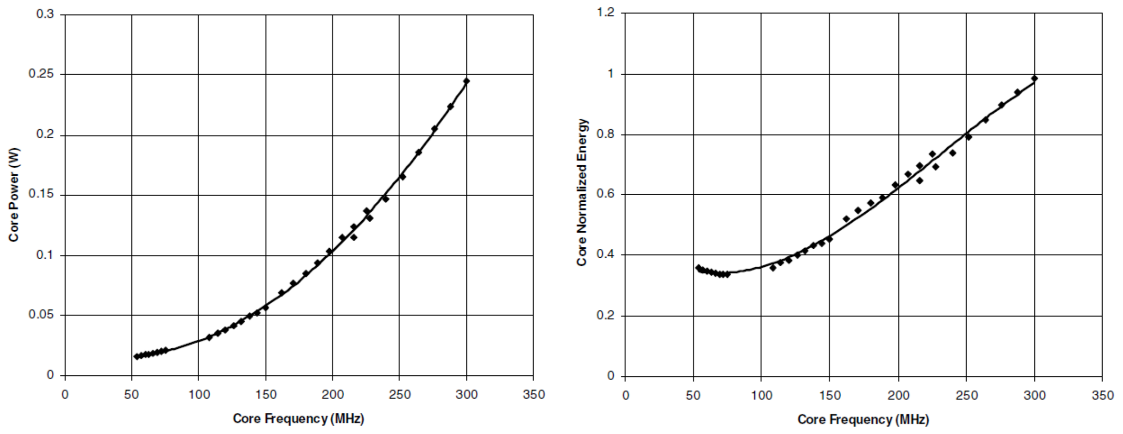


Figure 1.8: Power and energy consumption at different frequency and voltage levels[47].

consumption. A separate supply voltage can be assigned on a gate-by-gate basis (Dual-V_{DD}) or to every module of the design (i.e., the voltage island technique). In Dual-V_{DD} approach [48], gates that are on the critical path of a block are assigned to the high V_{DD} while gates on the non-critical path are assigned to low V_{DD}. When gates operating at low V_{DD} are fan-ins to gates operating at high V_{DD}, level converters are required in order to avoid short-circuit power. As level converters contribute to additional power consumption, minimizing the number of converters is an additional design issue [49]. Clustered Voltage Scaling (CVS) [50] or Extended Clustered Voltage Scaling (ECVS) [51] can be used to assign the required V_{DD} to the targeted gates. In CVS, the cells driven by each supply voltage are clustered such that level conversion is only required at the output flip-flops. ECVS removes restrictions on level converter assignment by allowing level conversion anywhere, and the supply voltage assignment to the gates is much more flexible. ECVS method is more complicated than CVS, however, it provides greater dynamic power savings.

The concept of voltage islands (also known as power islands) arose to allow areas of a single chip to operate at voltage levels and frequencies independent from one another [58]. Fig. 1.9 shows an example SoC with 16 IP blocks before and after creating voltage islands. In Fig. 1.9(a), all the blocks are assigned to a high V_{DD} level, while in Fig. 1.9(b), the performance-critical blocks are assigned to a high V_{DD} level, and the other blocks are assigned to lower V_{DD} levels, depending on their speed requirements. Then,the overall power dissipation of the design in Fig. 1.9(b) can be far less than that in Fig. 1.9(a) due to reduced dynamic power. Available white-space can be used for placing level shifters, routing and the allocation of decoupling capacitance to reduce power supply noise.

Voltage island design can be implemented at either the floorplan / placement stage [59][60] or post- floorplan / placement stage [61][62]. The major impact of using multiple supply voltages is the need to treat the supply voltage as another design constraint. In voltage island design, IP blocks assigned to unique islands require proper delivery of the specific supply voltage. If the supply voltage is generated off-chip, proximity to its corresponding supply voltage pins must be ensured. On the other hand, on-chip supply

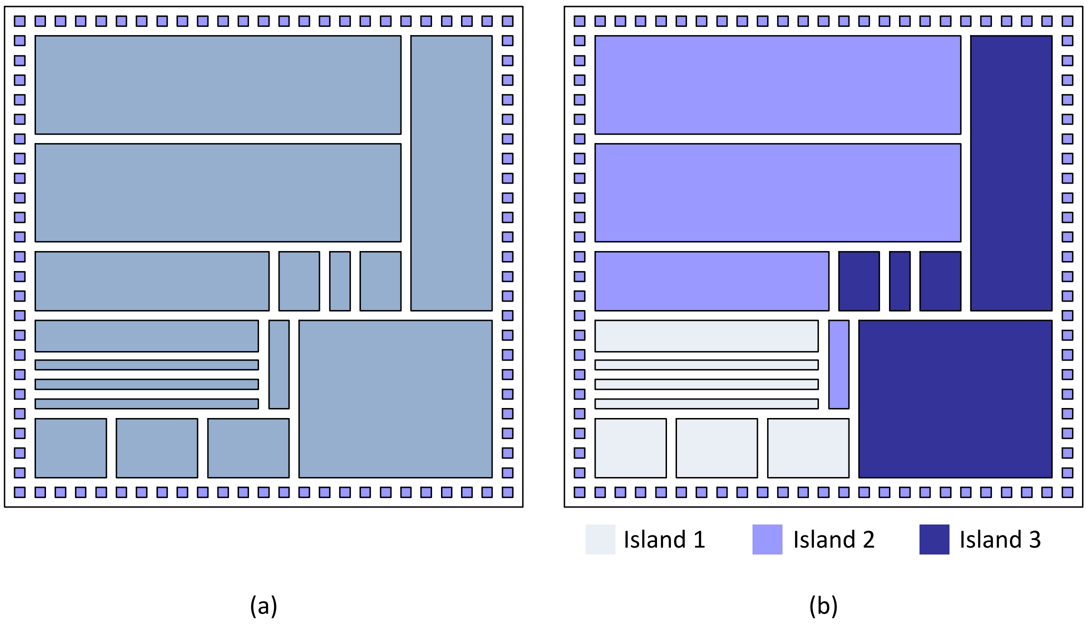


Figure 1.9: SoC with (a) single voltage island and (b) many voltage islands.

voltage generation requires extra routing to each island. Similar to CVS and ECVS, communication between islands require level converters. Clock-tree generation must take into account buffers residing in different islands. Designers must also account for coupling noise between adjacent lines driven by high and low voltages arising from the different V_{DD} values.

Further reductions in power consumption are achieved when voltage islands are combined with DVFS [45][64]. For example, many tasks may not perform any floating-point operations in CPU and so the floating-point unit within an SoC could be run at a minimal voltage and frequency to conserve power. In addition, there are times when simply turning off the power to a unit is not an option, since memory contents or other state information must be preserved. Voltage islands with DVFS provide the flexibility to reduce the voltage and/or frequency in such cases to minimal levels. Fig. 1.10 shows an example of voltage islands within an SoC. Various methods have been proposed to address design challenges in such systems consisting of multiple voltage islands [44][65]-[67]. Fig. 1.10 shows an example SoC with 5 voltage islands able to perform DVFS using two different V_{DD} levels.

1.4.3 Leakage-power-oriented reduction techniques

In general, circuit blocks are constantly turning on and off, depending on the computation requirements of an application. Due to this "bursty" nature of operation, they spend a significant amount of time in their idle mode. Different circuit techniques have been proposed to reduce leakage power of such circuits. Leakage optimization can also be

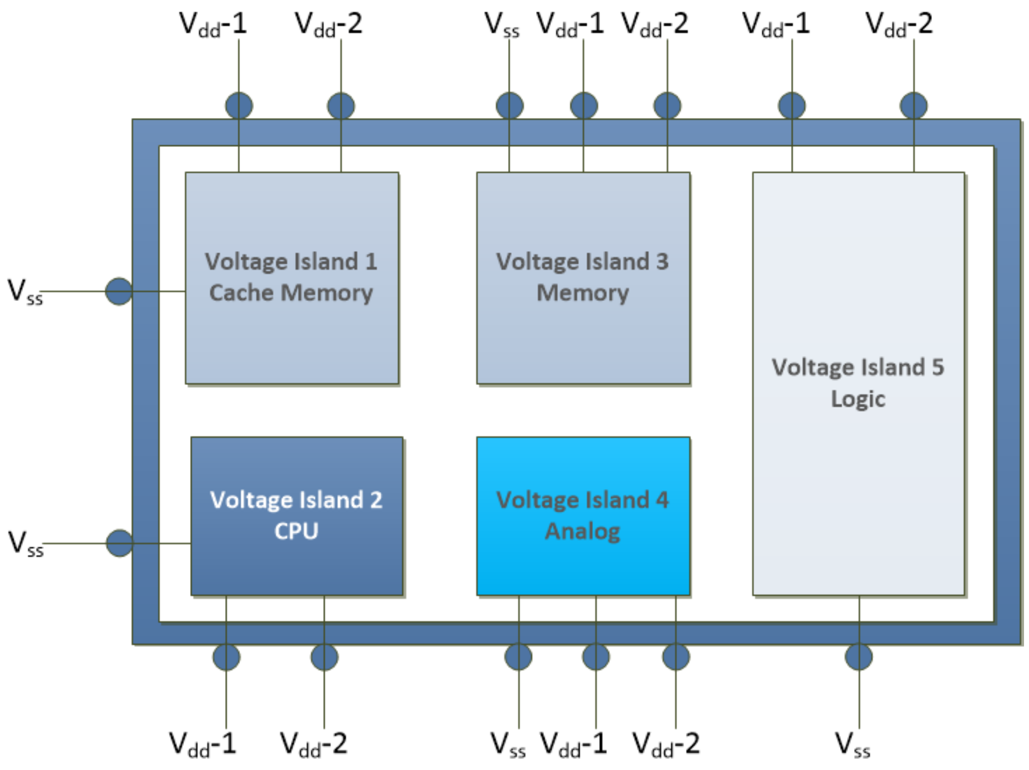


Figure 1.10: Multi-V_{dd} SoC with voltage islands.

performed at design time. Such approaches exploit the delay slack in non-critical paths to reduce leakage power. Runtime techniques reduce leakage power when circuits are not required to run at the highest performance level. A variety of methods that are based on the above mentioned techniques are discussed below.

Leakage current flowing through two serially stacked off CMOS transistors is always less than a single off device. This phenomenon is known as transistor stacking [28][29]. By replacing one off transistor with serially-connected, stacked transistors reduces leakage current significantly [30][31]. In addition, due to the transistor stacking effect, leakage current also depends upon the input vectors presented to a gate. Thus, if the input vector of a circuit entering in the standby mode is chosen carefully, then leakage power can be minimized by maximizing the number of stacked transistors in the off state. Numerous techniques for choosing the sleep vector have been proposed [32][33]. Another method to reduce static power is the dual-V_{th} technique. Today's Application Specific Integrated Circuits (ASIC) designers can choose standard cells from cell libraries with different V_{th}. During design synthesis, a high V_{th} can be assigned to some cells in the non-critical paths and low V_{th} transistors are assigned to cells in the critical paths so that performance is not sacrificed [34]. In this way, leakage power savings can be obtained while maintaining the desired performance. A number of methods utilize the slack in the non-critical paths to assign a high V_{th} [35][36]. The variable threshold CMOS method is a body-biasing technique for leakage power reduction [37]. Using adaptive body biasing the threshold voltage of the transistors can be increased in the standby mode while reduced dynamically

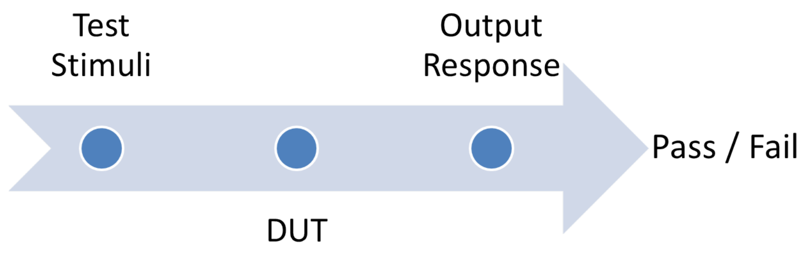


Figure 1.11: Typical Test.

in the active mode of operation depending upon the required performance level [40][41]. Dynamic threshold voltage scaling is a method of controlling the threshold voltage using substrate biasing proposed in [42]. Another method, which has the same effect as body-biasing is to raise the NMOS source voltage while tying the NMOS body to ground [43]. Finally, power gating is a major approach for leakage power reduction. This technique is implemented by inserting a high-V_{th} sleep transistor between the power supply and the original circuit [38][39]. The goal is to switch between active and sleep modes by selectively turning the sleep transistor on/off.

1.5 VLSI Testing

1.5.1 Terms and definitions

The objective of VLSI testing is to minimize the number of defective chips, resulting from imperfect manufacturing processes, shipped to customers. As shown in Fig. 1.11, it typically consists of applying a set of test stimuli to the inputs of the digital logic while analysing the output responses. Both input test stimuli and output response analysis can be generated and performed externally (Off-line Test) or inside the chip (On-line Test). Designs that produce the correct output responses for all input stimuli pass the test and are considered to be fault-free. Designs that fail to produce a correct response at any point during the test sequence are assumed to be faulty.

Two major defect mechanisms can cause the digital design to malfunction, manufacturing defects and soft errors. Manufacturing defects are physical defects introduced during chip fabrication that cause functional failures in the design. Manufacturing defects can result in static faults, such as stuck-at faults, or timing faults, such as delay faults. A general consensus, known as the rule of ten, says that the cost of detecting a faulty device increases by an order of magnitude as we move through each stage of manufacturing, from device level to board level, to system level, and finally, to system operation in the

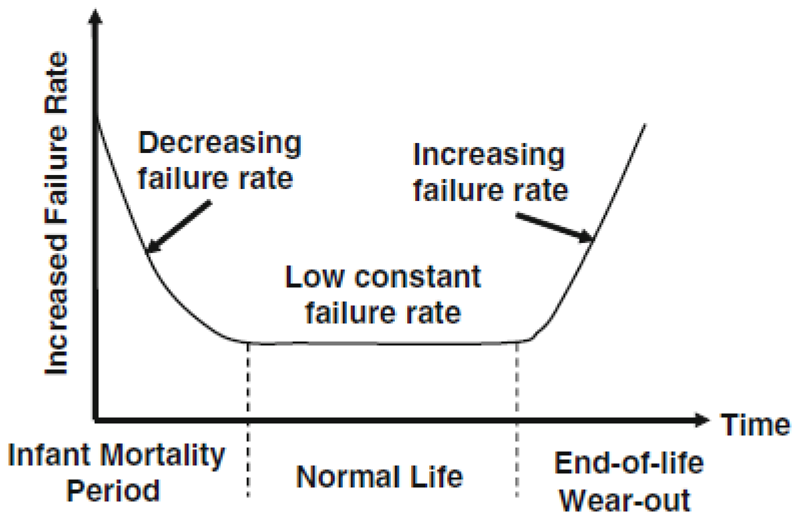


Figure 1.12: Rate of failure of integrated circuits at different phases of life [124].

field [119]. Soft errors are transient faults induced by environmental conditions such as α - particle radiation that cause a fault-free circuit to malfunction during operation [120], [121]. The probability to deal with a soft error increases as feature sizes decrease [122]. The transient faults are non-repeatable, and thus, they cannot be detected during manufacturing. The VLSI industry, to cope with the above challenges, created a framework of methods known as Design for Reliability (DFR) that aims to improve the reliability and availability of the produced chips.

An important aspect of the DFR is the yield Y of the manufacturing process. The later is defined as the percentage of acceptable parts among all parts that are fabricated, namely

$$Y = \frac{\text{Number of acceptable parts}}{\text{Number of parts fabricated}} \tag{1.4}$$

The yield can be reduced due to random defects (catastrophic yield loss) and process variations (parametric yield loss) that come from imperfections in the manufacturing process. Automation of and improvements in the IC fabrication process line drastically reduced the random defects. Thus, nowadays, the process variations are the dominant source of yield loss. Design for Yield Enhancement (DfY) [123] and Design for Manufacturability (DFM) are common terms in IC industry that are used to denote an effort to reduce the effects of process variations and to avoid random defects, correspondingly.

The bathtub curve [124] shown in Fig. 1.12 is a typical device or system failure chart indicating how early failures, wear-out failures, and random failures contribute to the overall device or system failures.

The infant mortality period (with decreasing failure rate) occurs when a product is in its early production stage. Failures that occur in this period are due to poor process or design quality. The product should not be shipped during this period to avoid massive field returns. The working life period (with constant failure rate) represents the product's "working life". Failures during this period tend to occur randomly. The wear-out period (with increasing failure rate) indicates the "end-of-life" of the product. Failures during this period are caused by age defects, such as metal fatigue, hot carriers, electromigration, dielectric breakdown. For electronic products, this period is of less concern because end users often replace electronic products before the devices reach their respective wear-out periods.

During manufacturing test, an error-free chip may fail the test and declared as faulty. For example, IR-drop can cause such an outcome. In addition, there are cases where a faulty chip passes the tests and it is declared as a good part, for example, due to insufficient test patterns. The ratio of field-rejected parts to all parts passing quality assurance testing is referred to as the reject rate, also called the defect level (DL),

$$DL = \frac{\text{Number of faulty chips passing final test}}{\text{Number of chips passing final test}} \tag{1.5}$$

For a given chip, the defect level DL is a function of process yield Y and fault coverage FC [125]

$$DL = 1 - Y^{(1-FC)} \tag{1.6}$$

where FC is defined as

$$FC = \frac{\text{Number of detected faults}}{\text{Total number of faults}} \tag{1.7}$$

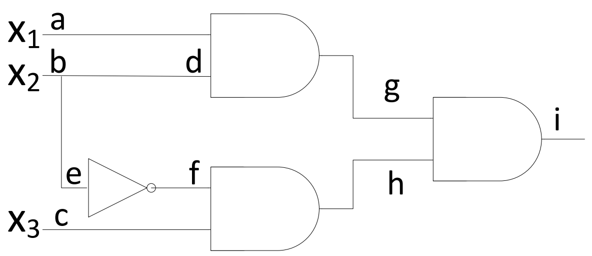
The defect level provides an indication of the overall quality of the testing process [126]. For example, a defect level of 300 parts per million (ppm) may be considered to be acceptable, whereas 50 ppm or less represents high quality. The goal of six sigma manufacturing, which is also referred to as zero defect, is 3.4 ppm or less [127].

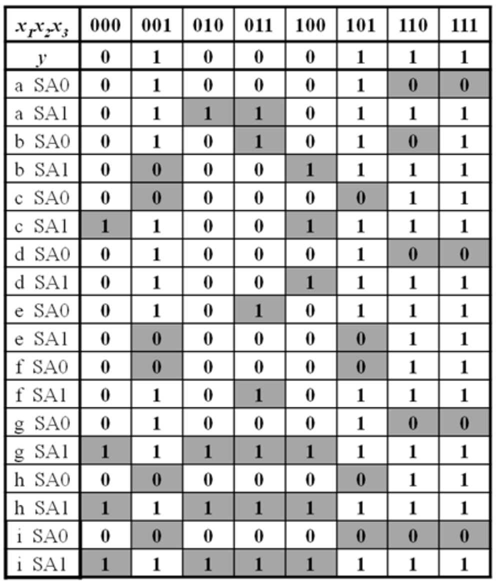
1.5.2 Fault models

A good fault model should satisfy two criteria: (a) it should accurately reflect the behaviour of the defects, and (b) it should be computationally efficient in terms of the time required for fault simulation and test generation [131]. In practice, a combination of different fault models is used in the generation and evaluation of test patterns so that the behaviour of the most possible defects can be captured. State-of-the-art fault models for general sequential logic are described in the following paragraphs.

A stuck-at fault transforms the correct value on the faulty signal line to appear to be stuck-at a constant logic value, either logic 0 or 1, referred to as stuck-at-0 (SA0) or stuck-at-1 (SA1), respectively. This model is commonly referred to as the line stuck-at fault model, where any line can be SA0 or SA1. It is also referred to as the gate-level stuck-at fault model where any input or output of any gate can be SA0 or SA1 [132]. Consider the example circuit shown in Fig. 1.13.

There are 18 ($2 \cdot 9$) possible faulty circuits under the single-fault assumption. In the right part of Fig. 1.13 [119], they are given the truth tables for the fault-free circuit and





$x_1x_2x_3$	000	001	010	011	100	101	110	111
y	0	1	0	0	0	1	1	1
a SA0	0	1	0	0	0	1	0	0
a SA1	0	1	1	1	0	1	1	1
b SA0	0	1	0	1	0	1	0	1
b SA1	0	0	0	0	1	1	1	1
c SA0	0	0	0	0	0	0	1	1
c SA1	1	1	0	0	1	1	1	1
d SA0	0	1	0	0	0	1	0	0
d SA1	0	1	0	0	1	1	1	1
e SA0	0	1	0	1	0	1	1	1
e SA1	0	0	0	0	0	0	1	1
f SA0	0	0	0	0	0	0	1	1
f SA1	0	1	0	1	0	1	1	1
g SA0	0	1	0	0	0	1	0	0
g SA1	1	1	1	1	1	1	1	1
h SA0	0	0	0	0	0	0	1	1
h SA1	1	1	1	1	1	1	1	1
i SA0	0	0	0	0	0	0	0	0
i SA1	1	1	1	1	1	1	1	1

Figure 1.13: Example circuit and its truth table [119].

the faulty circuits for all possible single stuck-at faults. The truth table entries where the faulty circuit produces an output response different from that of the fault-free circuit are highlighted in gray. As a result, the input values for the highlighted truth table entries represent valid test vectors to detect the associated stuck-at faults. With the exception of line d SAl, line e SA0, and line f SAl, all other faults can be detected with two or more test vectors. Then, test vectors 011 and 100 must be included in any set of test vectors that will obtain 100% fault coverage for this circuit. These two test vectors detect a total of ten faults, and the remaining eight faults can be detected with test vectors 001 and 110. Therefore, the set of these four test vectors obtains 100% single stuck-at fault coverage for the example circuit.

At the switch level, a transistor can be stuck-off or stuck-on, which are also referred to as stuck-open or stuck-short, respectively. A stuck-open transistor fault can cause the gate to behave like a dynamic level-sensitive latch. Thus, a stuck-open fault requires a sequence of two vectors. The first vector sensitizes the fault by establishing the opposite logic value to that of the fault-free circuit at the faulty node and the second vector propagates the faulty circuit value to a point of observability. Stuck-short faults can produce a conducting path between power (VDD) and ground (VSS) and may be detected by monitoring the power supply current during steady-state operation. This technique of monitoring the steady-state power supply current to detect transistor stuck-short faults is called IDDQ testing [133].

Consider the two-input CMOS NOR gate shown in Fig. 1.14. Let us assume that transistor N_2 is stuck-open. When the input vector $AB = 01$ is applied, output Z should be a logic 0, but the stuck-open fault causes Z to be isolated from V_{ss}. Since transistors

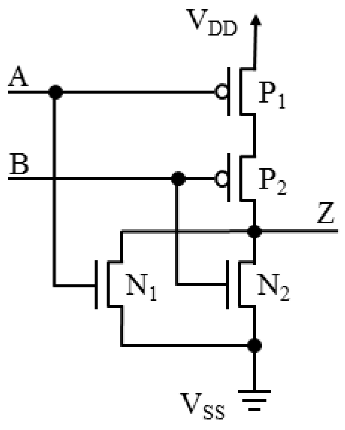


Figure 1.14: Example CMOS NOR gate.

P_2 and N_1 are not conducting at this time, Z keeps its previous state, either a logic 0 or 1. In order to detect this fault, an ordered sequence of two test vectors $AB_1 = 00$ and $AB_2 = 01$ is required. For the fault-free circuit, the first input produces $Z = 1$ and the second produces $Z = 0$. But, for the faulty circuit, while the first test vector produces $Z = 1$, the subsequent one will retain the value of $Z = 1$. Thus, a stuck-open fault requires a sequence of two vectors for detection rather than a single test vector for a stuck-at fault.

If transistor N_2 is stuck-short, there will be a conducting path between V_{DD} and V_{SS} for the test vector $AB_1 = 00$. This creates a voltage divider at the output node Z where the logic level voltage will be a function of the resistances of the conducting transistors. This voltage may or may not be interpreted as an incorrect logic level to the output node Z.

Defects can include opens and shorts in the wires that interconnect the transistors that form the circuit. A resistive open can affect the propagation delay of a signal path. A short between two wires is commonly referred to as a bridging fault. The case of a wire being shorted to V_{DD} or V_{SS} is equivalent to the line stuck-at fault model. However, when two signal wires are shorted together, bridging fault models are needed. The first bridging fault model proposed was the wired-AND/wired-OR bridging fault model. The most recent bridging fault model is the dominant-AND/dominant-OR bridging fault model. It is assumed that one driver dominates the logic value of the shorted nets for one logic value only [134].

The various bridging fault models that have been proposed in the past years are shown in Fig. 1.15. The first bridging fault model models the logic value of the shorted nets as a logical AND or OR of the logic values on the shorted wires. This model is referred to as the wired-AND/wired-OR bridging fault model. It was originally developed for bipolar

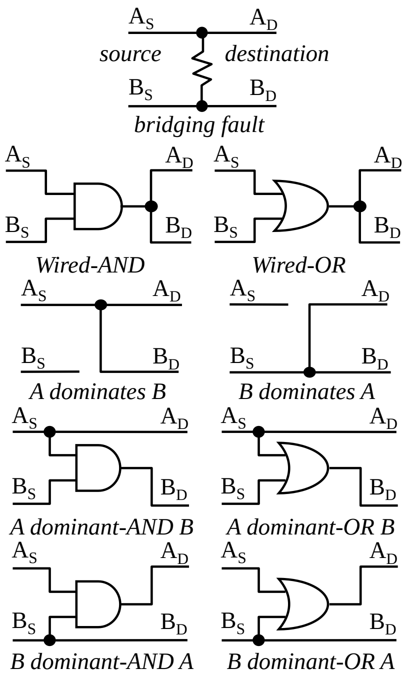


Figure 1.15: Examples of bridge fault models [119].

VLSI and does not accurately reflect the behaviour of typical bridging faults found in CMOS devices. Therefore, the dominant bridging fault model was proposed for CMOS VLSI where one driver is assumed to dominate the logic value on the two shorted nets. The dominant bridging fault model does not accurately reflect the behaviour of a resistive short in some cases. Thus, a new bridging fault model has been proposed, referred to as the dominant-AND/dominant-OR bridging fault. In this case, one driver dominates the logic value of the shorted nets but only for a given logic value.

Resistive opens and shorts in wires as well as parameter variations in transistors can cause excessive delays such that the total propagation delay falls outside the specified limit. Delay faults have become more prevalent with decreasing feature sizes. A variety of different delay fault models are available. In gate-delay fault and transition fault models, a delay fault occurs when the time interval for a transition through a single gate exceeds its specified range. The path-delay fault model, on the contrary, considers the cumulative propagation delay along any signal path through the circuit. Thus, the path-delay fault model seems to be more practical for testing than the gate-delay fault or the transition fault model. A critical problem encountered when dealing with path-delay faults is the large number of possible paths in practical circuits. The small delay defect model takes into consideration the timing delays associated with the fault sites and propagation paths from the layout [135].

As with transistor stuck-open faults, delay faults require an ordered pair of test vectors to sensitize a path through the logic circuit and to create a transition along that path in

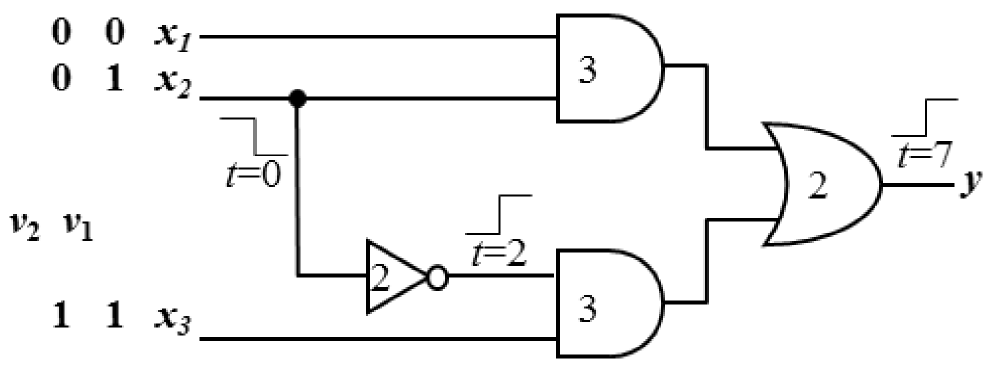


Figure 1.16: Example circuit [119].

order to measure the path delay. Let us consider the circuit in Fig. 1.16.

The two test vectors, v_1 and v_2, shown in the Fig. 1.16 are used to test the path delay from input x_2. Assuming the transition between the two test vectors occurs at time $t = 0$, the resulting transition propagates to the output y, through the circuit with the fault-free delays at time $t = 7$. A delay fault along this path would create a transition at some later time, $t > 7$. Such a measurement requires a high-speed, high-precision test machine. With decreasing feature sizes and increasing signal speeds, the problem of modelling gate delays becomes more difficult. In deep sub-micron region, the component of delay contributed by gates reduces while the delay due to interconnect becomes dominant. In addition, if the operating frequencies also increase with process scaling, then the on-chip inductances can play a role in determining the interconnect delay for long wide wires, such as those in clock trees and buses.

1.5.3 The logic test process

Logic testing refers to the testing of the digital logic portion of a DUT. The thoroughness of such testing process strongly depends on the quality of test patterns. Thus, a quality assessment to test patterns is required to determine if an accepted level of product quality can be achieved. Fault simulation is the process for such quality assessment. The tasks involved in fault simulation are illustrated in Fig. 1.17.

As shown, the input to fault simulation consists of a fault list, the DUT netlist and the input stimuli that needs to be evaluated. For each fault from the fault list, the DUT is simulated in the presence of the fault. The output responses with respect to the given input stimuli are then compared with the expected fault-free responses to determine whether the fault can be detected by the given input stimuli.

The fault list comprises from a set of target faults. Subsets of faults in the fault list may present identical behaviour for every test pattern, namely they are equivalent faults.

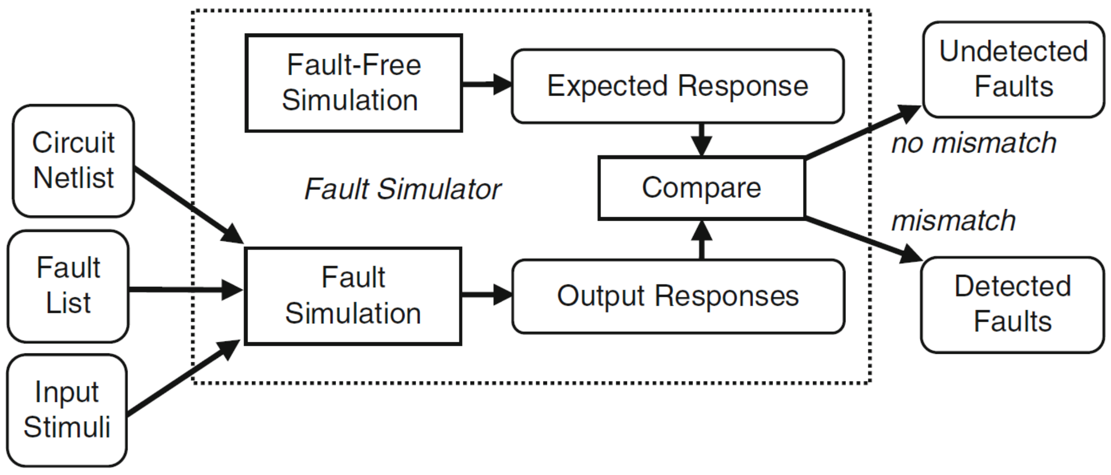


Figure 1.17: Fault simulation process [127].

Obviously, only one fault from each subset of equivalent faults needs to be simulated, while the remaining ones can be safely deleted. This process is known as fault collapsing and reduces drastically test time and test cost since the size of a collapsed fault list is typically about 40% of the original one [127]. The netlist is derived via RTL synthesis of the DUT down to a gate-level design.

The input stimuli can be created either functionally or structurally. In the first case, ideally, all possible input test patterns, that is every entry in the truth table, are applied to DUT. The structural testing constitutes a more practical approach. The test patterns are selected according to the circuit structural information and a set of fault models. The fault models provide a quantitative measure of the fault detection capabilities for a given set of test patterns for the targeted fault model. This measure is called fault coverage. Any input pattern or sequence of input patterns that produces a different output response for a faulty circuit from that of the fault-free circuit is a candidate test pattern or sequence of test patterns for detecting the fault. An Automatic Test Pattern Generation (ATPG) method is employed to find a set of test patterns that detects all targeted faults in a DUT. ATPG for a given target fault consists of two steps: fault activation and propagation. Fault activation establishes a signal value at the fault site opposite to the value produced by the fault. Fault propagation propagates the fault effect forward by sensitizing a path from the fault site to a primary output.

Structural testing saves test time and improves test efficiency because the total number of test patterns is decreased by targeting specific faults that would result from defects in the manufactured device. Moreover, when test patterns are generated to detect faults of one model, their application order is further processed by re-ordering techniques, so that additional fault models can be served by the same test pattern sequence. In this case, the required number of test vectors is further reduced and, in turn, test time and test cost are decreased. A common practice in industry is to target delay faults first, followed by gate-level stuck-at faults, bridging faults, and finally, transistor-level stuck faults [127].

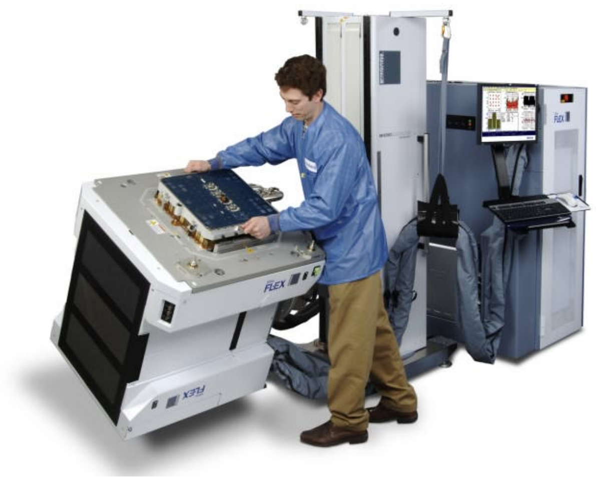


Figure 1.18: Automatic test equipment from Teradyne Inc. [141].

However, structural testing cannot guarantee detection of all possible manufacturing defects because the test patterns are generated based on specific fault models.

In general, the logic test process consists of the following steps:

- Definition of the targeted fault models. Calculations related to DL and manufacturing yield are based on this step,
- Decision upon the Design-for-Testability (DFT) techniques that should be used in design to meet the test requirements,
- Generation and evaluation of test patterns in terms of fault coverage,
- Execution of manufacturing test upon a batch of chips to separate fault from good ones,
- Only in case of low DL or yield Y, implementation of Failure Mode Analysis (FMA).

1.5.4 Test equipment

To perform (off-line) manufacturing test on a DUT after it is fabricated, it is required an Automatic Testing Equipment (ATE) or simply tester, a test fixture, an ATPG and a test program. Production testers are usually expensive pieces of equipment with configurable I/O ports (able to drive DUT pins, measure DUT signals, act as a load) and sufficient storage behind them for test patterns and the expected responses. The tester drives input pins from memory on a cycle-by-cycle basis and samples and stores the levels on output pins. Fig. 1.18 shows a typical production tester.

As shown in Fig. 1.18, in the background, there is a bay cabinet holding the drive electronics and the controlling workstation. The test head is shown on the front. Initially, a specialized handler (not shown in picture) feeds the DUT to a test fixture attached to the tester, using mechanical means. The handler adds a constant time to the test

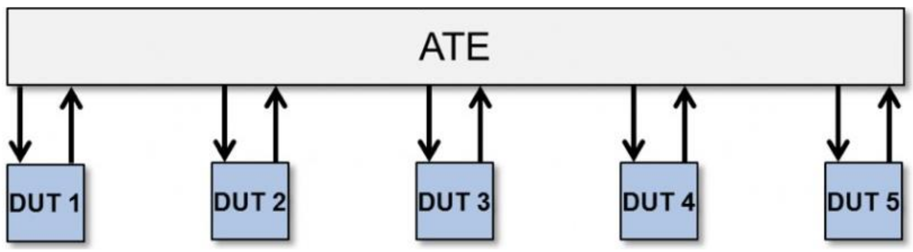


Figure 1.19: Multisite testing example.

process, typically around one second [9]. Thus, in many cases, multiple handlers are used to deal with two or four SoCs at once to reduce the cost of testing (multi-site testing). The test fixture can be provided in the form of a probe card or a load board. Then, the test program is compiled and downloaded into the tester and the tests are applied to the bare die. This program is normally written in a high-level language, such as the IMAGE language used by Teradyne (based on C), that supports a library of primitives for a particular tester. The test program specifies a set of input patterns and a set of output assertions that are generated by an ATPG tool. If an output does not match the asserted value at the corresponding time, the tester will report an error. In this case, the DUT is marked as faulty (with an ink dot) and the failing tests may be displayed for reference and stored for later analysis. In the case of a probe card, the card is raised, moved to the next die on the wafer, lowered, and the test procedure repeated. In the case of a load board with automatic part handling, the tested part is removed from the board and sorted into a good or bad bin. A new part is fed to the load board and the test is repeated. In most cases, these procedures take a few seconds for each part tested.

According to the function of the parts that are being tested, testers are categorized as digital, memory, or analog. The cost of the testers increases with the number of pins. The high cost per pin is primarily because they are testing leading-edge technology using existing technology. Scheduling their use efficiently can offset this high cost. That is, cost is kept low by making test time minimal. Another way of minimizing test application time and cost is to combine off-chip with on-chip tests.

1.5.5 Multisite test

Multisite Testing, i.e. testing multiple dice at the same time, is becoming more common, even on the most complex devices as ATE are becoming better architected. Multisite testing under the right conditions may has the most significant impact on throughput and cost of test of any other input factor[102]. A number of test cost reduction strategies based on multi-site test have been proposed with the goal to make test cost scale with technology progress [98][99][100][101]. An example of multisite testing is shown in Fig.

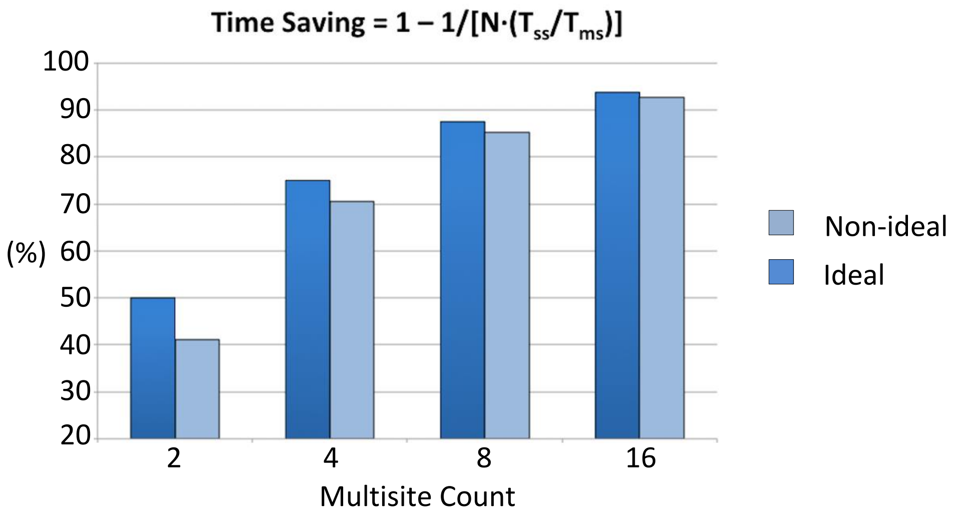


Figure 1.20: Theoretical vs practical test time savings due to multisite testing [76].

1.19.

The number of dice that can be tested at once, or the multisite count, depends on the number of ATE channels and the number of test pins per die. The time saved, compared with a single-site approach, increases with the multisite count, N, according to the equation $1 - 1/N$. Thus, in Fig. 1.19, where the number of die tested at once is five, if there are enough ATE channels to accommodate the additional chip I/Os, the time saving is 80%. However, the test time saving in the previous example is theoretical. The real time saving when applying multisite testing is calculated in a more complex way and the average time to test each die using multisite test, T_{ms}, is almost always greater than that for single-site test, T_{ss}. [102][103][104] describe the models that derive the test time and cost savings of multi-site test. Fig. 1.20 shows how theoretical multisite test time savings diverge from practical results.

Multisite test time can be increased for various reasons, including:

- Problems with the probe card touching down on die at the edge of a wafer, which means that they have to be tested separately and/or repeatedly,
- Too few test resources on the tester to support full multisite testing due to costs,
- The time to position the probe interface to contact the bonding pads of the DUT
- The branching in the test program flow, which can make one site's execution time longer than another's.

Any increase of the multisite test time affects throughput, defined as the multisite count divided by the multisite test time, and measured in terms of units tested per hour

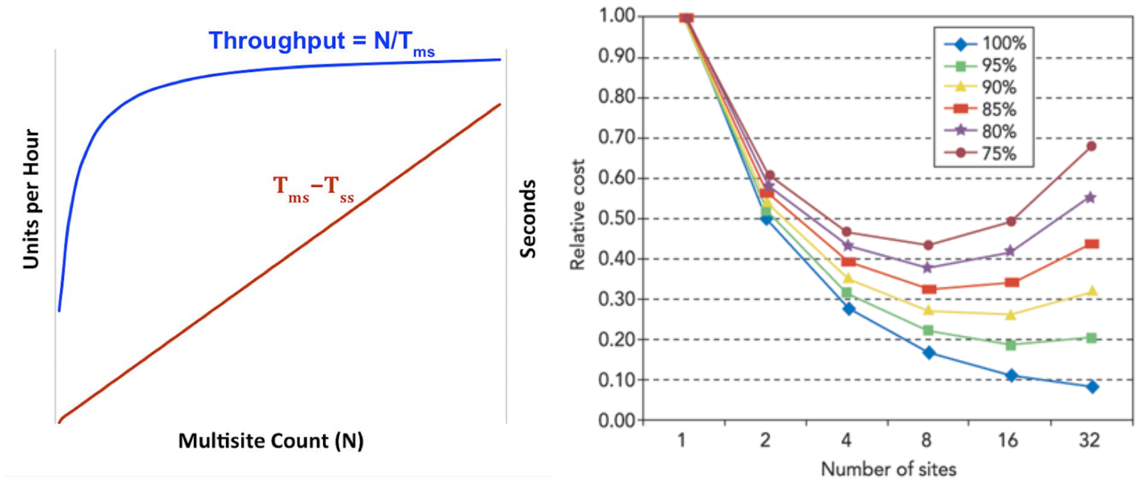


Figure 1.21: Effect of tester efficiency as multisite count rise[76][105].

(UPH). If the multisite test time per die continues to increase significantly with site count, the throughput can reach a point of diminishing returns [102][103][76][105]. A tester must be more than 75% efficient to provide any real benefit beyond quad site. To be cost-effective beyond eight sites, a tester must be more than 90% efficient[105], as it is illustrated in Fig. 1.21.

1.5.6 Thermal-aware test

The deep sub-micron technology is characterized by high power density. The later, in cases of economically beneficial but limited cooling support, can develop upon an SoC excessive heat that may damage it or downgrade its performance and lifetime expectations. To overcome the overheating problem, temperature-aware chip design methodologies have been proposed, that are usually applied during the placement and routing phase of the SoC's design flow [182] - [184]. However, these methods target the normal operation of the SoC and they are not taking into account the impact of power hungry defect screening processes which, most of the times, totally neglect power consumption and violate the design specifications of SoCs [198][199][200].

Overheating during test may force good ICs to fail. In other words, overheating can decrease the yield and rise the production cost. Even if SoCs escape permanent damage, they can still fail the test due to excessive interconnection delays. These are increased approximately by 5% for every $10^{o}C$ increase in the temperature of an SoC [184]. Thus, overheating can lead to temporary path delay faults in a good chip and, hence, testing will produce incorrect results. On the other hand, even if a chip passes the test, its exposure to high temperature could reduce its lifetime (aging effect: electromigration, time dependent dielectric breakdown).

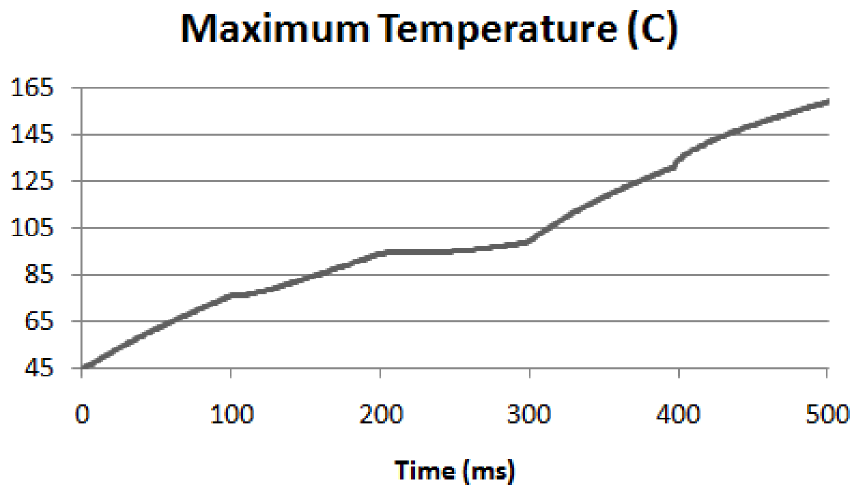


Figure 1.22: Effect of inter-core distance in the thermal behaviour of an embedded core [201].

Thermal-aware test schedule seems to be governed by certain principles that influence the thermal behaviour of the embedded cores in an SoC and should be taken into account during the scheduling of a test in the various DFT designs. These principles are identified and analysed below.

In [201], an SoC is considered with 36 embedded cores and a Test Access Mechanism (TAM) that allows the parallel test of six cores, thus, six different test sessions should be run to test all the cores. During each test session, the remaining cores are considered idle. The test application time for each test session has been set to 100ms and the ambient temperature is assumed to be 45^oC. The thermal profile of a core tested during the 5^{th} test session is shown in Fig. 1.22. As it is illustrated, although the core is not tested for the first 400ms, its temperature increases. This is due to the fact that, during this time period, other cores were tested and the temperature rise of these cores has a lateral thermal impact on the core in question. Furthermore, in [201] is accented that depending on the proximity of the other cores being tested, the increase of temperature to the core in question can be significant.

In [201], the contribution of the ambient temperature to the maximum temperature reached by a core is also studied. To this goal, a variety of test methods were applied in an SoC for 25^oC, 35^oC and 45^oC ambient temperature values. The results are shown in Fig. 1.23. The graph illustrates that the difference between ambient temperature values is directly added / subtracted to core's maximum temperature for the same test method.

Moreover, a careful analysis of the graph in Fig. 1.23 clearly shows that the longer the test methods are applied upon the DUT, the higher the temperature that is developed in it. This observation suggests that if test methods are not used with caution, the chip can be damaged during test.

The table in Fig. 1.24 shows power density, maximum temperature and fault coverage

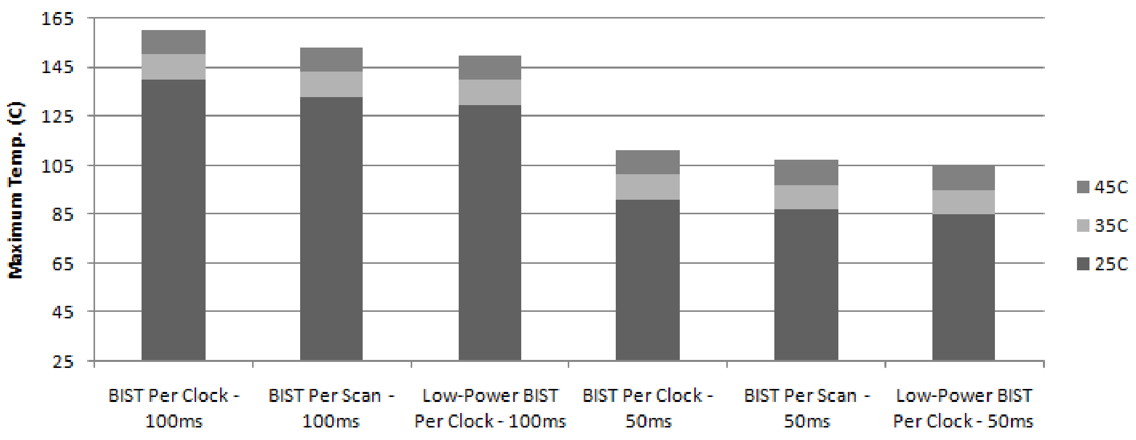
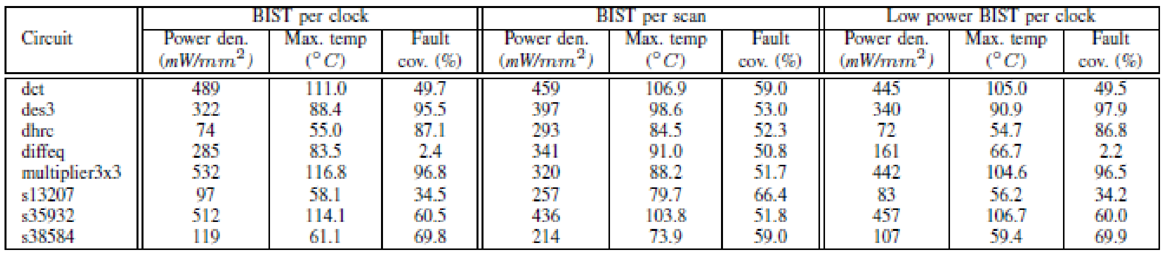


Figure 1.23: Effect of ambient temperature in the thermal behaviour of an embedded core [201].



Circuit	BIST per clock			BIST per scan			Low power BIST per clock		
	Power den. (mW/mm^2)	Max. temp ($^\circ C$)	Fault cov. (%)	Power den. (mW/mm^2)	Max. temp ($^\circ C$)	Fault cov. (%)	Power den. (mW/mm^2)	Max. temp ($^\circ C$)	Fault cov. (%)
dct	489	111.0	49.7	459	106.9	59.0	445	105.0	49.5
des3	322	88.4	95.5	397	98.6	53.0	340	90.9	97.9
dhrc	74	55.0	87.1	293	84.5	52.3	72	54.7	86.8
diffeq	285	83.5	2.4	341	91.0	50.8	161	66.7	2.2
multiplier3x3	532	116.8	96.8	320	88.2	51.7	442	104.6	96.5
s13207	97	58.1	34.5	257	79.7	66.4	83	56.2	34.2
s35932	512	114.1	60.5	436	103.8	51.8	457	106.7	60.0
s38584	119	61.1	69.8	214	73.9	59.0	107	59.4	69.9

Figure 1.24: Effect of power density in the thermal behaviour of an embedded core [201].

values for various test methods for a fixed period of testing. The results prove that when the power density is increased, the maximum temperature is increased too.

In addition, Fig. 1.25 presents the maximum temperature values for a single-chain scan design when one, two, three and four time periods are used. For example, if there is only one time period, all cores are tested at the same time. It can be concluded that, although testing more cores in parallel decreases the overall test application time, the temperature of the chip increases. Therefore decreasing the test application time is not enough for generating temperature-aware tests. It is important to take into account both the power consumption and test application time simultaneously.

The table in Fig. 1.26 shows the average power consumption and the switching activity of ISCAS'89 benchmarks. The results show that switching activities are much higher during test, with an 8.5% average during normal operation and a 26.9% average during test. On average, switching activity increases by a factor $4.1x$. However, the average increase in power consumption is only $1.6x$. Based on the average switching activities and the linear dependence of power consumption on switching activity, one might expect to see an approximate $4.1x$ increase in power consumption as well. This discrepancy is due to the high power consumption of the clock tree. The reported numbers represent the switching activity in the combinational logic. However, a large fraction of the power consumption is actually due to the toggling of the flip-flop clock inputs. This power

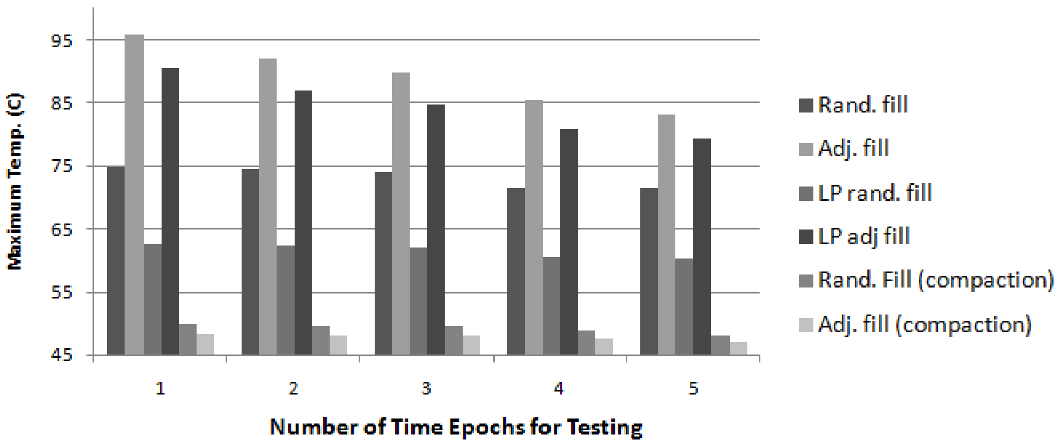


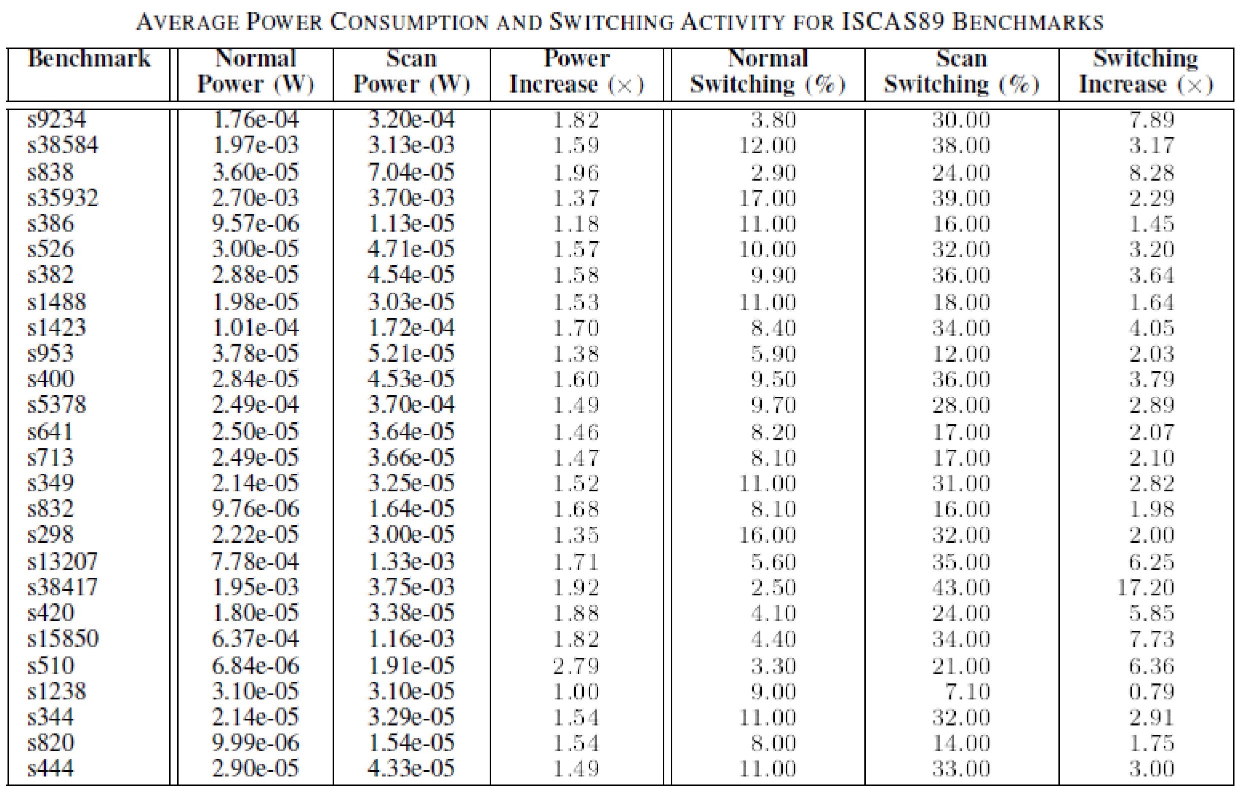
Figure 1.25: Effect of test time and power density in the thermal behaviour of an embedded core [201].

consumption is independent of the reported switching activity, and thus the 4.1x increase in switching activity results in only a 1.6x increase in power consumption. The 1.6x increase in average power implies that if the normal operating frequency is more than twice the testing frequency, power consumption will be higher during normal operation. Thus, thermal problems will occur during scan-chain testing only in three circumstances:

- The circuit is tested at a frequency greater than 1/2 normal operating frequency, e.g., during at-speed testing or built-in self test. This would result in higher power consumption than during normal operation.
- The circuit has a greater inter-register combinational logic depth than the ISCAS'89 benchmarks, resulting in a greater dependence of total power consumption on combinational switching activity. This would result in higher power consumption than during normal operation.
- The circuit is tested in a thermal environment that is inferior to that used during normal operation, e.g., if the heat sink and fan used during testing, results in a higher thermal resistance to the ambient than during normal operation.

1.5.7 Design for Testability

The nanometre technology allowed for the implementation of complex VLSI designs. Such designs require special features to be integrated in their digital logic, to enable controllability and visibility even of their deepest elements / modules. The process of integrating test logic in a VLSI design is called DFT and it has become a requirement in the IC industry. They have been developed four DFT methodologies, the ad hoc, the scan design, the Built-In Self-Test (BIST), and the test compression methodology. The most popular of them and in use today are the scan design and the scan-based logic BIST [128][129]. Both



AVERAGE POWER CONSUMPTION AND SWITCHING ACTIVITY FOR ISCAS89 BENCHMARKS

Benchmark	Normal Power (W)	Scan Power (W)	Power Increase (×)	Normal Switching (%)	Scan Switching (%)	Switching Increase (×)
s9234	1.76e-04	3.20e-04	1.82	3.80	30.00	7.89
s38584	1.97e-03	3.13e-03	1.59	12.00	38.00	3.17
s838	3.60e-05	7.04e-05	1.96	2.90	24.00	8.28
s35932	2.70e-03	3.70e-03	1.37	17.00	39.00	2.29
s386	9.57e-06	1.13e-05	1.18	11.00	16.00	1.45
s526	3.00e-05	4.71e-05	1.57	10.00	32.00	3.20
s382	2.88e-05	4.54e-05	1.58	9.90	36.00	3.64
s1488	1.98e-05	3.03e-05	1.53	11.00	18.00	1.64
s1423	1.01e-04	1.72e-04	1.70	8.40	34.00	4.05
s953	3.78e-05	5.21e-05	1.38	5.90	12.00	2.03
s400	2.84e-05	4.53e-05	1.60	9.50	36.00	3.79
s5378	2.49e-04	3.70e-04	1.49	9.70	28.00	2.89
s641	2.50e-05	3.64e-05	1.46	8.20	17.00	2.07
s713	2.49e-05	3.66e-05	1.47	8.10	17.00	2.10
s349	2.14e-05	3.25e-05	1.52	11.00	31.00	2.82
s832	9.76e-06	1.64e-05	1.68	8.10	16.00	1.98
s298	2.22e-05	3.00e-05	1.35	16.00	32.00	2.00
s13207	7.78e-04	1.33e-03	1.71	5.60	35.00	6.25
s38417	1.95e-03	3.75e-03	1.92	2.50	43.00	17.20
s420	1.80e-05	3.38e-05	1.88	4.10	24.00	5.85
s15850	6.37e-04	1.16e-03	1.82	4.40	34.00	7.73
s510	6.84e-06	1.91e-05	2.79	3.30	21.00	6.36
s1238	3.10e-05	3.10e-05	1.00	9.00	7.10	0.79
s344	2.14e-05	3.29e-05	1.54	11.00	32.00	2.91
s820	9.99e-06	1.54e-05	1.54	8.00	14.00	1.75
s444	2.90e-05	4.33e-05	1.49	11.00	33.00	3.00

Figure 1.26: Effect of test time and power density in the thermal behaviour of an embedded core [155].

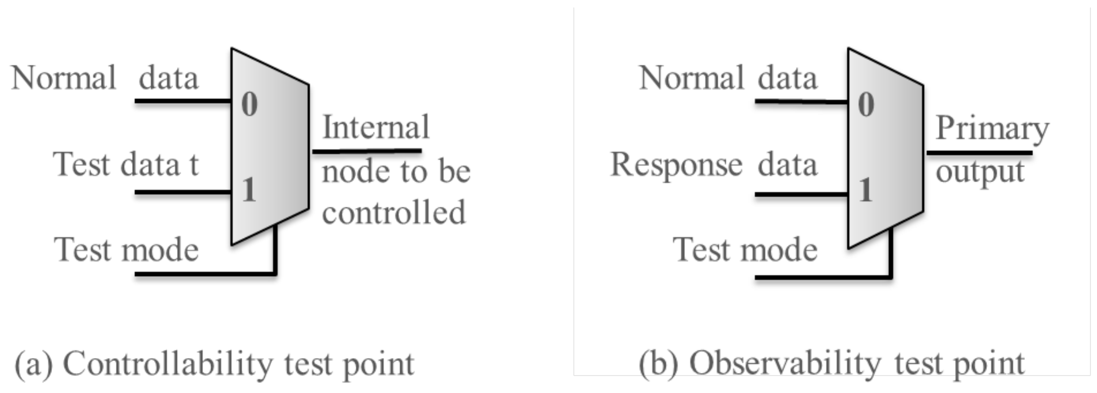
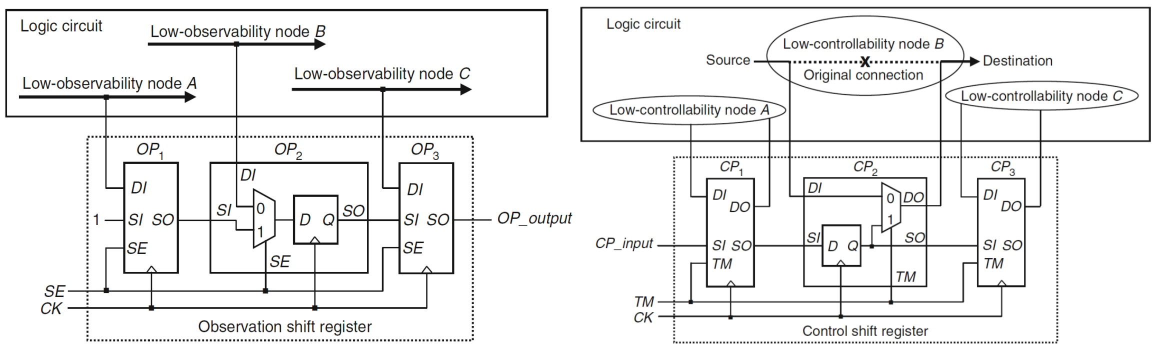

Normal data
0
Test data t
1
Internal node to be controlled
Test mode
(a) Controllability test point
Normal data
0
Response data
1
Primary output
Test mode
(b) Observability test point

Figure 1.27: Controllability and Observability points.



(a) Observation point insertion (b) Control point insertion

Figure 1.28: Example of ad-hoc methodology [127]

.

techniques have proven to be effective in producing efficiently testable VLSI designs. Test compression is used as supplementary DFT technique to further reduce test data volume and test application time during manufacturing test [119][130]. Ad-hoc methods are not in great use today. In the following lines, we describe shortly each one of the aforementioned methods.

Ad-hoc methods were the first DFT techniques introduced in the 1970s [75] to target only those portions of the circuit that were difficult to test. The circuitry shown in Fig. 1.27, typically referred to as test points, was added to improve the observability and/or controllability of internal nodes [119].

Fig. 1.28a shows an example of observation point insertion for a logic circuit with three low-observability nodes. OP2 shows the structure of an observation point, which is composed of a Multiplexer (MUX) and a D flip-flop. An SE signal is used for MUX port selection. When SE is set to 0 and the clock CK is applied, the logic values of the low-observability nodes are captured into the D flip-flops. When SE is set to 1, the D flip-flops within OP1, OP2, and OP3 operate as a shift register, allowing us to observe the captured logic values through OP output during sequential clock cycles. As a result, the observability of internal nodes is greatly improved. Fig. 1.28b shows an example of control point insertion for a logic circuit with three low-controllability nodes. CP2 shows the structure of a control point, which is composed of a MUX and a D flip-flop too. The original connection at a low controllability node is cut, and a MUX (Fig. 1.27) is inserted between the source and destination ends. During normal operation, TM is set to 0 such that the value from the source end drives the destination end. During test, TM is set to 1 such that the value from the D flip-flop drives the destination end. The D-flip-flops in CP1, CP2, and CP3 are designed to form a shift register so that the required value can be shifted into the flip-flops using CP_input and used to control the destination ends of low-controllability nodes. As a result, the controllability of the circuit nodes is dramatically improved.

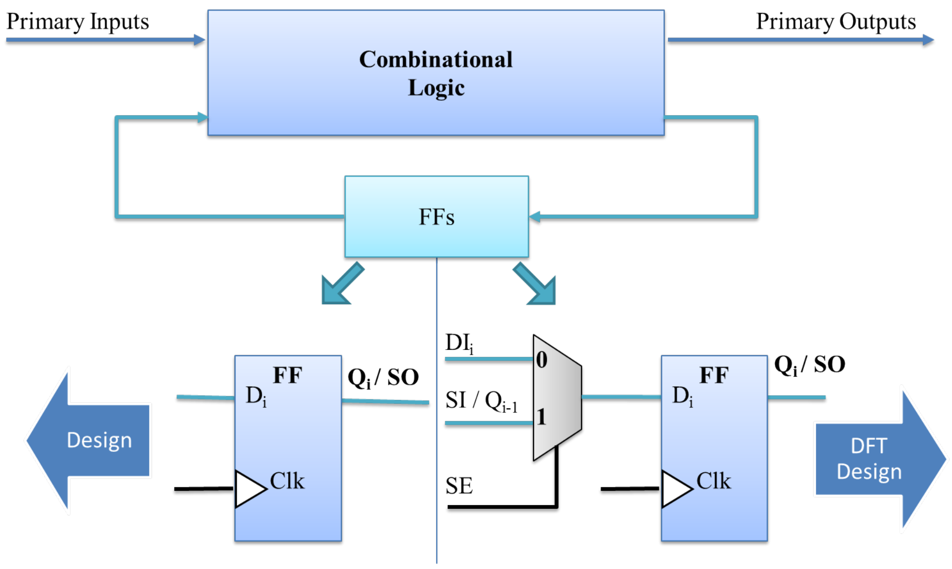


Figure 1.29: Muxed-D scan cell.

Scan design, the most widely used DFT approach, is implemented by replacing selected storage elements in a design with specialized ones, they called scan cells, and then connecting them into one or more shift registers, which are called scan chains. The fundamental scan architectures [119] include the muxed-D scan design, where storage elements are converted into muxed-D scan cells, the clocked-scan design, where storage elements are converted into clocked scan cells, and the Level-Sensitive Scan Design (LSSD) scan cells, where storage elements are converted into LSSD shift register latches [118]. An example muxed-D scan cell is depicted in Fig. 1.29.

As it is illustrated, the selected storage elements are replaced by scan cells, each of which has one additional scan input (SI) port and one shared/additional scan output (SO) port. By connecting the SO port of one scan cell to the SI port of the next scan cell, one or more scan chains are created. The scan-inserted design operates in three modes: normal mode, shift mode, and capture mode. In normal mode, all test signals are turned off, and the scan design operates in the original functional configuration. In both shift and capture modes, a TM signal is often used to turn on all test-related functions in compliance with scan design rules. The scan design rules [119] are necessary to simplify the test, debug, and diagnosis tasks, improve fault coverage, and guarantee the safe operation of the DUT.

Consider the example of a muxed-D scan design illustrated in Fig. 1.30, where each storage element has been reconfigured as a scan cell, as shown in Fig. 1.29. The scan cells are connected in series to form a shift register, or scan chain, that has direct access to a primary input (scan in) and a primary output (scan out). During the shift operation, when the scan enable is set to active, the scan chain is used to shift in a test pattern through the primary input scan in. After the test pattern is shifted into the scan cells, it is applied to the logic cloud. The circuit is then configured in capture mode, by setting the scan enable to inactive, for one clock cycle. The response of the combinational logic in the cloud with respect to the test pattern is then captured in the scan cells. The scan

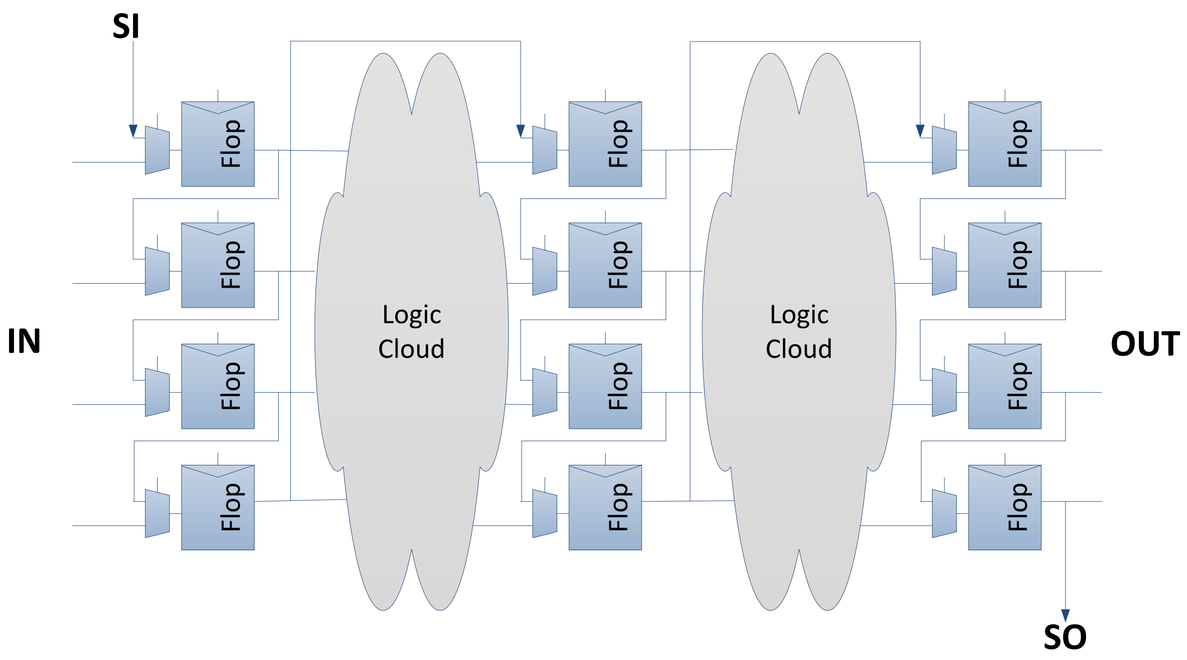


Figure 1.30: An example muxed-D scan design [9].

chain is then configured in scan mode again to shift out the response captured in the scan cells for observation. While shifting out the response, the next test pattern can be shifted into the scan cells concurrently.

In BIST, as illustrated in Fig. 1.31, a Test Pattern Generator (TPG) is used to automatically supply the internally generated test patterns to the DUT and an Output Response Analyzer (ORA) is used to compact the output responses from DUT [131]. The TPG and ORA are either embedded in the chip or elsewhere on the same board where the chip resides.

The test compression is commonly used to reduce the amount of test data that need to be stored on the ATE [130]. Reductions in test data volume and test application time by 10x or more can be achieved. This result is typically accomplished by including a

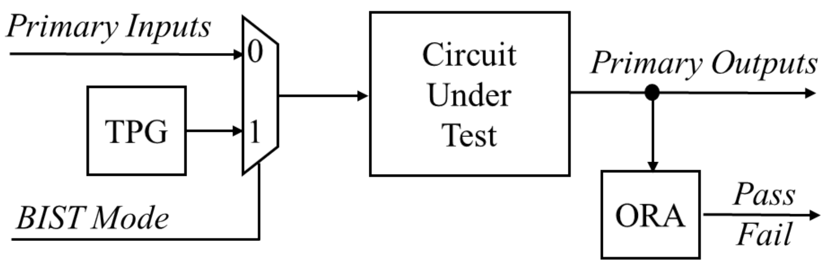


Figure 1.31: Built-in self-test architecture [119].

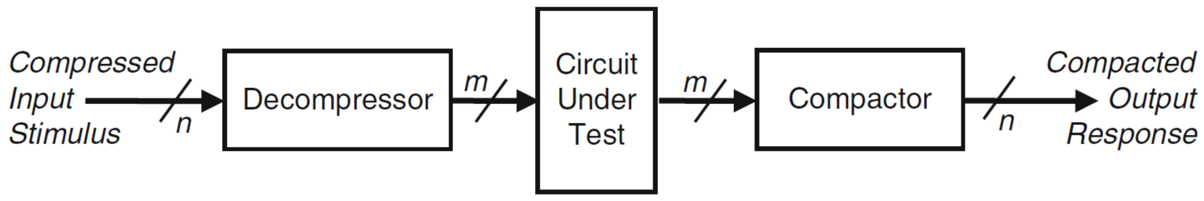


Figure 1.32: Test compression architecture [127].

decompressor before the m scan chain inputs of the DUT to decompress the compressed input stimuli and also adding a compactor after the m scan chain outputs of the DUT to compact the output responses, as illustrated in Fig. 1.32.

Typically DFT logic is inserted at the RTL level to ensure the quality of the fabricated chips. Then, the derived design is verified and test patterns are generated to ensure that test requirements are met. The test requirements are often specified in terms of DL and manufacturing yield Y, test cost, and whether it is necessary to perform self-test and diagnosis.

1.6 SoC Testing

1.6.1 Basic principles

Since SoCs are designed in modular fashion, their testing can be performed in a modular manner too (Fig. 1.33). In this case, the embedded cores in the DUT are only activated when they are tested. The gain is the ability to reduce the test application time and control the test power consumption. Specifically, the modular test enables the test designer to plan the test order of the cores such that test time and power constraints are met.

In order to enable modular test, each embedded core in an SoC must be transformed to a testable unit. This is succeeded using a core test wrapper, or simply wrapper. Its role is dual. First, to isolate the core so that it can act as a stand-alone test unit. Second, to define the interface between the core and the infrastructure for test data transportation, the TAM, so that test access can be efficient. Nowadays, the SoC industry mainly uses the core test wrapper described by IEEE 1500 Standard for Embedded Core Test (SECT) [77]. This standard is similar to 1149.1 [78] in that its main objective is to standardize boundary test circuitry (wrappers) for cores. However, unlike 1149.1, it provides parallel access capability for a core. Thus, test application time for an SoC can be significantly improved. Furthermore, in contrast to 1149.1, where control signals are mainly generated by a finite state machine that is controlled by a single input, in 1500 standard the control signals can be directly applied to a core, thus providing more test flexibility.

The wrapped cores (the testable units) in an SoC do not have direct access to SoC pins. In order to transport test data, namely test vectors (test stimuli) and test responses,

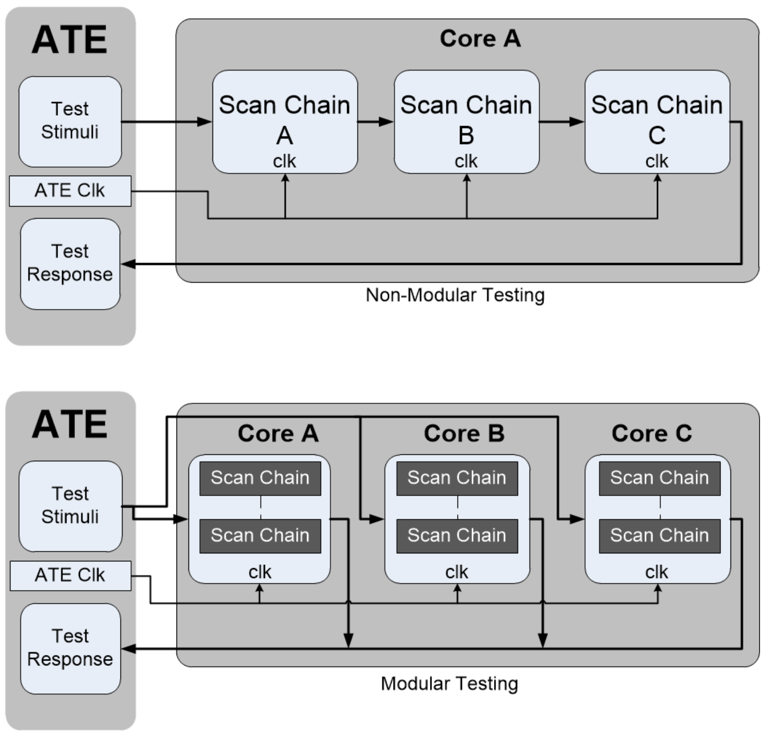


Figure 1.33: Non-Modular vs Modular Testing.

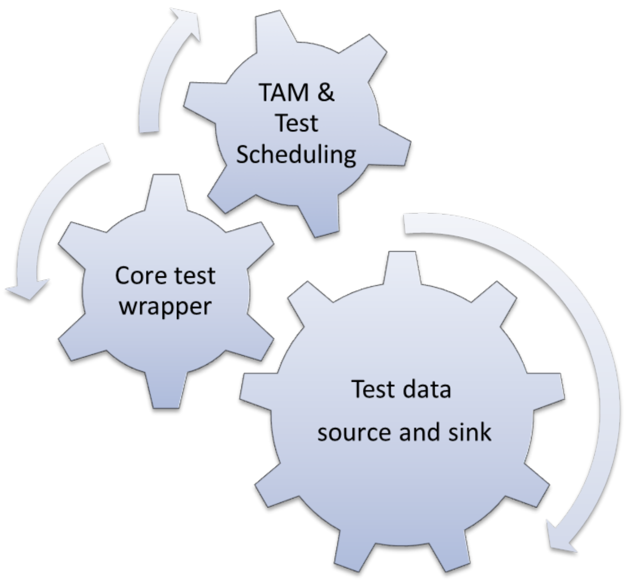
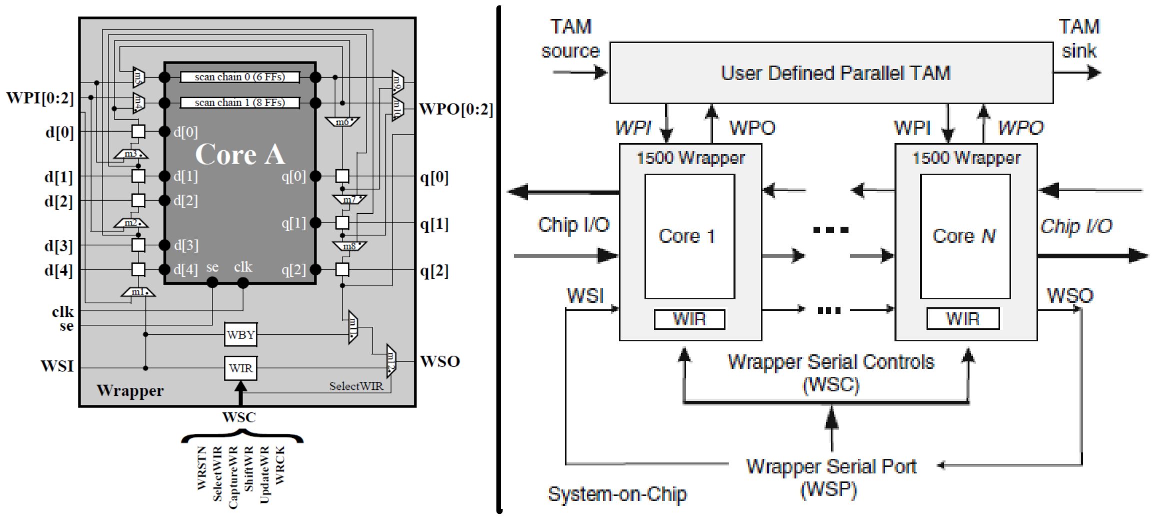


Figure 1.34: SoC generic test architecture.



(a) Example Core and its IEEE 1500 wrapper (b) SoC architecture based on IEEE 1500 standard

Figure 1.35: IEEE 1500 standard [77].

to and from embedded cores, an infrastructure is needed, the TAM (Fig. 1.36). To reduce the cost of DFT structures embedded in the SoCs, the TAM resources are typically shared among different cores. However, sharing of TAM resources leads to test conflicts that are resolved using TAM optimization and test scheduling techniques. For a given SoC where the embedded cores are wrapped such that each core is a testable unit and a TAM exists, the test scheduling is the process of planning the order in which the cores are to be tested. The scan chains at each testable unit are formed into wrapper chains, and given the test patterns, each testable unit is associated with a test time. The objective of test scheduling is to guide the test application such that the overall test cost, which is directly related to test application time, is minimal. SoC test scheduling constitutes a significant part of our research work and it will be analysed further in the next chapter.

1.6.2 Test wrapper - IEEE 1500 standard

The IEEE 1500 standard [77] describes in detail the main core-isolation mechanism that is used in the SoC test process. The primary structure is a wrapper surrounding the boundary (I/O signals) of each core that facilitates the isolation and access of the core from its SoC environment. The test patterns that are communicated through the wrapper can be generated using any ATPG method. An overall architecture of an SoC with N cores, each wrapped by an IEEE 1500-compliant wrapper, is shown in Fig. 1.35.

The Wrapper Serial Port (WSP) is a set of I/O signals of the wrapper for serial operations, which consists of the Wrapper Serial Input (WSI), the Wrapper Serial Output

(WSO), and several Wrapper Serial Control (WSC) signals. Each wrapper has a Wrapper Instruction Register (WIR) to store the instruction to be executed in the corresponding core, which controls operations in the wrapper including accessing the Wrapper Boundary Register (WBR), the Wrapper Bypass Register (WBY), or other user-defined function registers. The WBR consists of Wrapper Boundary Cells (WBC) that can be similar to the Boundary-Scan Cell (BSC) [18] or a more sophisticated cell with multiple storage devices on its shift path. The WSP supports the serial test mode (TM) similar to that in the boundary-scan architecture, but without using a Test Access Port (TAP) controller. This means that the WSC signals defined in the IEEE 1500 standard can be directly applied to the cores for enhanced test flexibility. In addition to the serial TM, the IEEE 1500 standard also provides an optional parallel TM with a user-defined, parallel TAM. Each core can have its own Wrapper Parallel Input (WPI), Wrapper Parallel Output (WPO), and Wrapper Parallel Control (WPC) signals.

1.6.3 Test Access Mechanism

As shown in Fig. 1.36, the TAM is a hardware architecture used to communicate test data to and from embedded cores. Numerous TAM configurations have been proposed in the literature. There are TAM architectures that use the existing resources in SoC, for example an existing functional bus [142]. However, most of the proposed TAMs are based on dedicated hardware. Such an approach offers increased flexibility, but with the cost of hardware overhead. In the next few lines, we present shortly such state-of-the-art TAM configurations. In the multiplexed architecture [143], only one core can get access to the available SoC TAM lines at a time, hence interconnect testing between cores is not easy to achieve. In the daisy-chain architecture [143], all cores can access all TAM wires during a test session and each core can be tested sequentially. However, it presents inefficiencies when only a subset of cores is to be accessed simultaneously, as for example in case of power aware testing. In the locally controlled TAM [144], a dedicated controller for each core is used, which allows the test procedures for all cores to be carried out simultaneously. This, however, requires significant hardware overhead. In the direct-access architecture [145], the available TAM wires are distributed over the wrapped cores. The optimal number of TAM wires to be assigned to a core depends on test requirements of the core, and a test plan considering the requirements of all cores may be necessary in order to minimize the total test time. The Test Bus architecture [146] uses both the multiplexing and the distribution configurations.

Furthermore, a number of TAM configurations have been proposed to minimize the overall SoC test time for a given number of TAM wires by determining the number of distinct TAMs, their widths, and the assignment of these TAMs to cores. Optimization approaches for test bus architectures can be found in [81], [85], [83] and [147].

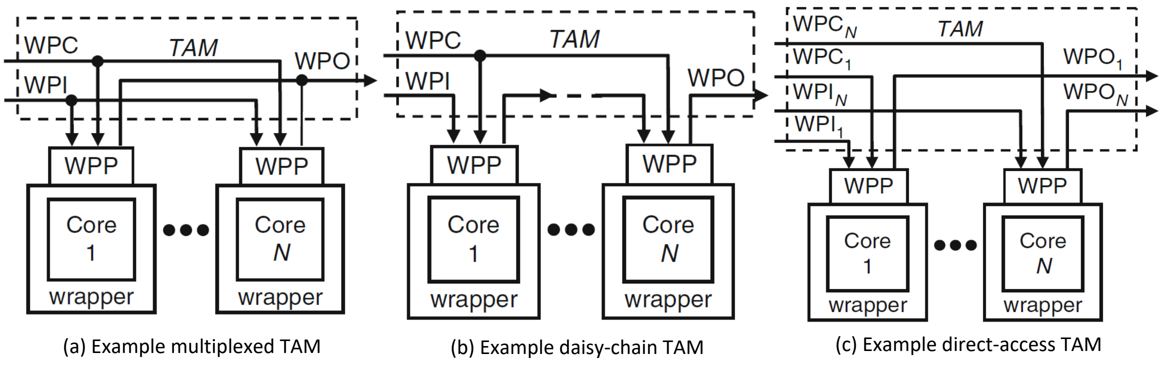

WPC
TAM
WPO
WPI
WPP
Core 1
Core N
wrapper
(a) Example multiplexed TAM
(b) Example daisy-chain TAM
WPC_N
WPC_1
WPI_N
WPI_1
WPO_1
WPO_N
(c) Example direct-access TAM

Figure 1.36: Example TAM structures.

1.6.4 SoC testing requirements

Fig. 1.2 presents the conceptual architecture of the SoC paradigm. The relationship between the embedded cores and the SoC seems to be analogous to that between ICs and a Printed Circuit Board (PCB). Therefore, a test architecture similar to a boundary scan could also be used for SoC testing too. Although many concepts developed for boundary scan have been applied to SoC testing, there are fundamental differences between SoCs and PCBs. Whereas in a PCB the different ICs have been designed, verified, fabricated, and tested independently from the board, manufacturing and testing of an SoC are implemented only after integration of the different cores. Even in the case where each core is accompanied by its own test set, the incorporation of all the test sets in the SoC testing process may not be a simple step. Each of the embedded cores may adopt different test strategies and technologies or the design and test sources may be written in different HDL languages, such as Verilog, VHDL, and Hardware C to GDSII. The main test problems for a multi-core SoC have been discussed in [119], [138], [139].

The embedded cores in an SoC may come from different vendors, in soft, hard or firm form. It is also possible that they are differentiated in terms of analog / digital / manufacturing technologies. Consequently, it is almost impossible for a test designer to develop all the required tests without the assistance of the core providers. The Virtual Socket Interface Alliance (VSIA) group and the IEEE Test Technology Technical Council (TTTC) undertook the task to facilitate the communication between core creators and SoC designers, through the establishment of the needed standards, such as the IEEE 1450.6 (Core Test Language (CTL)) [140]. Moreover, due to IP protection considerations, the internal structural information of a core should remain hidden. Hence, the core provider, in order to protect its property without hindering the test designer's efforts, should develop a core test set that can be used with very limited or no modifications.

A multi-core SoC may incorporate a number of deeply embedded cores. To access and test such cores in an efficient manner, a TAM is required. In addition, deeply embedded cores could have a TAM-plug-and-play feature to ease the system integration. Furthermore, a core itself may consist of cores in a hierarchical manner. In this case, a TAM only for cores at the top level of a hierarchy is insufficient. An efficient and effective hierarchical test structure is needed to test the cores at the lower level of the hierarchy, too. Hierarchical cores could also have a TAM-plug-and-play feature at any hierarchical level to simplify the integration work.

The clock rate inside a core can be significantly higher than the one supported by SoC's pins. In addition, ATE test clocks, in many cases, cannot support at-speed testing even if the core is isolated and well accessed through a TAM. Raising the test clock rate using a dedicated phase-lock loop would significantly complicate the design, resulting in unacceptable test costs. In this case, employing normal functional units to create the required at-speed test environment seems to allow for efficient, effective, and economic testing.

An SoC may incorporate digital, analog and memory devices. Each one of them may require ATE with differentiated specifications for testing. This can be proved extremely expensive. Using BIST methods to move some test control or test data generation mechanism into the SoC can potentially reduce the use of external ATE and reduce the test cost.

When the cores in an SoC are tested sequentially, long test time is required and hence, test cost is increased. Parallel testing or test scheduling is necessary to reduce test time. However, excessive parallel testing may lead to excessive power consumption in SoC. The implications may include incorrect test results or even damage in the devices under test. A test schedule must be carefully planned so as not to violate any constraint or limit of test power.

1.6.5 Bending the cost curve

The increasing design size, the complexity and the power-aware design of SoCs transformed the landscape in the manufacturing test. New plans, new paths need to be shaped so that the test cost can be retained in acceptable levels that will not diminish the benefits of the SoC paradigm.

An SoC test developer, or integrator in the SoC era, has to consider how to develop a complete test plan for a mix of proprietary cores, delivered in different formats (e.g., soft, hard or firm cores), implemented with different technologies, operating at different speeds, using different power management techniques. The developer must consider the total amount of test time and test data volume required to test all embedded cores, since the cost of testing an SoC is usually proportional to them [9]. The developer must consider the high test power consumption, which can be several factors higher than the functional power consumption for which an IC is designed, so that deficits due to yield loss and damaged SoCs can be avoided. Finally, the developer must fit optimally the above considerations in the framework of the accessible test resources, and optimize the performance of the available automation tools.

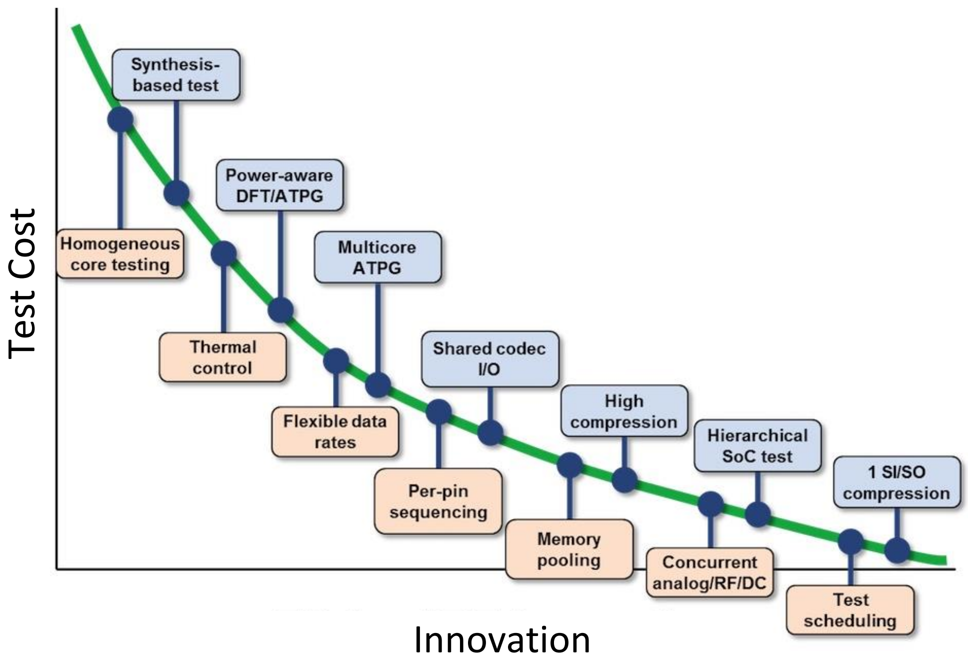


Figure 1.37: Effect of testing innovation[76].

The key to keep the cost in manufacturing test of an SoC low is making test time minimal while power and other design constraints are met.

1.7 Thesis Organization

Chapter 2 presents the state of art in multi-V_{dd} test and SoC test scheduling, two topics that are closely related to this research work. It starts with the presentation of the multi-V_{dd} test. Its nature, the most important V_{dd}-dependent defects and their impact to the test process and cost, are described in detail. A brief overview of the existed methods to reduce the multi-V_{dd} test cost is given too. Concerning the SoC test scheduling, the basic principles are analysed initially. Then, existed power- and thermal- aware test scheduling methodologies are described in detail.

Chapter 3 presents thoroughly and in depth the research work. It starts with the research directions and then, the main research objectives are defined. A detailed description of the methodology to succeed the target goals follows. Each proposed method is accompanied by explanatory examples.

Chapter 4 presents the tools and the work-flows that have been developed during this research work. They provided with an integrated SoC design and test environment that enabled the efficient and reliable deployment and execution of experiments upon the methods discussed in chapter 3.

Chapter 5 includes a set of carefully selected, evaluation-based experimental procedures that prove the innovation and the added value of the proposed methods.

Finally, chapter 6 provides conclusions and directions for future work.

Chapter 2

Background

2.1 Multi-V_{dd} test

Multi-V_{dd} testing was proposed over a decade ago to improve reliability [163]. It was shown that testing between $2 \cdot V_{th}$ and $2.5 \cdot V_{th}$, where V_{th} is the transistor threshold voltage, achieves high-defect coverage for resistive bridges. In the modern low power multi-V_{dd} designs, the repetitive test in the supported operating voltages seems to be a necessity. Some manufacturing defects have V_{dd} dependency, which implies that defects can become active only at certain voltage setting. In other words, when single-V_{dd} testing is applied to multi-V_{dd} designs the defect coverage is reduced. There are two major types of defects that show V_{dd}-dependent detectability, the resistive bridge and resistive open defects. Non-resistive defects, in general, are not V_{dd}-dependent.

2.1.1 Resistive bridge defects

Resistive bridge defects are appeared when a fault metal connection is created between two lines of the circuit. Since physical defect between an interconnect line and power supply or ground line, commonly referred to as hard-short, is unaware of V_{dd} settings [164], the focus is given to the resistive bridge between signal lines. A typical resistive bridge is shown in Fig. 2.1.

In [166], it was shown that a resistive bridge changes the voltages on the bridged lines from $0V$ (logic-0) or V_{dd} (logic-1) to some intermediate values, which are dependent to the resistance of the bridge, R_{sh}. Then, the logic behaviour of the physical defect can be expressed in terms of the logic values perceived by the gate inputs that are driven by

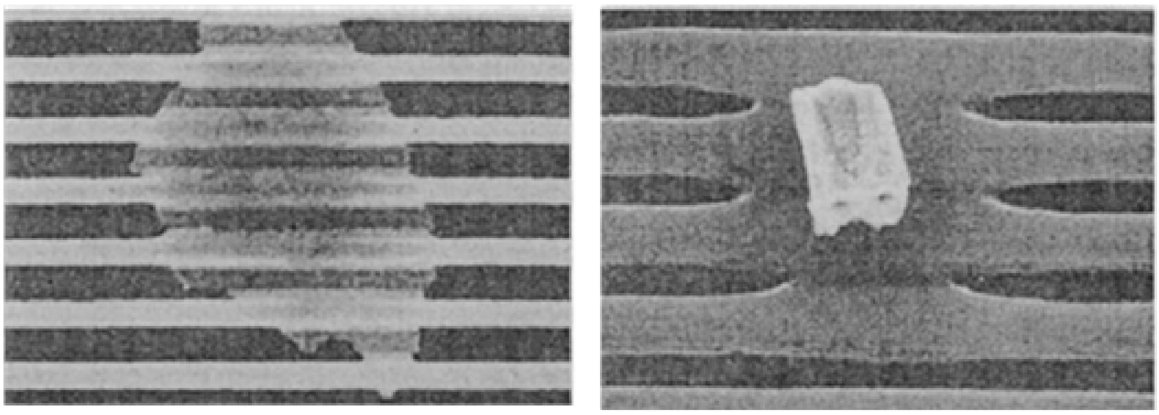


Figure 2.1: Resistive Bridge Defect [165].

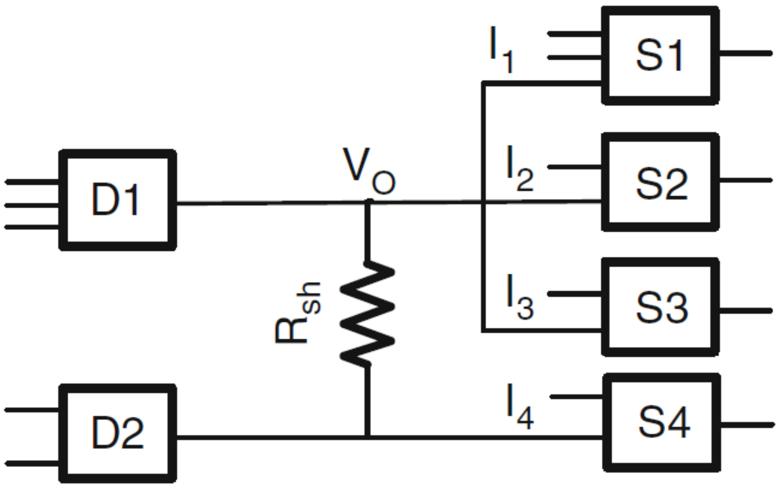


Figure 2.2: An example bridge fault scenario [127].

the bridged nets based on their specific input threshold voltage. A typical bridge fault scenario is illustrated in Fig. 2.2.

D1 and D2 are the gates driving the bridged nets, while S1, S2, S3, and S4 are successor gates, that is gates having inputs driven by one of the bridged nets. The resistive bridge affects the logic behaviour only when the two bridged nets are driven at opposite logic values. For example, consider the case when the output of D1 is driven high and the output of D2 is driven low. Let us assume that the shown bridge R_{sh} affects only the output of D1. Then, S1, S2, and S3 are affected by the resistive bridge. The dependence of the voltage level V_o on the output of D1 on the equivalent resistance of the physical bridge is shown in Fig. 2.3.

The deviation of V_o from the nominal voltage level V_{dd} elevates as R_{sh} decreases. To translate this analogue behaviour into the digital domain, the input threshold voltage levels V_{th1}, V_{th2} and V_{th3} of the successor gates S1, S2, and S3 have been added to the V_o plot. For each value of the bridge resistance R_{sh}, the logic values at inputs I1, I2, and I3

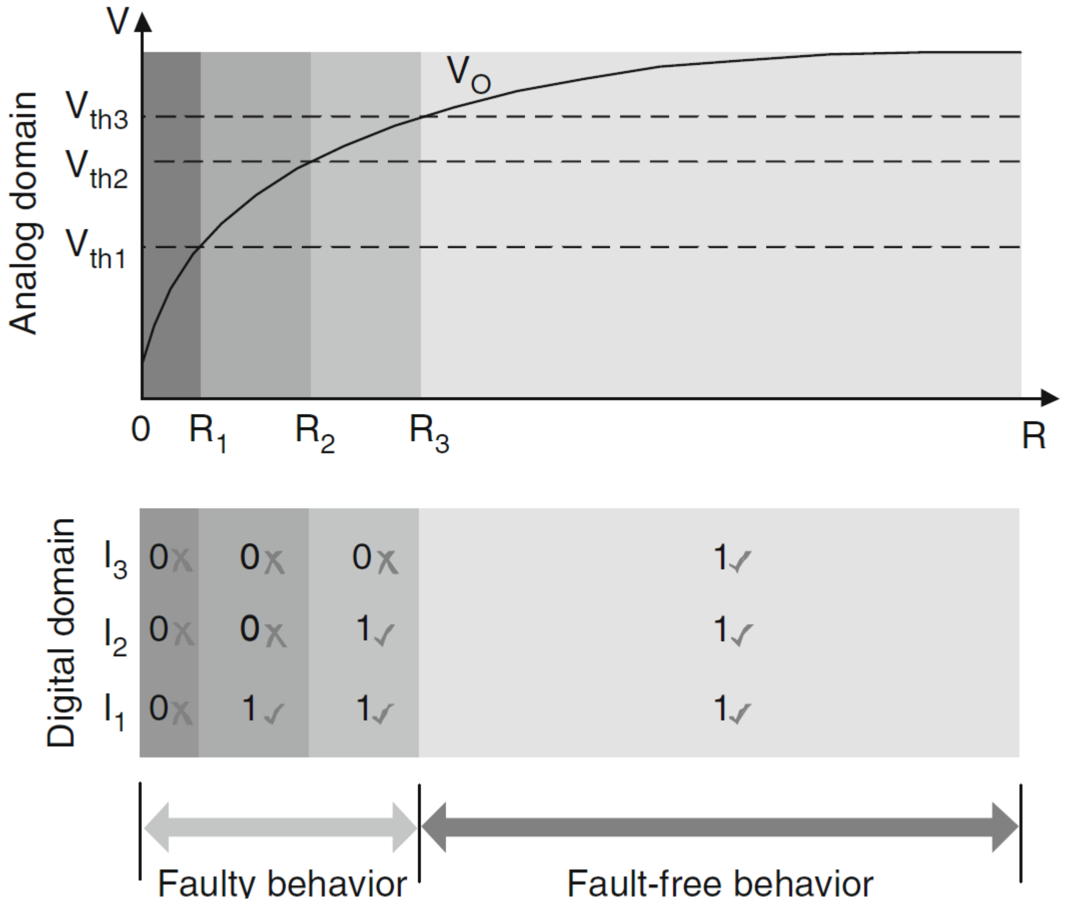


Figure 2.3: Behaviour of a bridge fault at a single-V_{dd} setting [127].

can be determined by comparing V_o with the input threshold voltage of the corresponding input. These values are shown in the second part of Fig. 2.3. Crosses are used to mark the faulty logic values and ticks to mark the correct ones. As it is illustrated, for bridge with $R_{sh} > R_3$, the logic behaviour at the fault site is fault-free, while for bridge with R_{sh} between 0 and R_3, one or more of the successor inputs produce a faulty logic value. The value of R_3 that represents the crossing point between faulty and correct logic behaviour is referred to as 'critical resistance'. Methods for determining the critical resistance have been presented in several publications [167].

A number of bridge resistance intervals can be identified based on the corresponding logic behaviour. For example, all bridges with $R_{sh} \in [0, R_1]$ exhibit the same faulty behaviour in the digital domain. Again, for bridges with $R_{sh} \in [R_1, R_2]$, successor gates S2 and S3 interpret the faulty value, while S1 interprets the correct value. Finally, for bridges with$R_{sh} \in [R_2, R_3]$ only S3 interprets a faulty value while the other two successor gates interpret the correct logic value. Consequently, each interval $[R_i, R_{i+1}]$ corresponds to a distinct logic behaviour occurring at the bridge fault site. The logic behaviour at the fault site can be captured using a data structure further referred to as Logic State Configuration (LSC), which can be looked at as logic fault model [89]. Several test generation methods for Resistive Bridge Faults (RBF) have been proposed for a fixed

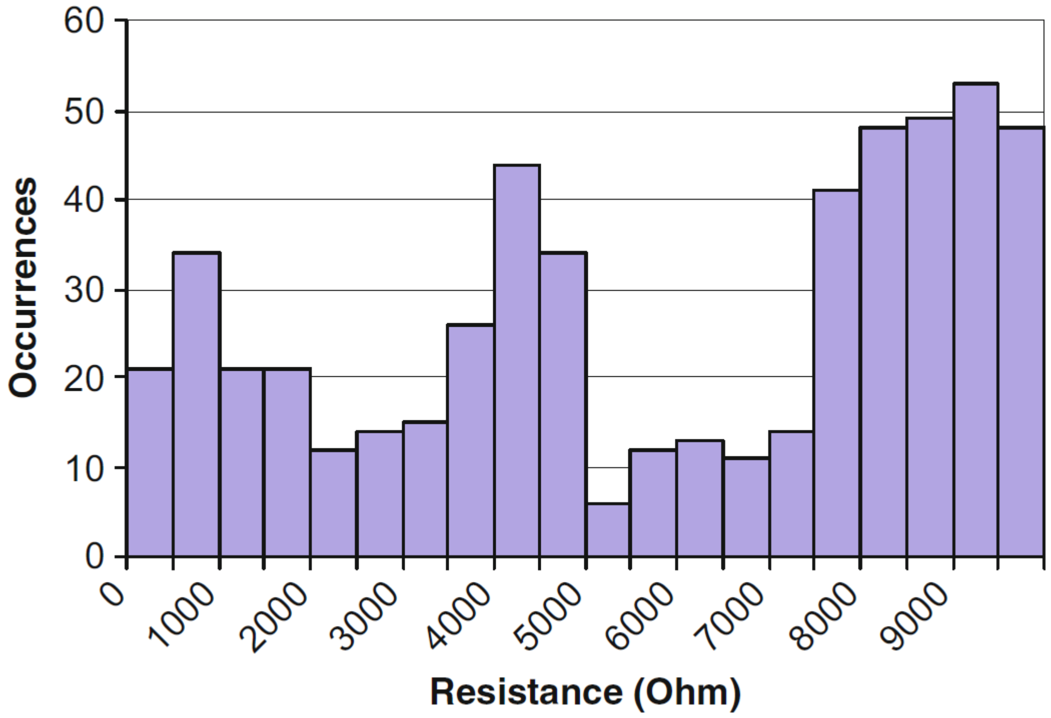


Figure 2.4: Resistance values that cannot be detected at lowest V_{dd} setting [89].

supply voltage setting [167][168].

In [169], it was studied the effect of varying the supply voltage on the defect coverage. The experimental results show that the fault coverage of a given test can vary both ways when the supply voltage is lowered, because not all faults can be covered using a single-V_{dd} setting during test. Thus, authors suggest to apply the tests at a lower and nominal supply voltage in order to improve the fault coverage. In [89], the same result has been illustrated in Fig. 2.4.

Fig. 2.4 shows the number of defects and respective resistance values, which cannot be detected at $V_{dd} = 0.8V$. The results are based on seven of the medium- and large-size ISCAS-85 and -89 benchmarks. The random spread of these defects across the resistance range suggests that to ensure high-defect coverage it will be necessary to test at more than one V_{dd} setting for 100% defect coverage.

Fig. 2.5 shows the relation between the voltage on the output V_o of gate D1 in the example circuit of Fig. 2.2 and the bridge resistance for two different supply voltages Vdd_A and Vdd_B. Three distinct logic faults LF_1, LF_2, and LF_3 have been identified for each V_{dd} setting. As depicted, the resistance intervals corresponding to LF_1, LF_2, and LF_3 differ from Vdd_A to Vdd_B since the value of V_o does not scale linearly with the input threshold voltages of S1, S2, and S3 in the two voltage settings. This means that a test pattern targeting a particular logic fault will detect different ranges of physical defects when applied at different supply voltage settings. For example, at Vdd_A, a test pattern targeting LF_3 will detect bridge with $R_{sh} \in [R_{2A}, R_{3A}]$, while at Vdd_B it will detect a

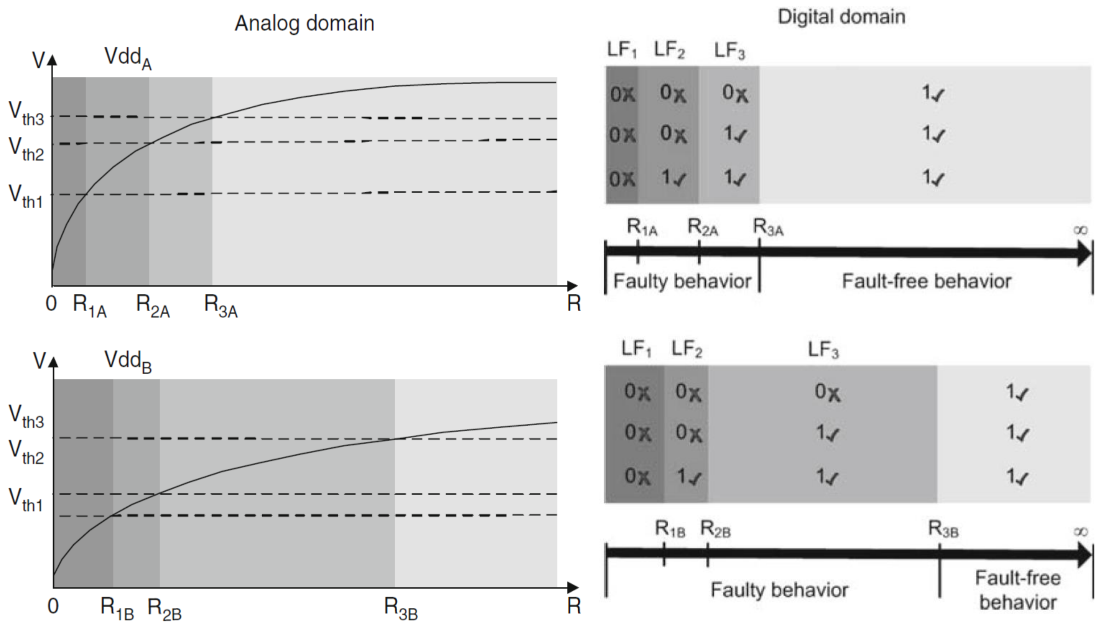


Figure 2.5: Effect of supply voltage on bridge fault behavior [89].

much wider range of physical bridge ($R_{sh} \in [R_{2B}, R_{3B}]$). In other words, a bridge with $R_{sh} = R_{3B}$ will cause a logic fault at Vdd_B but not at Vdd_A. This result accents the need for using multiple V_{dd} settings during test when bridge faults are targeted.

2.1.2 Resistive open defects

Open defects are common in deep-sub-micron CMOS. They refer to unconnected nodes in a manufactured design that were connected in the original design (Fig. 2.6).

Open defects can be classified as full or strong opens with resistance greater than

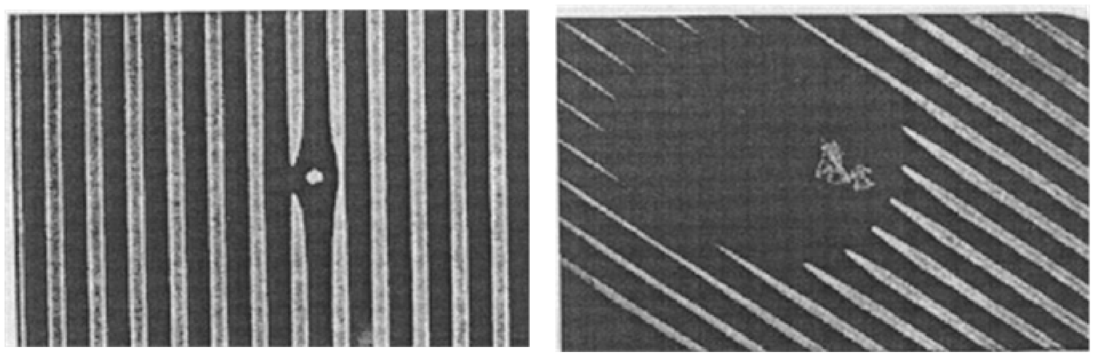


Figure 2.6: Examle open defects [165].

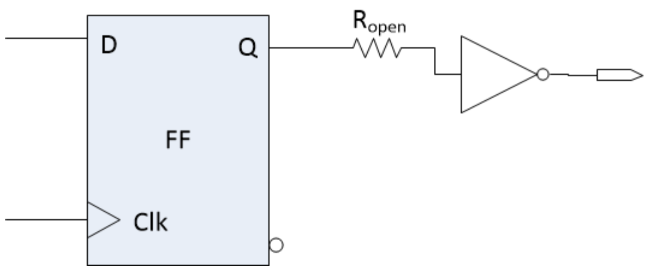


Figure 2.7: A typical resistive open fault model.

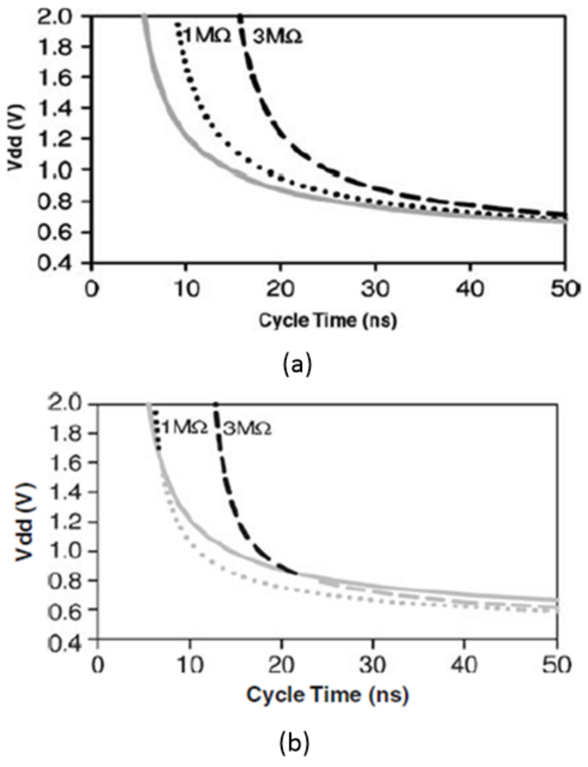


Figure 2.8: Comparison of path delays due to resistive open [90].

$10Mohm$ and resistive or weak open with resistance less than $10Mohm$ [170]. Strong open causes logic failures that can be tested using static tests and it is not V_{dd}-dependent [170]. Resistive open can be modelled as a resistor between two unconnected nodes. The inductive/capacitive component is limited and can be neglected for simplicity [90], [91]. A typical resistive open fault model is shown in Fig. 2.7.

The resistive open shows timing-dependent effects and it is screened using delay tests, namely tests that are used to catch defects that create additional than expected delay and thereby cause a malfunction of the DUT [90]. In delay fault testing, a defect is detectable only when it causes longer delay than that of the longest path in a fault-free design [171]. Fig. 2.8a and 2.8b show the delay caused by two different resistive opens, modelled as $1Mohm$ and $3Mohm$ resistances, in the longest and a short path of DUT, when different power supply settings are used ($V_{nominal} = 1.8V$).

The solid gray line in the first graph (Fig. 2.8a) depicts the delay of the longest path in a fault-free design and at various voltage settings. As it is illustrated, the additional delay added to the expected delay due to the two modelled open defects increases as the supply voltage becomes higher (up to $2V$) while higher delay is observed at $3Mohm$ than $1Mohm$. The graph in Fig. 2.8b shows that in the shorter path, the delay due to $1Mohm$ resistance, even at the higher supply voltage, is detected marginally and it becomes undetectable at lower V_{dd} settings. Correspondingly, the $3Mohm$ defect resistance is detected up to $0.9V$ and then it becomes undetectable too. The authors of [171], based on the above mentioned results, concluded that resistive open defects show better detectability at higher voltage settings and become undetectable at low ones. Similar observations were reported in [91]. However, it was showed in [90] that, in some cases, resistive open defects are better detectable at low voltage settings. In addition, in [91], the effect of the transmission-gate open and the resistive open defects upon the delay behaviour of designs operating at multi-V_{dd} settings was studied. The experiments showed that as the supply voltage setting is reduced, the transmission gate opens tend to and finally behave as stuck-at fault at lower V_{dd} settings. Similar observations were reported in [172]. The aforementioned findings show that interconnect resistive opens, with exceptions, are better detectable at higher voltage settings while transmission gate opens are better detectable at lower voltage settings. Therefore, high fault coverage is ensured by performing tests at more than one power supply voltage.

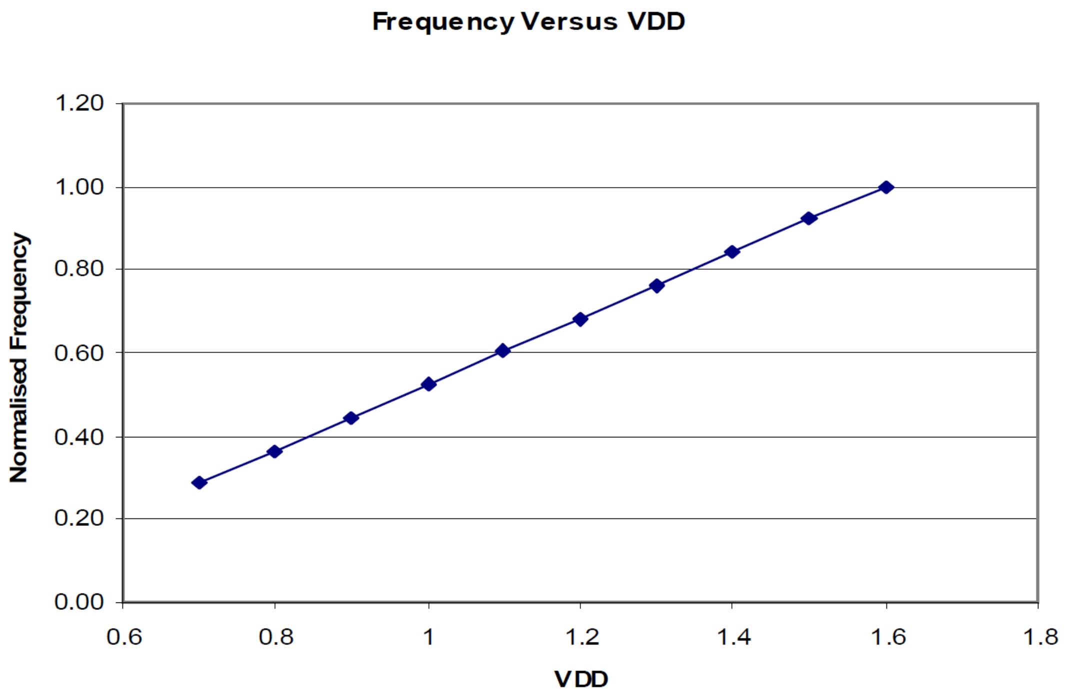
2.1.3 Multi-V_{dd} test and scan shift frequencies

Testing at different voltage levels may assume different maximum scan frequencies, which depend on the voltage levels (DUT can be tested using high scan frequencies at high voltage levels, and vice versa). The relationship between test speed and supply voltage has been studied initially for the needs of very-low-voltage testing [173][174][175]. They concluded that since the propagation delay of a CMOS gate becomes much longer at a reduced supply voltage, the test speed depends on how the critical path delay of a circuit changes as the supply voltage is reduced. Moreover, they proceeded in a delay - voltage relationship analysis and proposed a number of expressions that calculate the critical path delay at any voltage [173].

In multi-V_{dd} test, the scan shift frequency is limited by three factors, namely the tester capability, the power constraints during scan operation and the scan chain capability. The scan chain capability is reduced when the supply voltage is reduced [91]. The equation in Fig. 2.9a from [176] gives the operating frequency that should be used at any V_{dd} in order to meet the timing requirements of the DUT. Using the K constants for $0.18\mu m$ CMOS that are given in [176], the relation between the circuit supply voltage and frequency was derived for a commercial processor[55]. The bulk-source voltage V_{bs}, was set to zero since the effect of body biasing is not considered. The value of the threshold voltage, V_{th1} was given as $0.359V$. The normal processor operating voltage has been set between 1.2 to 1.6V. Fig. 2.9b shows the relation between the supply voltage and normalised frequency. At 1.2V, the operating frequency is 70% of that at 1.6V.

$$f = (L_d K_6)^{-1}((1+K_1)V_{dd} + K_2 V_{bs} - V_{th1})^{\alpha}$$

(a)



(b)

Figure 2.9: Supply voltage versus frequency for a commercial Processor [55][91].

2.1.4 Multi-V_{dd} test methods

In subsections 2.1.1 and 2.1.2, it has been shown that testing in multiple voltage settings is required to achieve high defect coverage of resistive bridge, transmission gate open and resistive open defects. This pathway leads to large test data volumes, which, in turn, may have a detrimental effect on the overall cost of test. To limit the economic impact of testing in this case, the test developer should keep the number of voltage settings required during test to a minimum and maximize the parallel execution of the required tests. In the first case, the test data volume and consequently the test time are decreased while in the second case the test time is directly reduced. The ultimate goal in both cases is spending test time equal to the one needed in the single-V_{dd} case. Furthermore, power consumption should be kept in nominal levels during testing in order to avoid unpredictable yield loss. A number of test methods targeting the above mentioned objectives are presented below.

The Test Point Insertion (TPI) method presented in [89] aims to reduce the required voltage settings in case of resistive bridging. In TPI, test points are used to provide additional controllability and observability at the fault-site to detect resistance intervals at the desired voltage setting, which are otherwise redundant and therefore helps reducing the number of test V_{dd}(s). The experimental results in [89] show that TPI can be used to reduce the number of V_{dd} settings during test, without affecting the defect coverage of the original test, thereby reducing test cost. However, the TPI scheme cannot guarantee single-V_{dd} test in most of cases.

The Gate Sizing (GS) method [94] [177] has similar objectives as the TPI. It targets resistive bridge defects that cause faulty logic behaviour to appear at a non-desired test voltage setting and tries to expose the same physical resistance at preferred test V_{dd}. This is achieved by adjusting the drive strengths of gates driving the bridge, such that higher resistance is exposed at the desired V_{dd} setting. The drive strength of the gates driving the bridged nets can be adjusted to increase the voltages on the bridged nets (e.g. V_O^{N1} in the example Fig. 1.35). This increase in voltage level can help expose maximum resistance at the desired V_{dd} setting, thus reducing the number of test voltage settings.

Multi-V_{dd} designs with multiple voltage islands use level shifters to communicate logic values across logic blocks that operate under different voltage settings [178]. In the multi-voltage-aware scan cell ordering technique [179], scan cells operating under the same voltage levels are connected together, so that the number of the required level shifters to communicate the test data from one scan cell to another is minimized. Furthermore, power dissipation is reduced due to the minimal use of the level shifters during test. Experiments were conducted using industrial design with four voltage domains and it was shown that multi-voltage-aware scan chain ordering succeeds 93% reduction in the number of level shifters, in comparison to scan chain ordering technique, which connects physically closer scan cells without considering its operating voltage.

In [180], a daisy-chaining scan approach is adopted to efficiently utilize tester resources and reduce test cost for multi-V_{dd} designs with multiple voltage islands. It incorporates bypass multiplexers in the TAM, so that specific power domains can be tested, according to the needs of the test process. An example of the method is shown in Fig. 2.10.

In the above example, let us consider a particular power mode, where power domains C and D are ON, while A and B are OFF. Then the upper multiplexers go in bypass mode, while the lower ones are in pass-through mode. This forms a scan chain between input SI, the lower multiplexers and output SO. The upper (bypass) multiplexers are placed on always-on power domain.

The Power Managed Scan (PMScan) method [181] proposes voltage scaling during test to provide a trade-off between test application time and test power. This is achieved by modifying the voltage regulation circuitry (used for adaptive voltage scaling) such that scan-shift operation meets acceptable timing, while supply voltage during scan shift is reduced. The voltage regulation circuitry changes the supply voltage to nominal during scan capture mode to ensure at-speed testing. The experiments showed that on average, this method reduces significantly the consumed dynamic and leakage power.

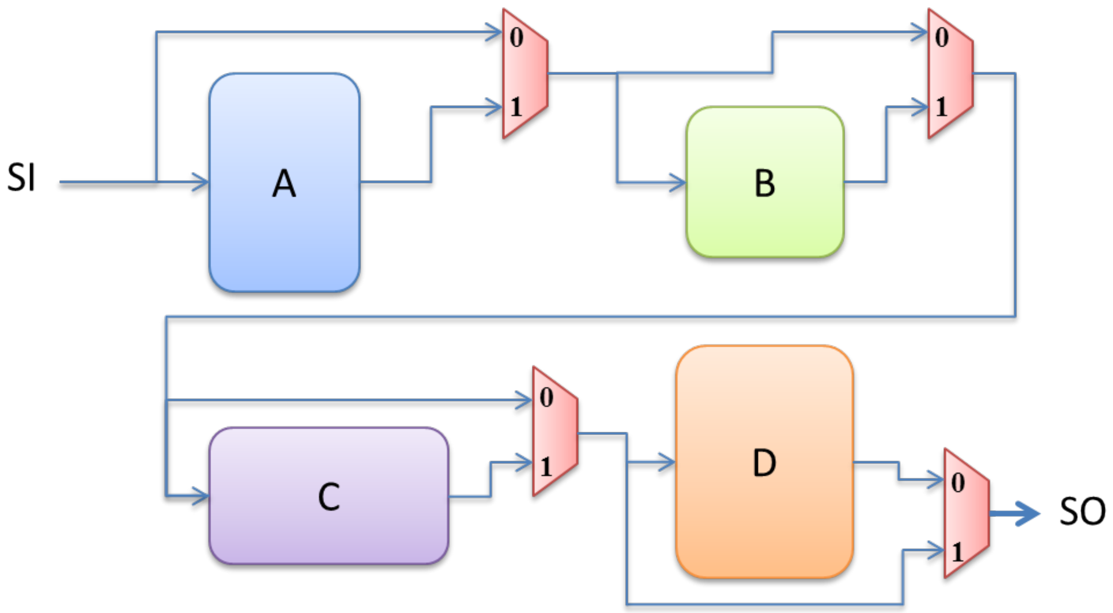


Figure 2.10: Example of the power - aware daisy-chaining scan path technique.

2.2 Test Scheduling

2.2.1 Basic principles

Let us consider a multi-core SoC with N_c cores $C_1, \ldots, C_{N_c}$ where each core C_i is wrapped so that it constitutes a testable unit. The scan chains at each testable unit C_i are formed into wrapper chains, and given the test patterns and scan frequency, each testable unit C_i is associated with a test time T_{C_i}. A TAM is available in the SoC that consists of N_B buses $B_1, \ldots, B_{N_B}$ connected to one or more cores. The objective of test scheduling is to guide the test application such that the overall test cost, which is directly related to test application time, is minimal.

Fig. 2.11a illustrates an example SoC with 4 cores, A, B, C, D, connected to a TAM. The TAM consists of two test buses B_1 and B_2. The bus B_1 has width 1 bit-line and is connected to cores A and B, while bus B_2 has width 2 bit-lines and transfers test data to cores C and D. Fig. 2.11b shows a possible test schedule for the example SoC. Each core test is depicted as a rectangle, where the vertical and the horizontal side represent the available bus width and the required test time, correspondingly.

A number of approaches have been proposed for test scheduling. In [185][186], the proposed methods target circuits with blocks of logic that can be scheduled independently. Early work on test scheduling for modular designs was performed by [148][150]. Test scheduling algorithms assuming reconfigurable core wrappers were proposed in [81], [82], [109], [151] - [153].

Fig. 2.12 shows a larger example of TAMs and the tests associated with each TAM for the ITC'02 design d695 [187]. The given TAM width of 64 is partitioned into five TAMs of width 3, 5, 17, 18, and 21. As it is illustrated, for example, core 6 has a dedicated bus

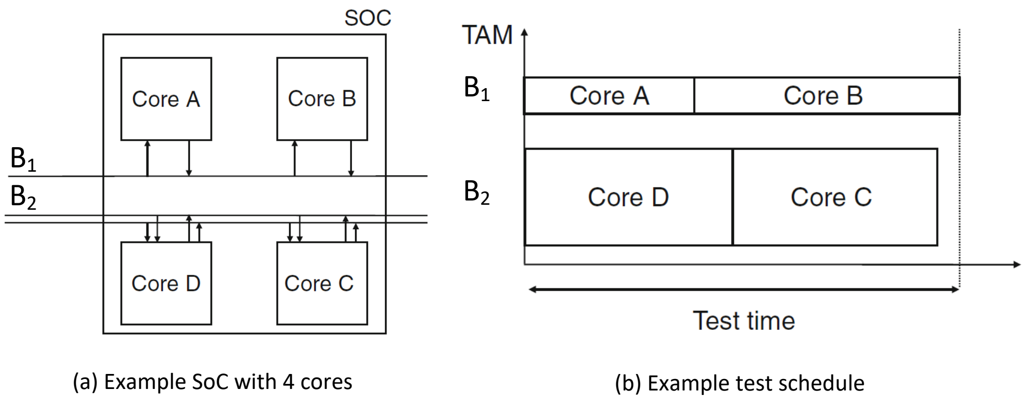


Figure 2.11: Test schedule for an SoC with 4 cores and a TAM with 2 buses.

and it can be tested immediately, while cores 2, 3 and 8 share a bus and they are tested in sequence.

The work on test architecture design and test scheduling, often, takes a given test architecture and optimizes the test schedule without taking into account the actual placement of cores in the system. In practice, it means that modifying the circuit slightly, and replanning the test, leads to potentially costly rerouting of TAMs. On the other hand, in [85], it is assumed a given floor-plan where each core is given x and y coordinates. The optimization function optimizes both the test application time and the cost of additional TAM routing.

2.2.2 Power-aware test scheduling

The objective of power-aware test scheduling is to define a test schedule, the order in which the cores are to be tested, such that a cost function, often related to test application time, is minimized while ensuring that certain power constraints are met. We consider a multi-core SoC with N_c cores $C_1, \ldots, C_{N_c}$ where each core C_i is wrapped so that it constitutes testable unit. The scan chains at each testable unit C_i are formed into wrapper chains, and given the test patterns, scan frequency and power consumption of each core, each testable unit C_i is associated with a test time $T_{C_iF_{scan}}$ and power P_{C_i}. A TAM is available in SoC that consists of N_B buses $B_1, \ldots, B_{N_B}$ connected to one or more cores. In addition, an upper bound P on power consumption of SoC is added to the problem formulation where P is defined by a model of power (e.g. average or peak power).

In [189], when core C_i is tested, it consumes power corresponding to P_{C_i} during a period of time $T_{C_iF_{scan}}$ and while C_i is not tested, the power dissipation is zero. The cores are grouped into sessions $S_1, S_2, \ldots S_n$. Cores assigned to the same session S_i are tested concurrently, and no new test can be started until all tests in current session are completed. The optimization objective is dual. First, the defined test schedule, assigning cores to sessions, should minimize the test application time. Second, in order to minimize

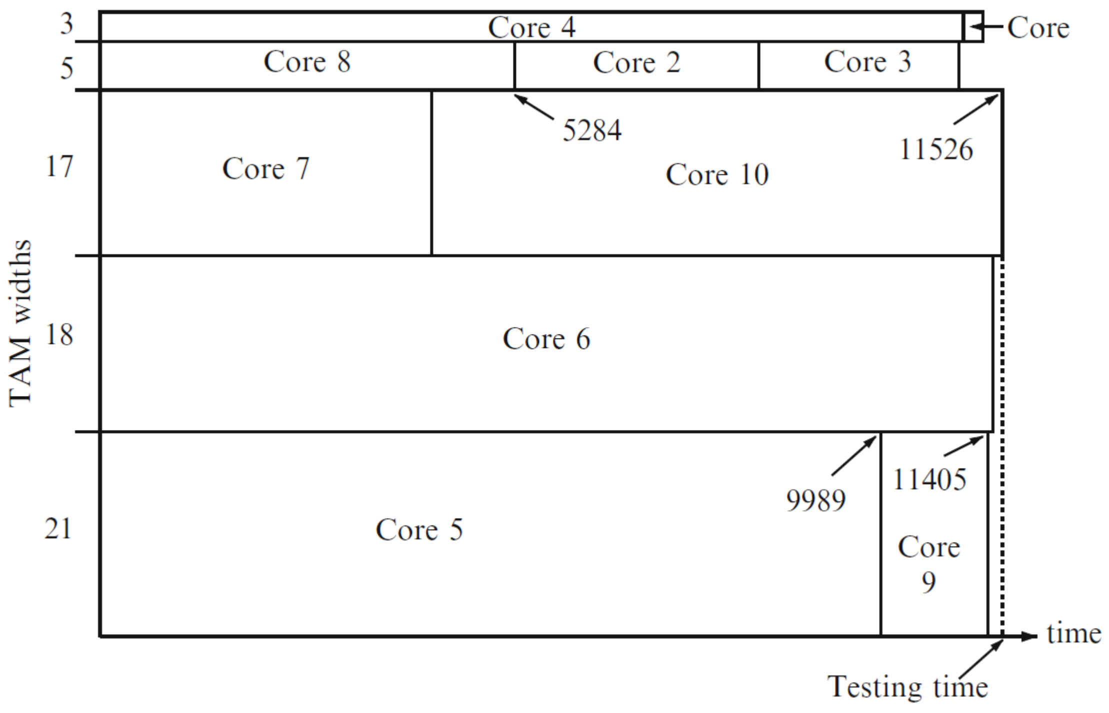


Figure 2.12: An example of a test architecture and a test schedule [188].

the routing overhead of added controller lines from the BIST controller to the cores (required to start the testing), cores that can share control lines, i.e., they are physically close, are to be grouped in the same test session. The defined test plan is not allowed at any time to consume more power than P. A test schedule derived from this work is shown in Fig. 2.13.

The authors in [190] assume, as in [189], that each core C_i is associated with a fixed test time and a single fixed test power. However, in this work it is assumed that there may be conflicts among the cores. In order to handle test conflicts, the problem is reformulated as a graph problem. A test compatibility graph $TCG(V, E)$ is used, and cores are vertices (nodes) and compatibility is modelled through the edges where an edge between two tests means that the corresponding cores can be tested at the same time. A power compatibility graph PCG is used to derive power compatible alternatives. An example of a PCG is shown in Fig. 2.14 where each node is a test and attached to each node is a test time and a test power consumption. The test schedule defined in [190] produce better results when compared to [189]. The work presented in [148] used the same assumptions as [190] and formulated an heuristic to schedule the tests. This work managed to define a test schedule with even better test application time than [190].

In [191][192][193][109][194], a TAM wire/wrapper-chain requirement added to the power - aware test scheduling. Specifically, in these works each core C_i is associated with a value on test time $T_{C_iF_{scan}}$, a value on test power consumption P_{C_i} and a value on Wrapper Parallel Port (WPP) width W_{C_i} (illustrated in Fig. 1.36). The objective is to define a test schedule where the box for each core C_i is assigned a start time such that

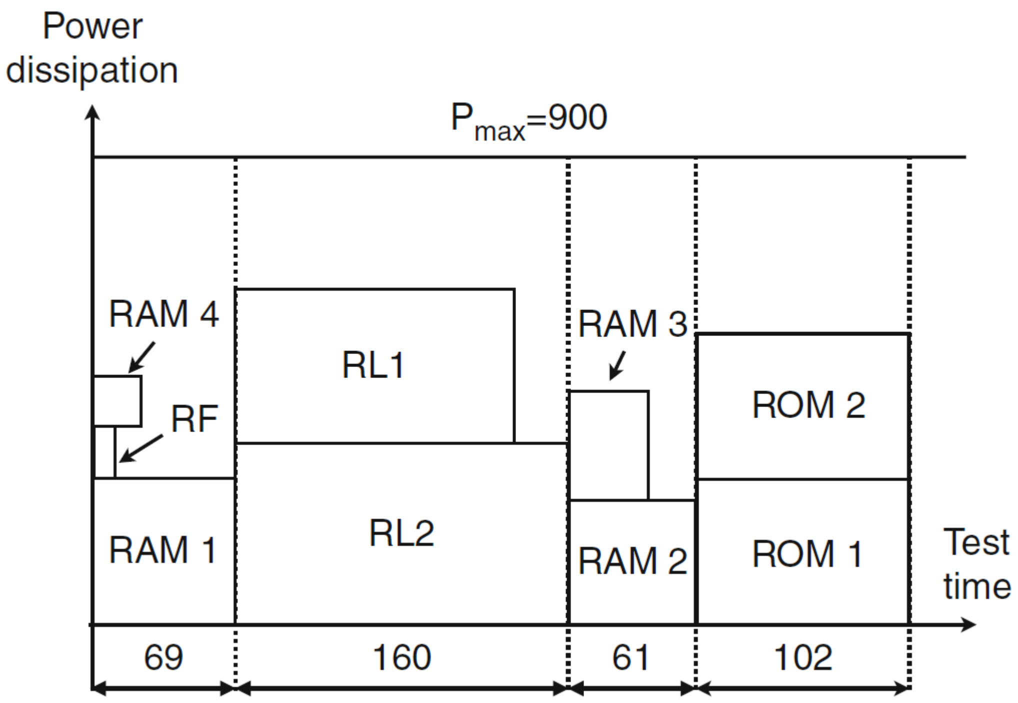


Figure 2.13: Test schedule derived by method [189].

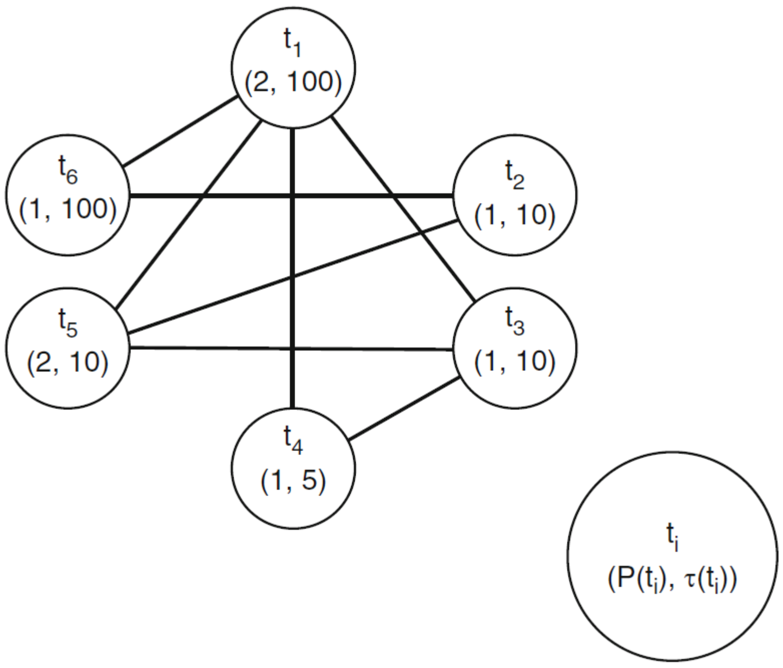


Figure 2.14: A power compatibility graph derived by method [190].

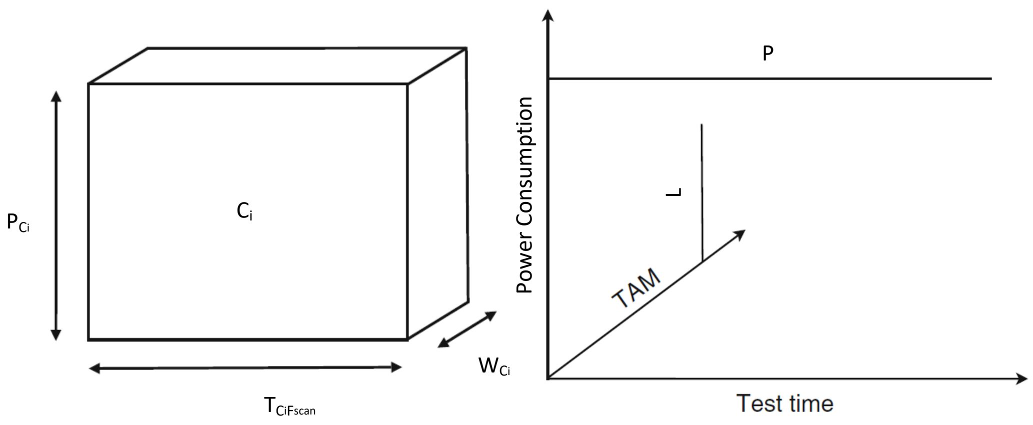


Figure 2.15: Example of power constraint [127].

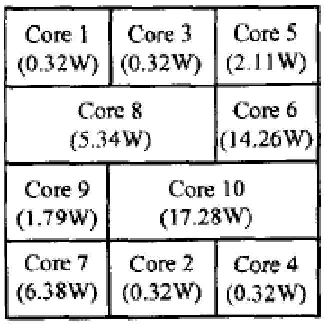
constraints on power consumption P and TAM width L are not violated at any time, as depicted in Fig. 2.15. Keeping aside the fact that packing three-dimensional boxes is not a trivial task, there are two paths that can be followed in this test scheduling approach. First, the number of wrapper chains at each core, namely the W_{C_i} is considered fixed so that the whole procedure can be simplified. Second, the number of the assigned wrapper chains W_{C_i} in each core C_i can vary, altering accordingly the test time of the core C_i, i.e. higher W_{C_i} means lower test time and vice versa. Naturally, the second option leads to more optimized test plans as shown in the above works.

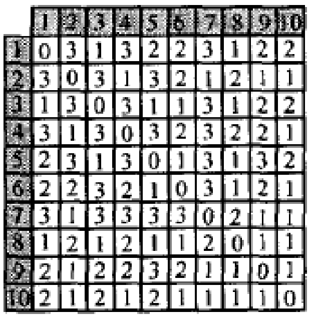
Moreove, a number of approaches have been proposed to address test infrastructure while considering test power consumption. In [150], the design of test architectures under place-and-route and power constraints was proposed. In [195][196], they are explored a number of test scheduling algorithms and in [197], it is proposed a technique to define the test resources in the system.

2.2.3 Thermal-aware test scheduling

The objective of thermal-aware test scheduling is to define a test schedule, such that the test application time is minimized while ensuring that certain thermal-oriented requirements are met. We consider a multi-core SoC with N_c cores $C_1, \ldots, C_{N_c}$. These cores are placed according to a floorplan, denoted with FP. Each core C_i is wrapped so that it constitutes a testable unit. The scan chains at each testable unit C_i are formed into wrapper chains, and given the test patterns, scan frequency and power consumption of each core, each testable unit C_i is associated with a test time $T_{C_iF_{scan}}$ and power P_{C_i}. A TAM is available in SoC that consists of N_B buses $B_1, \ldots, B_{N_B}$ connected to one or more cores. In addition, an upper bound TL on maximum temperature of SoC is added to the problem formulation that either should not be exceeded or it should be minimized.

In [202], two thermal-aware test scheduling methods are proposed to reduce the tem-

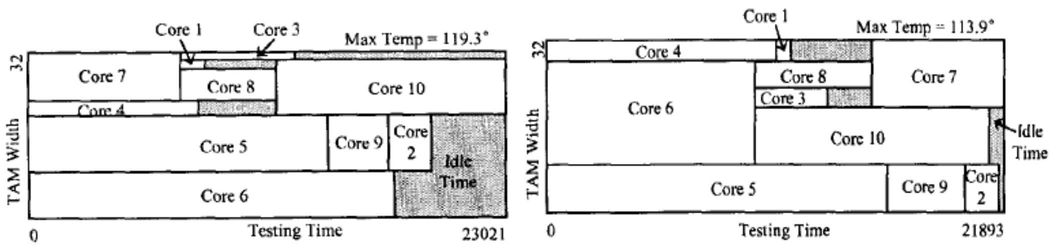




	1	2	3	4	5	6	7	8	9	10
1	0	3	1	3	2	2	3	1	2	2
2	3	0	3	1	3	2	1	2	1	1
3	1	3	0	3	1	1	3	1	2	2
4	3	1	3	0	3	2	3	2	2	1
5	2	3	1	3	0	1	3	1	3	2
6	2	2	3	2	1	0	3	1	2	1
7	3	1	3	3	3	3	0	2	1	1
8	1	2	1	2	1	1	2	0	1	1
9	2	1	2	2	3	2	1	1	0	1
10	2	1	2	1	2	1	1	1	1	0

(a) Example SoC floorplan (b) Distance between cores

Figure 2.16: Floorplan of an example SoC and inter - core distance [202].

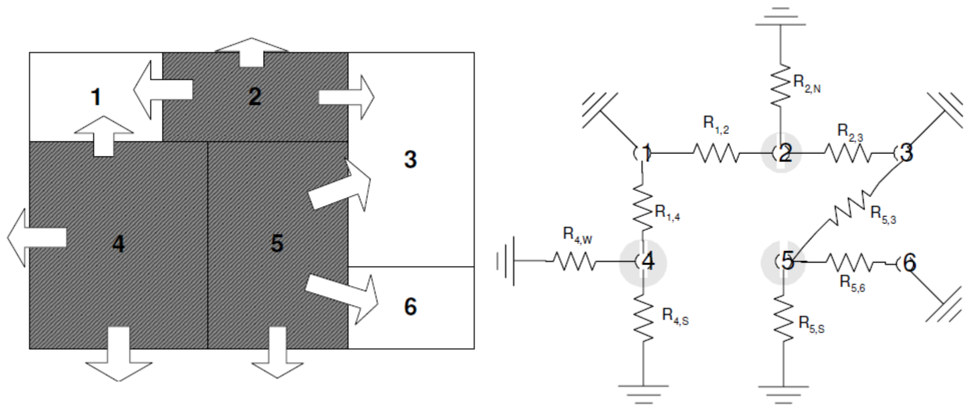


(a) Base case (b) Progressive weighting method

Figure 2.17: Base case vs progressive weighting [202].

perature in the hotspot of an SoC. Hotspot is called the spot in floorplan that presents the maximum temperature in a given time. The first method is based on the fact that a major factor affecting heat transfer between two cores is the distance between them. If two cores are geometrically close to each other, then their temperature difference can cause significant heat transfer between them and quickly change their temperatures. Therefore, if the concurrent scheduling of two hot cores that are close to each other can be avoided, then the maximum temperature that is developed on-chip during test can be reduced. Fig. 2.16 shows an example floorplan and a table with the distances between cores. The later is used to determine the test schedule.

The second method minimizes the hotspot in an SoC using a progressive weighting technique. Each core C_i is associated with a thermal weight. A large weight indicates that the core is more likely to have a high temperature during test. During the scheduling process, the total weight of cores at any point of time is maintained below a predetermined threshold. Thus, it is ensured that hot cores are not scheduled simultaneously. Fig. 2.17 shows the results of the two methods. Clearly, the second method provides with better results.



(a) Example floorplan (b) Test session thermal model

Figure 2.18: Example floorplan and thermal model [160].

The work in [160] follows the basic principles of the base case that we examined in [202]. However, thermal awareness is achieved via a low complexity test session thermal model which is shown in Fig. 2.18. In the thermal model that is depicted in Fig. 2.18b, a small equivalent thermal resistance associated with an active core C_i means good heat exchange between the core and the ambient, which predicts a low core temperature during test. On the other hand, a large equivalent thermal resistance associated with an active core means poor heat exchange with the ambient, and therefore signals a potential hot spot during test. According to the model, it is a derived a thermal characteristic for each test session that provides with a normalized means for selecting the appropriate core to be added to the test session.

In [155], a Mixed Integer Linear Program (MILP) is used to find an optimal solution to the test scheduling problem. The MILP formulation minimizes the total test schedule time while constraining the maximum temperature of any core and ensures that no resource constraints are violated. A thermo-resistive model has been incorporated into the MILP-based formulation. In addition, it is proposed an heuristic based on a lookahead scheme for large problem instances, namely SoCs with a large number of embedded cores.

In [157], it is proposed a partition-based thermal-aware test scheduling algorithm. Specifically, each core test is divided into a number of stages, where each stage has different average power consumption. The power consumption of the test that targets hard-to-detect faults can be much larger than the power consumption of the test that targets easy-to-detect faults. Thus, it is considered that the partitioning of the tests to testing stages can benefit the power and thermal constrained test scheduling. A simple motivation example is illustrated in Fig. 2.19. Two tests, T1 and T2 are to be scheduled. T1 has a test power consumption of 10 and test length of 20. T2 has a test power consumption of 15 and test length of 20. Supposing a power constraint of 20, the final schedule without partition is shown is Fig. 2.19c and total test length is 40. By partitioning T1 and

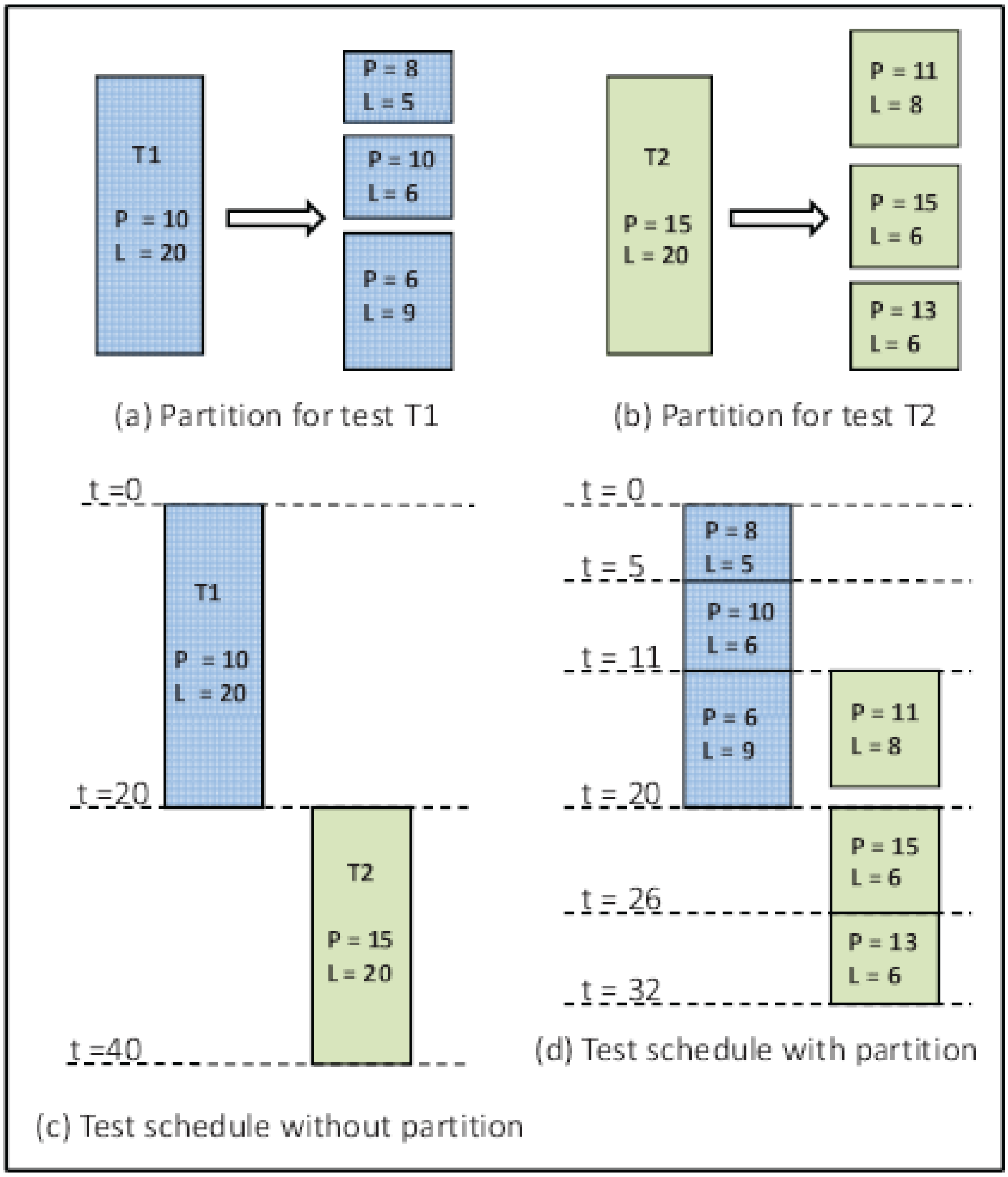


Figure 2.19: Motivation example of method [157].

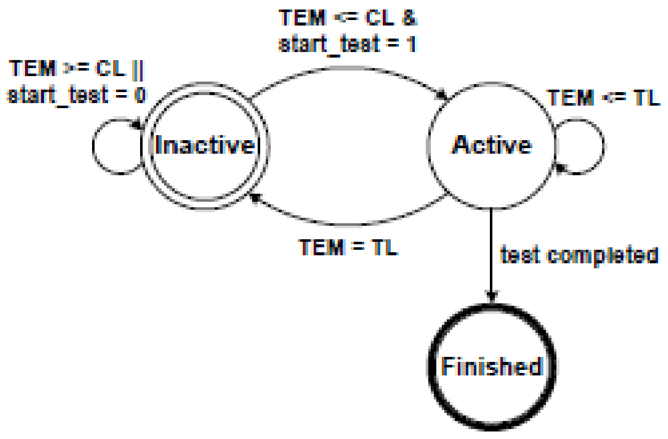


Figure 2.20: The finite state machine model [157].

T2 into three partitions, each partition will have different power and different length, as shown in Fig. 2.19a and 2.19b, where P means test power consumption and L means test length. Under the same power constraint of 20, the test schedule with partition is shown in Fig. 2.19d, which has a smaller total test length of 32. The reduction of total test length by partition comes from overlapping between two test partitions with lower power consumptions.

In [203], a Finite State Machine (FSM) model has been developed to control test set partitioning and interleaving so that they can minimize the test application time while the temperatures of cores under test are kept below a given temperature limit during the entire test process. This model is shown Fig. 2.20. There are three states for a core, namely inactive, active, and finished, which correspond to the cases that the core is not being tested, the core is being tested, and the test application is completed on the core, respectively. When the test scheduling process starts, it is assumed that all cores are at the inactive state and their temperatures are equal to the ambient temperature. When a core is selected for test and the required test-bus bandwidth is allocated for the test, the state of the core moves from inactive to active. While test patterns are applied to the core, the temperature of the core increases. The state of the core remains active until the temperature reaches a given temperature limit or the test is completed. As soon as the test is completed, the state of the core moves from active to finished. Otherwise, when the core temperature reaches the threshold, the core state moves from active to inactive and remains unchanged until the core temperature decreases to a given stop-cooling temperature, from which a new round of state transitions between active and inactive is repeated until the test is completed. The test scheduling process terminates when all cores are at the finished state.

Chapter 3

Research Work

3.1 Research directions

This research targets the reduction in test application time of moderate, large and very large SoCs, that consist of embedded cores placed in multiple voltage islands operating at different voltage or frequency settings. The testing environment of an SoC consists of a variety of special features. The multi-V_{dd} nature of an SoC supporting advanced power management techniques imposes additional requirements too. The main aspects of the SoC testing environment, the required considerations due to multiple supply-voltage levels and voltage islands in an SoC and the targeted research areas and goals are presented below.

As SoCs designed in modular fashion are becoming increasingly common, manufacturing testing can be performed in a modular manner too (Fig. 1.33). In this case, the embedded cores in the DUT are only activated when they are tested. The gain is the ability to reduce the test application time and control the test power consumption.

Specifically, the modular test enables the test designer to plan the test order of the cores such that test time, power and thermal constraints are met. In our research, we consider SoCs that support modular testing.

In order to enable modular test, each embedded core in an SoC must be transformed to a testable unit. As shown, this is succeeded using a core test wrapper, or simply wrapper. Its role is dual. First, to isolate the core so that it can act as a stand-alone test unit. Second, to define the interface between the core and the infrastructure for test data transportation, the TAM, so that test access can be efficient. Nowadays, the SoC industry mainly uses the core test wrapper described by IEEE 1500 Standard for Embedded Core Test (SECT) (Fig. 1.35) [77]. This standard is similar to 1149.1 [78] in that its main objective is to standardize boundary test circuitry (wrappers) for cores. However, unlike 1149.1, it provides parallel access capability for a core. Thus, test application time for an SoC can be significantly improved. Furthermore, in contrast to 1149.1, where control signals are mainly generated by a finite state machine that is controlled by a single input, in 1500 standard the control signals can be directly applied to a core, thus providing more test flexibility. In our research, we consider SoCs that use wrappers based on the 1500 standard.

The wrapped cores, or testable units, in an SoC do not have direct access to SoC pins. In order to transport test data (or test vectors or test stimuli) and test responses, to and from embedded cores, the TAM is needed (Fig. 1.36). To reduce the cost of DFT structures embedded in the SoCs, the TAM resources are typically shared among different cores. However, sharing of TAM resources leads to test conflicts that are resolved using TAM optimization and test scheduling techniques. Many test scheduling techniques have been proposed in the literature, as we have already described in section 2.2. All these methods aim to produce the most efficient TAM design and test schedule for minimizing the SoC test-application time. Unfortunately, these methods cannot be directly applied on multi-V_{dd} designs with voltage islands as they do not consider the test requirements and additional constraints imposed by them [88]. Test scheduling for multi-V_{dd} designs, especially those that consist of multiple voltage islands, is considerably more challenging than traditional test scheduling for single-V_{dd} designs. On the other hand, the inherent characteristics of multi-V_{dd} designs and the associated test requirements allow for increased parallelism in the test scheduling. Time Division Multiplexing (TDM) [95] exploits this potentiality. We propose a TDM scheme along with sophisticated test scheduling techniques, able to effectively limit down the test cost that is introduced by the multi-V_{dd} aspect and the voltage islands of an SoC.

Most production lines employ multi-site test processes to increase the ATE utilization and decrease the overall time for testing a large volume of chips [96] - [106]. Multi-site testing exploits the available ATE channels to concurrently test multiple chips; therefore, in this case, the minimization of the time needed for testing a single chip is not the primary optimization goal [107]. Instead, efficient exploitation of the ATE channels to increase the number of dice that are tested in parallel is more beneficial for a production batch of SoCs [108]. The number of SoCs that can be tested in parallel depends, among other parameters, on the availability of ATE channels [96] - [106]. For a fixed number of ATE channels, a dual objective must be pursued to optimize multi-site testing, namely the minimization of the number of ATE channels needed per SoC and the minimization of the test time per chip. Our research shows that the TDM is an effective candidate for succeeding the above goals, when multi-V_{dd} / multi-island designs are tested. Furthermore, we extend the TDM architecture and we form a Space and Time Division Multiplexing (STDM) scheme that is optimized for multi-site testing.

As shown in section 2.2, the effectiveness of a test scheduling method for minimizing test time depends on the test access mechanism that is used in an SoC. Single-V_{dd} TAM optimization techniques consider neither the highly constrained test environment of multi-V_{dd} SoCs nor the benefits provided by TDM, therefore they are not suitable for multi-V_{dd} SoCs. We propose the first TAM optimization technique for multi-V_{dd} SoCs. Special care has been given in the TAM optimization of large and very large SoCs.

Excessive power consumption during test, increases the overall chip temperature, and in many cases, it creates localized overheating. This phenomenon is called hotspot and it causes permanent damage to silicon, reliability failures and eventually yield loss. Thermal aware testing resolves thermal−related issues by reducing the temperature at hotspots in an SoC. To this cause, we propose a thermal and delay aware X−fill method that uses the chip layout information to (a) remove hotspots at critical nets and create a thermal safe neighbourhood around them and (b) exploit scan cells in non - critical nets to increase the un−modelled defect coverage of the generated test vectors.

The organization of the chapter is as follows. Section 3.2 lists the main objectives of this research work. Section 3.3 describes in detail the TDM method and effective TDM test scheduling techniques for DVFS-based SoCs. Section 3.4 presents and analyses a space- and time- division multiplexing method able to optimize multisite testing. Sections 3.5 presents the first TAM optimization technique for DVFS-based SoCs. Finally, section 3.6 presents a critical path-oriented and thermal-aware X-fill technique able to provide with high unmodelled defect coverage.

3.2 Research objectives

A succinct thesis statement is as follows: this research develops an efficient, integrated and computational-friendly methodology to minimize test time of moderate, large and very large Multi-core / Multi-V_{dd} SoCs with voltage islands while power constraints are met.

The following research objectives were identified for this work:

- To enhance the TDM test scheduling method, a highly effective technique targeting Multi-core/Multi-V_{dd} SoCs, with a new, productive set of tools that will allow for power awareness, TAM optimization, scalability to deal with large SoCs and efficient multi-site use.

- To develop a novel method that offers high multi-site test efficiency so that test time can be reduced for an entire production batch of SoCs,
- To develop a novel, highly parallelized technique to optimize the TAM a) for minimizing test-time when TDM is used to schedule tests, and b) for multi-site testing. The potential effect on test time due to power constraints is explored as well.
- To develop an active set of tools for the proposed TAM optimization method that will enable its effective application upon large Multi-core/Multi-V_{dd} SoCs. The potential effect on results due to power constraints is explored too.
- To develop a thermal and delay aware X−fill method that remove hotspots in critical areas of the chip while offering high un−modeled defect coverage.

3.3 Time-Division Multiplexing for testing DVFS-based SoCs

3.3.1 Introduction

Power consumption is a major burden for electronic systems [68]. DVFS offers an effective trade-off between power consumption and system performance as it adaptively adjusts the power supply-voltage and the frequency depending on the workload of the SoC [44, 45]. Further reductions are achieved by partitioning SoCs into voltage islands with separate supply rails and unique power characteristics [58, 45, 64, 63]. Various methods address design challenges in systems consisting of multiple voltage islands [65], [44, 66, 67]. DVFS has been implemented in several state-of-the-art processors [52, 53, 55, 56, 176].

The adoption of DVFS has a significant impact on test strategies for multi-core SoCs. Specifically, defects are manifested in different ways at the various supply-voltage levels of a Multicore/Multi-V_{dd} SoC [89], [90]. Therefore, fault-free behavior must be ensured at each voltage or frequency setting that the SoC uses during normal operation. It was shown in section 2.1 that certain resistive bridge and open defects can be detected only at specific supply voltage levels. These methods clearly highlight the need to test a multi-V_{dd} SoC at multiple voltage or frequency settings in order to assure fault-free field operation. A drawback of testing at different voltage levels is an increase in test time and thus, in test cost. To reduce test cost, a method is presented in [94], which modifies a carefully selected part of the core under test to achieve the detection of all detectable resistive bridging faults at a single voltage level. Consequently, it eliminates the need to carry out time-consuming and costly testing using multiple test voltage levels. While this technique is effective in reducing the test cost for resistive bridging faults in multi-V_{dd} designs, it does not target other types of defects. Moreover, it imposes design changes that are often not acceptable in practice. Therefore, to avoid design changes and detect a wider range of defects than those modelled as bridging faults, testing at multiple voltage levels is necessary for fault detection. Such a requirement increases considerably both the test time and, in many cases, the test data volume, thus the complexity of the test scheduling problem for the SoC is increased.

As shown in section 2.2, traditional test scheduling techniques, resolve conflicts caused by the sharing of TAM resources in single-V_{dd} SoCs, but they are not suitable for multi-V_{dd} designs [112]. Multi-V_{dd} designs impose additional constraints that do not exist in single-V_{dd} designs. At the same time, lower supply voltages reduce the maximum-allowable shift frequency for scan chains, which further increases test time. As a result, test scheduling for multi-V_{dd} designs is more difficult than that for single-V_{dd} SoCs. The test scheduling method proposed in [112] considers the additional constraints imposed by the use of multiple voltage settings, but it does not tackle the problem of using low shift frequencies at the lower voltage settings.

We propose a novel TDM architecture, which tackles the problem of reduced shift frequencies in DVFS-based SoCs. This solution is also useful for reducing test time when scan shift frequencies have to be reduced because of power concerns during scan testing. TDM uses the highest frequency supported by the SoC to shift multiplexed test data into the SoC, and then it demultiplexes and shifts the test data into multiple cores using lower shift frequencies. The TDM architecture is supported by an effective Integer Linear Programming (ILP)-based test scheduling method, which provides optimal results for SoCs of moderate size. For larger SoCs, a rectangle-packing method combined with simulated annealing is proposed, which offers near-optimal results. Finally, for scenarios that require very short run times, a greedy approach is proposed.

Our results lead to the counter-intuitive conclusion that the use of low shift frequencies can reduce test time when TDM is adopted. Therefore, in contrast to the standard optimization goal of minimizing the test time for each core, the proposed test scheduling methods target the maximum exploitation of the TAM bandwidth instead. Experimental results for two representative industrial SoCs highlight the clear benefits of applying the proposed technique to multi-V_{dd} designs.

The organization of the rest of the section is as follows. In Section 3.3.2, we present the motivation for this work. Section 3.3.3 describes the proposed TDM scheme and Section 3.3.4 presents the problem formulation. Sections 3.3.5, 3.3.6 and 3.3.7 present the three test scheduling approaches.

3.3.2 Motivation

Multi-V_{dd} SoCs require long test application times due to the high volume of tests that must be repeatedly applied at the various voltage levels. Test application time is dominated by the process of serially loading test data into the cores through scan chains, which are usually not designed to operate at the rated speed of the cores. As a result, the ATE transfers and loads test data to cores using a slow scan shift frequency, which increases the test time of the cores. Even in the case that the tester supports higher scan frequencies, this capability cannot be exploited, thus leaving tester potential underutilized.

The above problem is exacerbated when cores are tested at lower power-supply voltages. When the power-supply voltage is reduced, the scan frequency is also reduced to avoid scan-path delay violations. As a result, at lower power-supply voltages, the maximum-allowable scan frequency is also reduced, typically in proportion to the reduction in power-supply voltage as shown in Fig. 1.7.

In addition, as shown in section 1.1, a variety of cores are embedded in a typical SoC, that have different characteristics and run in different frequencies. Then, each core may support a different scan frequency too. Table 3.1 shows ten cores that were selected from the IWLS suite [232] and synthesized using commercial tools and the 45 nm Nangate technology [115]. For testing every core, the full scan design methodology was adopted. The cores were wrapped using the the IEEE 1500 Std. wrapper. For each core, test vectors targeting transition faults were generated using the Launch-on-Capture scheme. These test vectors were applied with increasing scan frequencies, until the timing simulation failed. This frequency is the maximum scan frequency supported by each core, and it is reported in the second and fourth row for each core in table 3.1. Let us consider an artificial SoC that consists of these ten IWLS cores and a local test bus that transfers test data to them. Let the bus be fed by a tester that supports a scan frequency equal to 400MHz. This capacity can only exploited by usb_funct, pci_bridge, wb_conmax and des_perf cores, while the remaining ones will leave the test bus capacity underutilized.

IWLS core	tv80	mem_ctrl	ac97	usb_funct	pci_bridge
Scan frequency	222	333	333	400	400
IWLS core	aes_core	wb_conmax	ethernet	des_perf	vga_lcd
Scan frequency	333	400	333	400	333

Table 3.1: Maximum scan frequencies in MHz for IWLS cores at the 45nm process technology.

Furthermore, due to low-pin-count and high multi-site configurations, testers usually conduct SoC testing using a single scan clock for the whole SoC. Thus, to avoid scan violations at any voltage setting, the lowest frequency for shifting test data has to be used, which corresponds to the slowest core and the lowest voltage level, and the test time increases even more.

Despite these limitations, cores that are tested at lower shift frequencies, and which share the same TAM resources, can potentially be tested concurrently, provided that there is a mechanism to multiplex and demultiplex their test data on the TAM resources. TDM offers such a mechanism. By using TDM, the test data can be transmitted by the tester at a higher frequency, while at the same time, they can be shifted into the scan chains of multiple cores at considerably lower frequencies, where these frequencies depend on the voltage setting used in each case. Not only does TDM exploit the gap between the shift

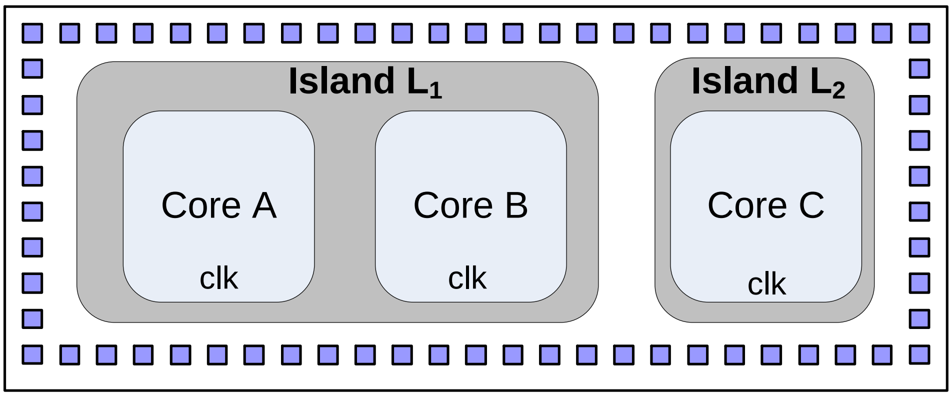


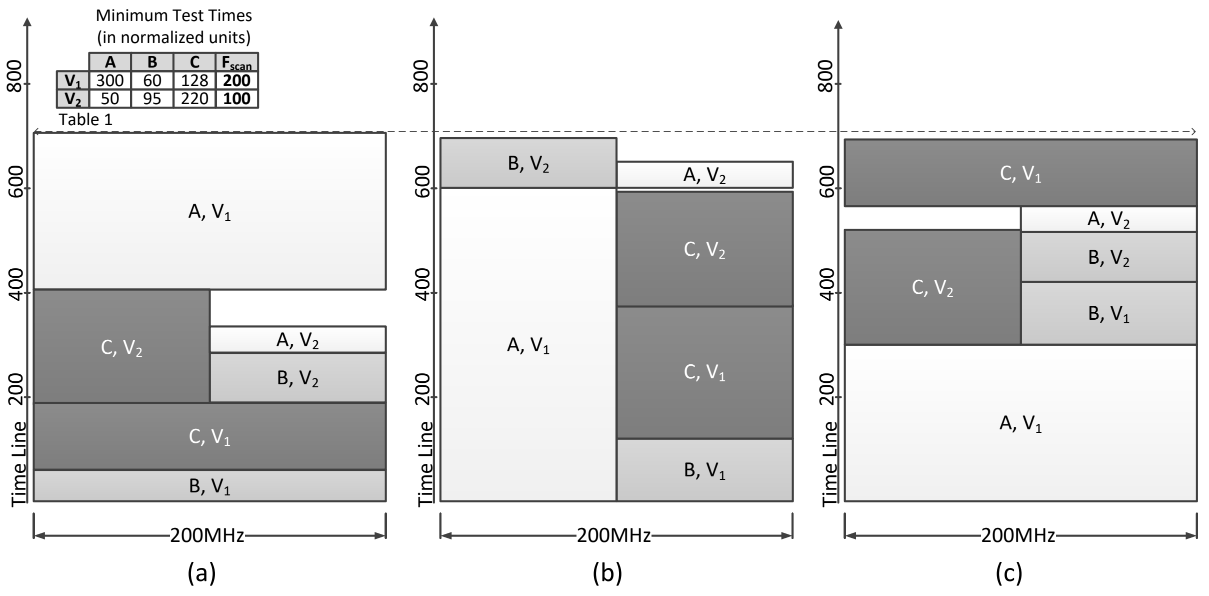
Figure 3.1: An example SoC with 3 cores and 2 voltage islands.

frequencies of the ATE and the cores, but it also exploits the differences between the shift frequencies of cores of different islands that are concurrently tested at different voltage settings.

Example 1. Let cores A, B, C in Fig. 3.1 be tested at voltage levels $V_1 > V_2$. Table 1 in Fig. 3.2 shows the test time for each core at the maximum allowed scan frequency at each voltage setting. Fig. 3.2a, Fig. 3.2b and Fig. 3.2c depict three different test schedules for one TAM resource. Every rectangle represents a test that corresponds to one core-voltage pair, and it has width equal to the scan frequency used and height equal to the time needed by the test at this scan frequency. The TAM resource is represented by a virtual bin that has width equal to the frequency supported by the test channel, in this example 200MHz, and unlimited height. The maximum height that is reached by the rectangles in the bin corresponds to the test time of the test schedule. As it is illustrated in Fig. 3.2a, the TAM resource suffices to test cores A, B, C at V_1 using the maximum rated frequency. In the same test schedule, core C is tested concurrently with cores A and B at V_2 using time multiplexing. Another alternative shown in Fig. 3.2b, is to use TDM to test cores A and B at V_1 concurrently, using a shift frequency equal to 100MHz, which is slower than the nominal shift frequency. In addition, core C, using time multiplexing, is tested in parallel with core A at V_1 and V_2 since it belongs to a different voltage island. Finally, cores A and B are tested concurrently at V_2. A third alternative is shown in Fig. 3.2c, where cores A and C are tested at V_1 using the full bandwidth of the TAM resource, while the remaining tests are multiplexed so that the test time can be minimized. Other alternatives can also be considered.

■

There exist many different scheduling scenarios that exploit the full capability of the tester for increasing the parallelism in loading test data into the cores, as depicted in Fig. 3.2. In addition, counter-intuitively and in contrast to what we expect, shifting test data into the scan chains at a lower than nominal frequency may be beneficial in



	A	B	C	F_{scan}
V$_1$	300	60	128	**200**
V$_2$	50	95	220	**100**

Figure 3.2: Three test schedules generated using TDM.

terms of ATE-channel-frequency utilization and this strategy may reduce the test time. However, the solution space is very large, and it is computationally challenging to find a good solution, especially when lower-than-nominal scan frequencies are also explored for reducing the overall test time.

In this work, we address the problem of scheduling tests in multi-V_{dd} SoCs assuming that each voltage setting imposes a different maximum shift frequency for each core. This problem is a generalization of the simpler scheduling problem that assumes a single shift frequency for all cores. The latter problem is NP-complete, therefore the scheduling problem being considered is also at least NP-hard. Besides the TDM architecture, we propose three different test scheduling approaches, each one offering different advantages:

1. The first approach is an ILP-based test scheduling approach. Even though ILP-based test scheduling is not scalable and it cannot provide optimal solutions for large SoCs, it is beneficial as it provides optimal or near optimal solutions for SoCs of small or moderate size. Moreover, it provides a good means to evaluate the performance of the heuristic methods.

2. The second approach is a rectangle-packing approach combined with a simulated-annealing optimization method. This method offers a trade-off between run time and performance (i.e., test time). In the general case, it offers test schedules that are close in terms of test time to the ILP-based approach, in less CPU time.

3. The last approach is a greedy approach. It offers inferior solutions to the previous two approaches, but it is very fast, and very practical for very large SoCs.

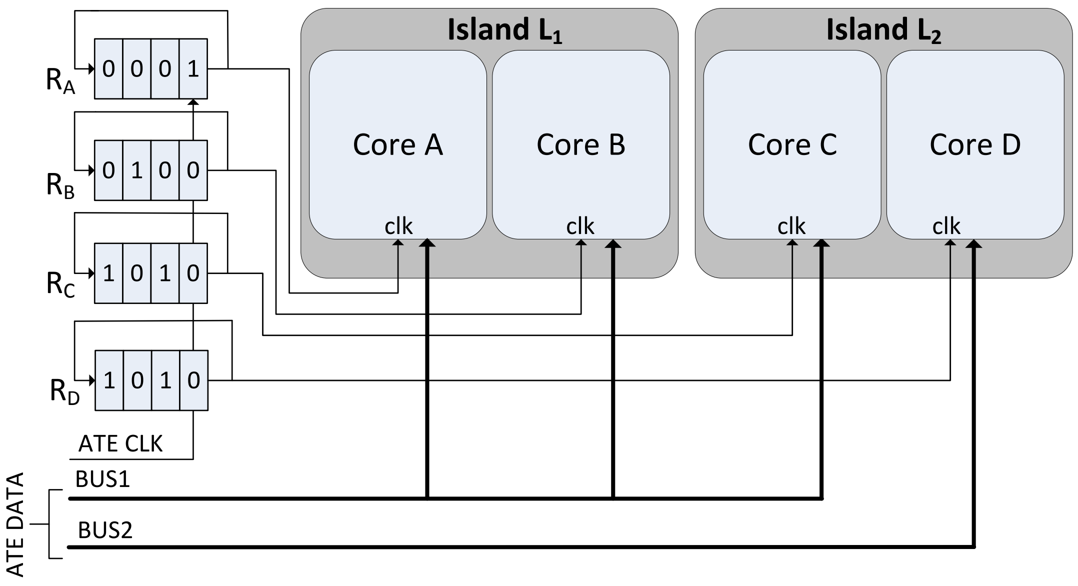


Figure 3.3: Proposed TDM scheme.

3.3.3 TDM scheme

Fig. 3.3 illustrates the proposed TDM scheme for an SoC consisting of cores A, B, C, D, and two test buses. There are two islands L_1, L_2, which support voltage settings V_1, V_2, V_3 and the nominal scan frequencies at these settings are $F, F/2$ and $F/4$, respectively. Voltage levels are generated using either of the following two ways: (a) external power supplies for the voltage islands, or (b) embedded power management with internal regulators. In either case, we assume that the SoC already embeds the necessary structures for functional purposes. Let the tester provide the ATE_CLK signal and the test data with frequency F. Each core is assigned one cyclical shift register with length equal to 4, which divides the scan frequency by a value equal to 1, 2 or 4. The scan frequency for each core is determined by loading the appropriate pattern into each register before the testing of the core begins. Every shift register is clocked with the fast ATE_CLK (frequency F) and provides a clock signal with frequency equal or smaller to F. The following example illustrates this method.

Example 2. Let us assume that cores A, B are tested at voltage V_3 and C, D are tested at V_2. Then, the highest frequencies that can be used for A, B, C, D are $F/4, F/4, F/2, F/2$, respectively. In order to provide scan frequency of $F/4$ to core A, register R_A in Fig. 3.3 is loaded with the pattern "0001". Then, during every 4 successive cycles of ATE_CLK, the rightmost cell of R_A receives the value '1' only once and applies one active clock edge at core A. Register R_B is loaded with the pattern "0100", which sets the scan frequency of core B equal to $F/4$ too. However, note that a different pattern from core A is used in order to offer non-overlapping loading of the test data from the common bus. Register R_C is initialized with pattern "1010" and thus core C receives one active clock edge every two

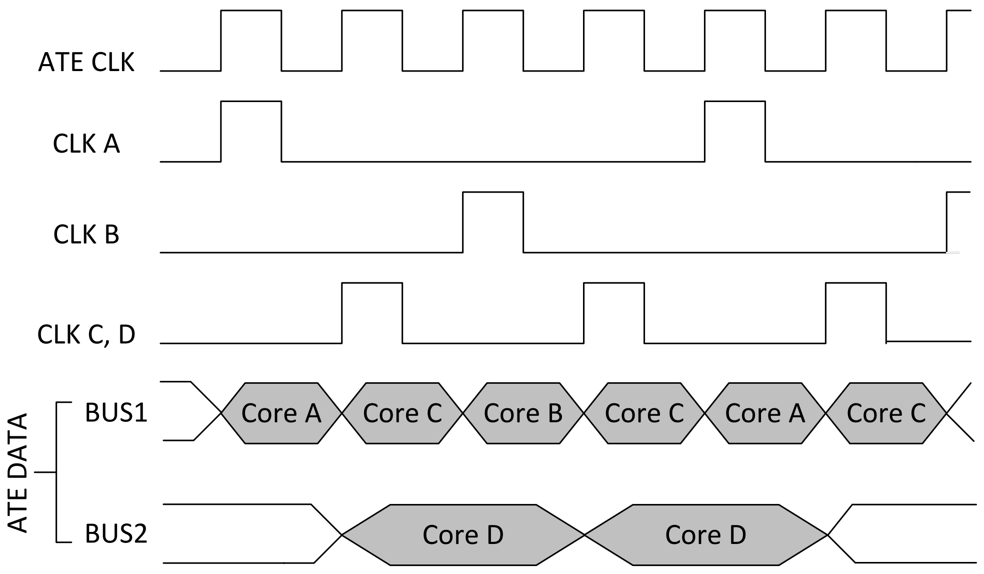


Figure 3.4: ATE_CLK, core-based clock and TAM bus waveforms for the example SoC (Fig.3.3), when TDM is used.

ATE_CLK cycles. Therefore, the scan clock frequency for core C is set to $F/2$. Similar to the previous case, the pattern loaded into R_C has non-overlapping '1' logic values with the patterns loaded into registers R_A, R_B. Finally, register R_D is initialized to the value of 1010, which corresponds to shift frequency equal to $F/2$. Note that a separate bus is used for core D, and thus the contents of R_D are independent to the contents of R_A, R_B, R_C. ■

From the above example we note that: a) all shift registers are concurrently shifted at every ATE_CLK cycle, and b) they are loaded once at the beginning of every test session with patterns that have no overlapping logic values of '1'. Therefore, at most one core at any ATE_CLK cycle receives the active edge of ATE_CLK among those that share a common bus. At the same time, at every ATE_CLK cycle, one test data vector is available at the bus, and the core which receives the active clock edge loads the test data from the bus. This is also illustrated in Fig. 3.4.

It is obvious that the tester channel utilization is increased without violating the timing specifications of the cores. Larger shift registers can be used to allow various levels of TDM (e.g., 16-bit registers offer division by 2, 4, 8 and 16). The proposed clock generation mechanism can be bypassed during normal operation and the capture cycles when on-chip PLLs are used.

It should be expected that the specific shift frequencies offered by the TDM technique will not exactly match the nominal shift frequencies that can be used at the various voltage settings. However, as shown later, the most important optimization goal is to exploit as much of the TAM frequency as possible for transferring the test data to the cores.

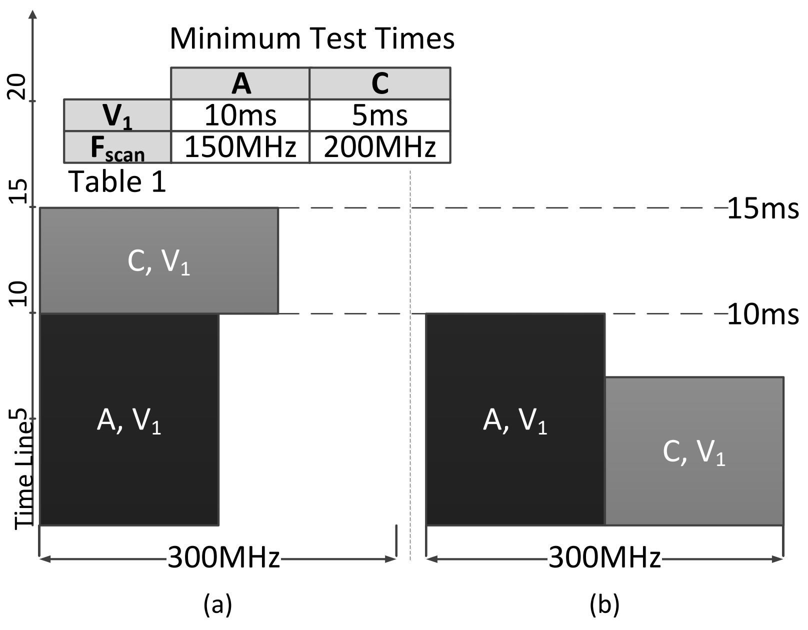


Figure 3.5: ATE_CLK, core-based clock and TAM bus waveforms for the example SoC (Fig.3.3), when TDM is used.

Example 3. Let us again consider the SoC of Fig. 3.3, tested by an ATE that supports a maximum shift frequency of 300 MHz. Let us assume that cores A, C have maximum shift frequencies at a voltage setting V_1 equal to 150 MHz and 200 MHz, respectively, and that the test time in each case is equal to 10 ms and 5 ms, as shown in table 1 of Fig. 3.5. When the maximum shift frequencies are used these two tests can be only applied serially, and the total test time is equal to 15 ms. As it is depicted in Fig. 3.5a, when core A is being tested at 150 MHz, the rest 150 MHz of the TAM frequency are left unexploited, and when core C is being tested at 200 MHz, the rest 100 MHz of the TAM frequency are left unexploited. If core C is loaded using shift frequency equal to 150 MHz instead of 200 MHz, then the test time for this core increases to 7 ms. However, in this case both tests can be applied concurrently, and the total test time drops to 10 ms. As it is illustrated in Fig. 3.5b, the TAM frequency is fully exploited for 7 ms. ■

Similar to [220], TDM can be combined with test data compression and BIST to derive even more benefits. Identical cores can be tested using a broadcast mode if identical patterns are loaded at the corresponding registers to concurrently shift test data from the bus. We ensure that active edges of the scan clock are not received by cores that are not being tested, by loading the corresponding shift register with the all-'0' pattern. For the wasted ATE_CLK cycles in which no core receives test data, the ATE repeat command can be used to avoid the storage of unnecessary test data in tester memory. Finally, testing of the logic in between the cores can be carried out by considering tests that excite multiple cores at a time.

3.3.4 Probem formulation: notation and constraints

Let us consider a multi-core SoC with N_c cores $C_1, \ldots, C_{N_c}$ and N_I voltage islands $L_1, \ldots, L_{N_I}$ ($N_c \geq N_I$). Each core belongs to exactly one of the N_I islands. We consider a set V of N_v voltage settings $V = \{V_1, \ldots, V_{N_v}\}$ for the SoC, which are sorted in descending order (i.e., $V_1 > \cdots > V_{N_v}$). Each island uses a subset of these voltage settings. For every island L_i there is an upper bound P_i on the average power that can be consumed by the cores in L_i.

We assume that Q buses $B_1, \ldots, B_Q$ comprise the TAM. Every bus is connected to one or more cores, and it spans one or more voltage islands. The maximum frequency F^{max} that is supported by the TAM resources for shifting test data into the cores is called hereafter as the *"frequency capacity*[1]*"* of the TAM. Every test that uses bus B_q consumes a fraction of the capacity F^{max} of this bus, which depends on the shift frequency used by this test. For example, any core that is tested using scan frequency $F^{max}/2, F^{max}/4, \cdots$ consumes half, a quarter, etc., of the capacity F^{max} of B_q.

Every test scheduling method that targets single-V_{dd} SoCs has to fulfill two main constraints:

C-1. Any two cores that share a TAM resource cannot be concurrently tested.

C-2. Any concurrent combination of tests applied at cores in the same island L_l, should not consume power that is greater than the specified power limit P_l of the island.

Constraints C-1 and C-2 guarantee the correctness of the power-aware test schedule generated for an SoC with a single voltage level. However, testing of multi-V_{dd} SoCs imposes additional constraints that do not exist for single-V_{dd} SoCs:

C-3. Every core C_i must be tested at multiple voltage levels V_j, which are used by the core during normal operation and comprise a subset of V.

C-4. For every core C_i and every voltage level V_j there is one maximum frequency $F_{i,j}^{max} \leq F^{max}$ that can be used for shifting test data into C_i. The shift frequency used to apply any test has to be lower or equal to $F_{i,j}^{max}$.

C-5. Any two cores that belong to the same island cannot be concurrently tested at different voltage levels.

It is obvious that constraints C-3, C-4 and C-5 increase the difficulty of test scheduling process and also increase test application time, especially when TAM resources are connected to cores that reside in different voltage islands. However, TDM offers one very important advantage: it relaxes constraint C-1 that prevents any two cores that share the same TAM resources to be concurrently tested. Specifically, when TDM is used, constraint C-1 is modified as follows:

[1]Throughout this work we use the terms frequency and bandwidth interchangeably to denote the volume of test data transferred over the bus.

C-1^T. Any number of cores that share the same TAM resources can be concurrently tested, provided that the aggregate shift frequencies used by the corresponding tests do not exceed the frequency capacity of the TAM resource.

By applying C-1^T instead of C-1, every shared TAM resource can be concurrently used to transfer test data of multiple cores, and the total test time is considerably reduced.

Given the above framework, our objective is to generate optimal test schedules in terms of the time required for testing multi-V_{dd} SoCs. Hereafter, we refer to the test of core C_i at power supply voltage level V_j using shift frequency F_k as test or task $\tau_{C_i V_j F_k}$. We assume that the maximum frequency F^{max} used for shifting test data is the same for all TAM resources. Depending on the specific TDM architecture used, a pre-specified set of N_f frequencies $F_1 > F_2 > \cdots > F_{N_f}$ is supported for shifting test data into a core ($F^{max} = F_1$). Each of these frequencies is equal to a fraction of F^{max}. Test data can be shifted into core C_i at voltage V_j using any shift frequency that is lower or equal to the maximum shift frequency $F_{i,j}^{max}$. Therefore, only the subset of frequencies $F_M > F_{M+1} > \cdots > F_{N_f}$ can be used, where $F_M \leq F_{i,j}^{max}$.

The test time that is needed for applying test $\tau_{C_i V_j F_{i,j}^{max}}$ is equal to $\tau_{C_i V_j F_{i,j}^{max}}$ and it depends on the number of test vectors applied to the core, the shift frequency $F_{i,j}^{max}$, and the core wrapper depth. In the case of industrial circuits the wrapper depth tends to be very large due to the long internal scan chains of the cores; thus the total number of capture cycles constitute a negligible portion of the test time, and they can be neglected. When a smaller shift frequency $F_k < F_{i,j}^{max}$ is used, the test time increases proportionally, as shown by the following equation:

$$\tau_{C_i V_j F_k} = \tau_{C_i V_j F_{i,j}^{max}} \cdot \frac{F_{i,j}^{max}}{F_k} \tag{3.1}$$

Let $P_{C_i V_j F_{i,j}^{max}}$ be the average dynamic power consumed by test $\tau_{C_i V_j F_{i,j}^{max}}$. Then, $P_{C_i V_j F_{i,j}^{max}}$ can be approximated by the following equation [9]:

$$P_{C_i V_j F_{i,j}^{max}} = a(C_i) \cdot C_{load}(C_i) \cdot V_j^2 \cdot F_{i,j}^{max} \tag{3.2}$$

where $a(C_i)$ is the average switching activity at the internal circuit nodes of C_i caused by the application of test $\tau_{C_i V_j F_{i,j}^{max}}$, and $C_{load}(C_i)$ is the total load capacitance that is charged/discharged during the shift operation. When a smaller frequency $F_k < F_{i,j}^{max}$ is used for shifting the test data into the core, the average dynamic power consumed decreases proportionally as shown by the following equation:

$$P_{C_i V_j F_k} = P_{C_i V_j F_{i,j}^{max}} \cdot \frac{F_k}{F_{i,j}^{max}} \tag{3.3}$$

3.3.5 ILP-based test scheduling approach

For every triplet (C_i, V_j, F_k) a binary variable $S_{C_i V_j F_k}$ is assigned, which is equal to '1' whenever the test of C_i at V_j is applied using shift frequency F_k. First, we ensure that exactly one shift frequency is selected for testing C_i at voltage V_j:

$$\sum_{k=1}^{N_f} S_{C_iV_jF_k} = 1,\ \forall i \in \{1 \dots N_c\},\ \forall j \in \{1 \dots N_v\} \tag{3.4}$$

For those voltage settings at which a core is not tested, the above constraint is omitted, and for those values of k that $F_k > F_{i,j}^{max}$, we set $S_{C_iV_jF_k} = 0$. Recall that higher frequencies are not supported at all voltage settings due to the likelihood of timing violations during scan shifting. Let $ST_{C_iV_jF_k}$ denote the start time of test $\tau_{C_iV_jF_k}$. Since only a single frequency can be used for applying the test on C_i at voltage level V_j, the start time $ST_{C_iV_j}$ and the end time $END_{C_iV_j}$ for this test are given by the following relations

$$ST_{C_iV_j} = \sum_{k=1}^{N_f} S_{C_iV_jF_k} \cdot ST_{C_iV_jF_k} \tag{3.5}$$

$$END_{C_iV_j} = \sum_{k=1}^{N_f} S_{C_iV_jF_k} \cdot (ST_{C_iV_jF_k} + \tau_{C_iV_jF_k}) \tag{3.6}$$

The product $S_{C_iV_jF_k} \cdot ST_{C_iV_jF_k}$ is not linear so it is replaced by the variable $y_{C_iV_jF_k}$ and new constraints are introduced. Let TB (TimeBound) be the maximum possible test time calculated as the summation of the time needed for every core and voltage setting when it is loaded using the minimum scan frequency F_{N_f}, that is

$$TB = \sum_{i\in[1\cdots N_c], j\in[1\cdots N_v]} T_{C_iV_jF_{N_F}}$$

i.e., we assume that the longest task is used for every core and no parallelism is allowed. Then we have the following three relations for each variable $y_{C_iV_jF_k}$:

$$\begin{aligned} y_{C_iV_jF_k} - TB \cdot S_{C_iV_jF_k} &\leq 0 \\ -ST_{C_iV_jF_k} + y_{C_iV_jF_k} &\leq 0 \\ ST_{C_iV_jF_k} - y_{C_iV_jF_k} + TB \cdot S_{C_iV_jF_k} &\leq TB \end{aligned}$$

Thus (3.5), (3.6) become

$$ST_{C_iV_j} = \sum_{k=1}^{N_f} y_{C_iV_jF_k} \tag{3.7}$$

$$END_{C_iV_j} = \sum_{k=1}^{N_f} (y_{C_iV_jF_k} + S_{C_iV_jF_k} \cdot \tau_{C_iV_jF_k}) \tag{3.8}$$

According to constraint C-5, any two cores C_{i_1}, C_{i_2} in the same island cannot be concurrently tested at different voltage settings V_{j_1}, V_{j_2} ($V_{j_1} \neq V_{j_2}$). Therefore, either test for C_{i_1}, V_{j_1} begins after the test for C_{i_2}, V_{j_2} finishes or vice versa. Using relations (3.7), (3.8) it is written as: "$\mathcal{C}_1$ or $\mathcal{C}_2$", where

$$\mathcal{C}_1 : \sum_{k=1}^{N_f} y_{C_{i_1}V_{j_1}F_k} \geq \sum_{k=1}^{N_f} (y_{C_{i_2}V_{j_2}F_k} + S_{C_{i_2}V_{j_2}F_k} \cdot \tau_{C_{i_2}V_{j_2}F_k})$$

$$\mathcal{C}_2 : \sum_{k=1}^{N_f} y_{C_{i_2}V_{j_2}F_k} \geq \sum_{k=1}^{N_f} (y_{C_{i_1}V_{j_1}F_k} + S_{C_{i_1}V_{j_1}F_k} \cdot \tau_{C_{i_1}V_{j_1}F_k}) \tag{3.9}$$

To show the linearization of (3.9), first, we introduce two binary variables $\theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}$ and $\theta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}$ which satisfy the equation $\theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} + \theta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} = 1$. Then constraint $\mathcal{C}_1$ or $\mathcal{C}_2$ can be written as

$$\theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot \left(\sum_{k=1}^{N_f} y_{C_{i_1}V_{j_1}F_k} - \sum_{k=1}^{N_f} (y_{C_{i_2}V_{j_2}F_k} + S_{C_{i_2}V_{j_2}F_k} \cdot T_{C_{i_2}V_{j_2}F_k}) \right) \geq 0$$

$$\theta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot \left(\sum_{k=1}^{N_f} y_{C_{i_2}V_{j_2}F_k} - \sum_{k_1=1}^{N_f} (y_{C_{i_1}V_{j_1}F_k} + S_{C_{i_1}V_{j_1}F_k} \cdot T_{C_{i_1}V_{j_1}F_k}) \right) \geq 0$$

Each of the terms

$$\begin{gathered}
\theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot y_{C_{i_1}V_{j_1}F_k} \\
\theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot y_{C_{i_2}V_{j_2}F_k} \\
\theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot S_{C_{i_2}V_{j_2}F_k} \\
\theta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot y_{C_{i_2}V_{j_2}F_k} \\
\theta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot y_{C_{i_1}V_{j_1}F_k} \\
\theta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot S_{C_{i_1}V_{j_1}F_k}
\end{gathered}$$

needs further linearization. To this end we introduce 6 variables $L^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}$, $L^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}$, $L^3_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}$, $L^4_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}$, $L^5_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}$, $L^6_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}$ which are set equal to the above terms respectively. Note that L_1, L_2, L_4 and L_5 are integer variables while L_3, L_6 are binary variables. We will show the linearization for L_1, L_3 and the rest of the variables are linearized in a similar manner. For $L^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}$ we have

$$\begin{aligned}
L^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k} - TB \cdot \theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} &\leq 0 \\
-y_{C_{i_1}V_{j_1}F_k} + L^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k} &\leq 0 \\
y_{C_{i_1}V_{j_1}F_k} - L^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k} + TB \cdot \theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} &\leq TB
\end{aligned}$$

For $L^3_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}$ we have

$$\begin{aligned}
\theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} + S_{C_{i_2}V_{j_2}C_{i_2}V_{j_2}} &\leq & L^3_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k} + 1 \\
\theta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} + S_{C_{i_2}V_{j_2}C_{i_2}V_{j_2}} &\geq & 2 \cdot L^3_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}
\end{aligned}$$

Then, constraint $\mathcal{C}_1$ or $\mathcal{C}_2$ becomes linear as follows:

$$\begin{aligned}
&\sum_{k=1}^{N_f}(L^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k} + L^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k} + L^3_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k} + \\
&\quad L^4_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k} + L^5_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k} + L^6_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}k}) \geq 0
\end{aligned}$$

In a similar way, we determine the concurrency between different tests: if the test for C_{i_1} at voltage V_{j_1} begins after the test for C_{i_2} at voltage V_{j_2} finishes or vice versa, then the two tests are not concurrent. Concurrency is determined for all cores sharing a common bus, excluding those that are in the same island and correspond to $V_{j_1} \neq V_{j_2}$ as they have been excluded by the previous constraint. Let $Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}$ be a binary variable which is equal to '1' if the tests for core C_{i_1} at V_{j_1} and core C_{i_2} at V_{j_2} overlap. Then we formally write:

$$\begin{aligned}
&\text{If } ST_{C_{i_1}V_{j_1}} \geq END_{C_{i_2}V_{j_2}} \text{ or } ST_{C_{i_2}V_{j_2}} \geq END_{C_{i_1}V_{j_1}} \\
&\text{then } Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} = 0 \text{ else } Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} = 1
\end{aligned} \tag{3.10}$$

To show the linearization of (3.10), two binary variables $\delta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}, \delta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}$ are introduced. Then the formula can be replaced by the following relations:

$$\begin{aligned}
ST_{C_{i_1}V_{j_1}} &\geq END_{C_{i_2}V_{j_2}} - TB \cdot (1 - \delta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}) \\
ST_{C_{i_1}V_{j_1}} &\leq END_{C_{i_2}V_{j_2}} + TB \cdot \delta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \\
ST_{C_{i_2}V_{j_2}} &\geq END_{C_{i_1}V_{j_1}} - TB \cdot (1 - \delta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}) \\
ST_{C_{i_2}V_{j_2}} &\leq END_{C_{i_1}V_{j_1}} + TB \cdot \delta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \\
1 - Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} &\geq \delta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \\
1 - Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} &\geq \delta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \\
1 - Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} - \delta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} - \delta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} &\leq 0
\end{aligned}$$

By replacing $ST_{C_{i_1}V_{j_1}}$, $END_{C_{i_1}V_{j_1}}$, $ST_{C_{i_2}V_{j_2}}$ and $END_{C_{i_2}V_{j_2}}$ in the first four relations with their equivalent notation we get the following four relations which are all linear

$$\sum_{k=1}^{N_f} y_{C_{i_1}V_{j_1}F_k} \geq \sum_{k=1}^{N_f} (y_{C_{i_2}V_{j_2}F_k} + S_{C_{i_2}V_{j_2}F_k} T_{C_{i_2}V_{j_2}F_k}) - \\ -TB \cdot (1 - \delta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}})$$

$$\sum_{k=1}^{N_f} y_{C_{i_1}V_{j_1}F_k} \leq \sum_{k=1}^{N_f} (y_{C_{i_2}V_{j_2}F_k} + S_{C_{i_2}V_{j_2}F_k} T_{C_{i_2}V_{j_2}F_k}) + \\ +TB \cdot \delta^1_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}$$

$$\sum_{k=1}^{N_f} y_{C_{i_2}V_{j_2}F_k} \geq \sum_{k=1}^{N_f} (y_{C_{i_1}V_{j_1}F_k} + S_{C_{i_1}V_{j_1}F_k} T_{C_{i_1}V_{j_1}F_k}) - \\ -TB \cdot (1 - \delta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}})$$

$$\sum_{k=1}^{N_f} y_{C_{i_2}V_{j_2}F_k} \leq \sum_{k=1}^{N_f} (y_{C_{i_1}V_{j_1}F_k} + S_{C_{i_1}V_{j_1}F_k} T_{C_{i_1}V_{j_1}F_k}) + \\ +TB \cdot \delta^2_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}$$

Constraint C-1^T bounds the number and type of tests that concurrently use the same TAM resource. Every supported shift frequency consumes a fraction of the capacity (in time) of the TAM resource. Any test may use a fraction of this capacity, but any concurrent combination of tests must not exceed the total capacity of the TAM resource. Let $W(F_k)$ be the capacity consumed by any test using shift frequency F_k. Let also $W(TAM)$ be the capacity available on the TAM resource. Then, for any two cores C_{i_1}, C_{i_2} connected to the same TAM resource and tested concurrently at supply voltages V_{j_1}, V_{j_2} the sum of their weights must not exceed $W(TAM)$. This is modelled by the following relation:

$$Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot \sum_{k=1}^{N_f} (S_{C_{i_1}V_{j_1}F_k} + S_{C_{i_2}V_{j_2}F_k}) \cdot W(F_k) \\ \leq W(TAM) \tag{3.11}$$

Again (3.11) is linearized as follows. The binary variables $b^{1-1}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2})}, b^{1-2}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}$ are introduced. The product term $Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot S_{C_{i_1}V_{j_1}F_k}$ is linearized as follows:

$$\begin{aligned} Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} + S_{C_{i_1}V_{j_1}F_k} \leq & \quad b^{1-1}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} + 1 \\ Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} + S_{C_{i_1}V_{j_1}F_k} \geq & \quad 2 \cdot b^{1-1}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \end{aligned}$$

In a similar way the term $Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \cdot S_{C_{i_2}V_{j_2}F_k}$ is linearized as follows:

$$\begin{aligned} Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} + S_{C_{i_2}V_{j_2}F_k} \leq & \quad b^{1-2}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} + 1 \\ Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} + S_{C_{i_2}V_{j_2}F_k} \geq & \quad 2 \cdot b^{1-2}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}} \end{aligned}$$

Consequently we get:

$$\sum_{k=1}^{N_f}\Big((b^{1-1}_{C_{i_1}V_{j_1}F_k}+b^{1-2}_{C_{i_2}V_{j_2}F_k})\cdot W(F_k)\Big)\le W(TAM)$$

In the same way we construct the constraints for triplets or larger sets of tests depending on the number of cores which are connected to any TAM resource. Let us see the case of three tests running concurrently. The constraint is re-written as follows: for any three cores $C_{i_1}, C_{i_2}, C_{i_3}$ which are tested concurrently at supply voltages $V_{j_1}, V_{j_2}, V_{j_3}$ and are connected to the same TAM resource the sum of their weights should not exceed the capacity of the TAM resource. Three different tests are executed concurrently (at least a common part of all of them) when every possible pair of them has a part which executes in parallel, i.e. when $Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}\cdot Conc_{C_{i_1}V_{j_1}C_{i_3}V_{j_3}}\cdot Conc_{C_{i_2}V_{j_2}C_{i_3}V_{j_3}}=1$. Then the constraint is written as follows:

$$\begin{aligned}&Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}\cdot Conc_{C_{i_1}V_{j_1}C_{i_3}V_{j_3}}\cdot Conc_{C_{i_2}V_{j_2}C_{i_3}V_{j_3}}\cdot\\ &\left(\sum_{k=1}^{N_f}(S_{C_{i_1}V_{j_1}F_k}+S_{C_{i_2}V_{j_2}F_k}+S_{C_{i_3}V_{j_3}F_k})W(F_k)\right)\le\\ &\qquad\le W(TAM)\end{aligned}$$

At first we need to linearize the product term $Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}\cdot Conc_{C_{i_1}V_{j_1}C_{i_3}V_{j_3}}\cdot Conc_{C_{i_2}V_{j_2}C_{i_3}V_{j_3}}$. To this end we introduce one binary variable $p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}}$ and we set the following relations:

$$\begin{aligned}Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}+Conc_{C_{i_1}V_{j_1}C_{i_3}V_{j_3}}+&\\ +Conc_{C_{i_2}V_{j_2}C_{i_3}V_{j_3}}\le p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}}+2&\\ Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}+Conc_{C_{i_1}V_{j_1}C_{i_3}V_{j_3}}+&\\ +2Conc_{C_{i_2}V_{j_2}C_{i_3}V_{j_3}}\ge 4\cdot p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}}&\end{aligned}$$

Using the above relations we have:

$$\begin{aligned}&\sum_{k=1}^{N_f}\big((S_{C_{i_1}V_{j_1}F_k}+S_{C_{i_2}V_{j_2}F_k}+S_{C_{i_3}V_{j_3}F_k})\cdot W(F_k)\big)\cdot\\ &\qquad\cdot p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}}\le W(TAM)\end{aligned}$$

Terms $p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}}, S_{C_{i_1}V_{j_1}F_k}, p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}}S_{C_{i_2}V_{j_2}F_k}, p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}}S_{C_{i_3}V_{j_3}F_k}$ need a final step of linearization. Therefore we introduce binary variables $b^{2-1}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k}$, $b^{2-2}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k}$, $b^{2-3}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k}$ for each scan frequency F_k and we have the following three pairs of relations

$$\begin{aligned}
p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}} + S_{C_{i_1}V_{j_1}F_k} &\leq & b^{2-1}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k} + 1 \\
p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}} + S_{C_{i_1}V_{j_1}F_k} &\geq & 2b^{2-1}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k} \\
p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}} + S_{C_{i_2}V_{j_2}F_k} &\leq & b^{2-2}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k} + 1 \\
p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}} + S_{C_{i_2}V_{j_2}F_k} &\geq & 2b^{2-2}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k} \\
p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}} + S_{C_{i_3}V_{j_3}F_k} &\leq & b^{2-3}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k} + 1 \\
p_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}} + S_{C_{i_3}V_{j_3}F_k} &\geq & 2b^{2-3}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k}
\end{aligned}$$

Finally we have:

$$\begin{aligned}
&\sum_{k=1}^{N_f} \Big((b^{2-1}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k} + b^{2-2}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k} + \\
&\quad + b^{2-3}_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}C_{i_3}V_{j_3}k}) \cdot W(F_k) \Big) \leq W(TAM)
\end{aligned}$$

Finally, we incorporate constraint C-2: for any two cores C_{i_1}, C_{i_2} that belong to the same island L_l and tested concurrently at voltages V_j, the sum of their average power consumption must not exceed the power limit P_l of the island. Variables $Conc_{C_{i_1}V_{j_1}C_{i_2}V_{j_2}}$ are extended in this step to include also cores that do not share a common bus, but belong to the same island. This is modelled as follows:

$$\begin{aligned}
Conc_{C_{i_1}V_jC_{i_2}V_j} \cdot &\sum_{k=1}^{N_f} (S_{C_{i_1}V_jF_k} \cdot P_{C_{i_1}V_jF_k} + \\
&S_{C_{i_2}V_jF_k} \cdot P_{C_{i_2}V_jF_k}) \leq P_l \qquad (3.12)
\end{aligned}$$

Triplets or larger sets of tests are constructed depending on the number of cores of each island. The linearization of (3.12) is similar to the linearization of (3.11).

Finally, if TST is the total test time of the SoC, the optimization objective of the ILP model is the following:

Minimize: TST
Subject to: $TST \geq END_{C_i,V_j}\ \forall i \in [1, N_c],\ j \in [1, N_v]$.

The complexity of the ILP formulation depends on constraint C-1^T, which ensures the highest number of relations. This number depends: (a) on the number of cores connected to each bus; (b) on the number of different islands that these cores belong to. Let "a" be the highest number of cores connected to any bus. In the worst case, each of the a cores belongs in a different island, and each core has to be tested at all N_v voltage levels. Then the complexity of constraint C-1^T is $O((N_v)^a + \binom{a}{a-1} \cdot (N_v)^{a-1} + \binom{a}{a-2} \cdot (N_v)^{a-2} + \dots) =$

$O((N_v)^a)$. Since C-1^T is the most complex constraint, we conclude that in the worst case the complexity of the ILP model is $O((N_v)^a)$. In the extreme and unrealistic case that all cores are connected to a single bus and also reside at different islands, the complexity is equal to $O((N_v)^{N_c})$. However, we have to note that in this case, a test scheduling method is not really needed as all tests can be applied sequentially using the maximum bandwidth of the channel.

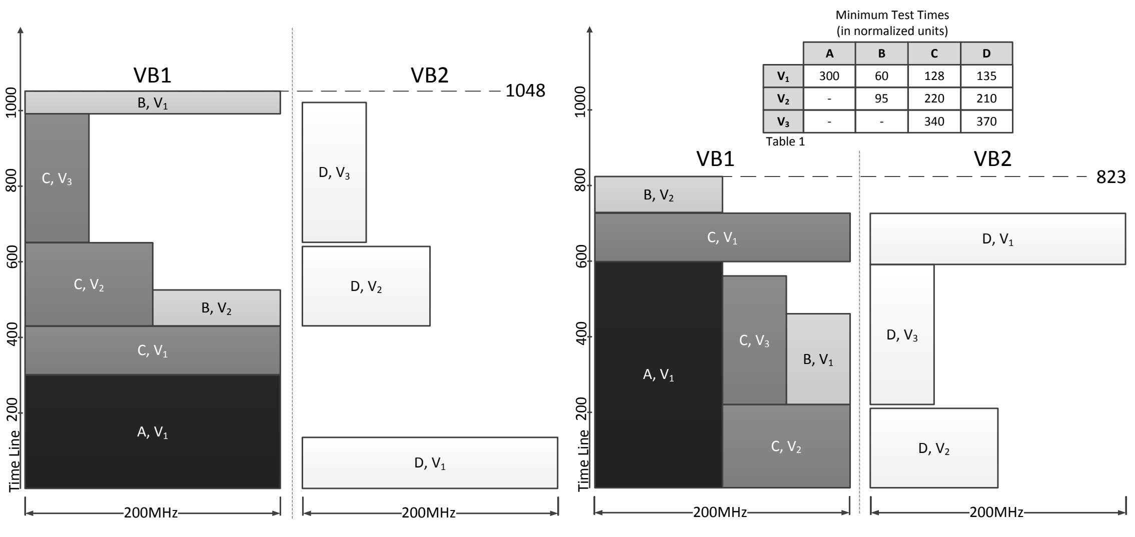
3.3.6 Rectangle packing and simulated annealing test scheduling approach

Even though ILP models can optimally solve NP-hard test-scheduling optimization problems [80], they do not scale well for large SoC designs. This limitation can be overcome using a heuristic based on the Rectangle Packing (RP) approach. Specifically, each candidate test $\tau_{C_iV_jF_k}$ is modelled as a rectangle $R_{C_iV_jF_k}$, with width equal to F_k and height equal to $\tau_{C_iV_jF_k}$. Every bus is assigned a virtual bin with unlimited height and width equal to F^{max} (F^{max} is the maximum frequency supported by the ATE channel). The objective of the RP approach is to pack a predetermined set of rectangles into Q virtual bins so that the occupied height in each bin is minimum. Recall that Q is the number of the available TAM buses, while the overall occupied height in a virtual bin represents the test time per TAM bus. Therefore, the maximum of the derived test times per TAM bus is the TST.

By setting the width of each bin to F^{max}, constraint C-1^T is satisfied, as rectangles with aggregate frequency higher than the capacity of the bus cannot be placed in parallel in the bin. Until the last part of Section 3.3.4 we temporarily ignore the power constraint C-2. In order to fulfill constraints C-3 and C-4, it suffices to ensure that all the required tests are applied to the cores using shift frequencies that do not violate the maximum shift frequency at each voltage level. This set of tests corresponds to a set of rectangles with specific dimensions, which are given as inputs to the RP algorithm. The packing of the selected rectangles is based on the Bottom-Left (BL) rule [221], which keeps the overall height in a bin, and thus the TST, as low as possible. In BL, each rectangle is placed at a position that does not violate constraint C-5 and it minimizes the y-coordinate of the top side of the rectangle. If there are several such valid positions, then we select the leftmost position on the x-axis.

During a packing step, we go through each unpacked rectangle and each possible placement of that rectangle in the virtual bin. Every rectangle-position pair is assigned a score, which is related to the position of the rectangle in the bin. Specifically, the lower the y-coordinate of the top side of the rectangle - position pair, the higher the score. Then, we select the pair with the highest score, we place the selected rectangle to the selected position, and we move to the next packing step. The procedure is completed when all tests have been scheduled. Let us consider the following example.

Example 4. Let us again consider the SoC of Fig. 3.3, with voltage levels V_1, V_2, V_3,



Minimum Test Times
(in normalized units)

	A	B	C	D
V_1	300	60	128	135
V_2	-	95	220	210
V_3	-	-	340	370

Table 1

(a) Example test schedule

(b) Test schedule derived by the proposed method

Figure 3.6: Rectangle Packing Example.

and nominal scan frequencies at each one of them $F_1 = 200$ MHz, $F_2 = 100$ MHz and $F_3 = 50$ MHz, respectively. Let the tester provide the ATE_CLK signal with frequency 200 MHz. The embedded table in Fig. 3.6(b) shows the test times per core at V_1, V_2, V_3, when the maximum rated shift frequency is used at each voltage level (dashes indicate that the corresponding tests are omitted). Let us consider the following set of tests, which use the maximum rated frequency per voltage level: $s_a = \{\tau_{AV_1F_1}, \tau_{BV_1F_1}, \tau_{BV_2F_2}, \tau_{CV_1F_1}, \tau_{CV_2F_2}, \tau_{CV_3F_3}, \tau_{DV_1F_1}, \tau_{DV_2F_2}, \tau_{DV_3F_3}\}$. This set constitutes a complete test for the SoC at hand. Each one of these tests is modelled as a rectangle. For example, the candidate test $\tau_{CV_3F_3}$ is modelled as a rectangle with width equal to $F_3 = 50$ MHz and height equal to $\tau_{CV_3F_3} = 340$ time units. Bin VB_1 corresponds to bus B_1, and bin VB_2 corresponds to bus B_2. Fig. 3.6(a) shows the test schedule produced by RP for the example SoC. The x-axis of each bin presents the maximum shift frequency, which is 200 MHz for this example, and the y-axis presents the test time, which is equal to 1048 for this particular case. ■

In the example presented above, we assumed that every test is applied using the maximum rated shift frequency. Even though this is an intuitive decision to minimize the test time, it is obvious from Fig. 3.6(a) that the available TAM bandwidth is underutilized, due to the large blank spaces left by the RP approach. TDM can overcome this problem by exploring also smaller frequencies. If some tests are applied at lower than the rated shift frequencies, then the shapes of the corresponding rectangles change, and they may better fit in the available area.

Example 5. Let us consider the following set of tests for the same SoC: $s_b = \{\tau_{AV_1F_2}, \tau_{BV_1F_3}, \tau_{BV_2F_2}, \tau_{CV_1F_1}, \tau_{CV_2F_2}, \tau_{CV_3F_3}, \tau_{DV_1F_1}, \tau_{DV_2F_2}, \tau_{DV_3F_3}\}$. Fig. 3.6(b) shows the test schedule provided by RP for set s_b. In this case, the test time has dropped by 21.5% without any compromise on test quality. Even though the time for testing cores A, B at voltage V_1 is increased 2 and 4 times respectively, the use of smaller frequencies permits the better packing of the rectangles in the area of the virtual bin. We note that the blank space was reduced 7 times as compared to Fig. 3.6(a). ■

The differentiation between s_a and s_b lead us to the conclusion that the best frequency for applying each test must be selected in order to minimize TST. One approach for effective exploration of the space of available frequencies is Simulated Annealing (SA). SA is a generic probabilistic metaheuristic that is able to locate a good approximation to the global optimum of a given function in a large search space. It succeeds by slowly decreasing the probability of accepting worse solutions as it explores the solution space [224, 223].

Initially, we define the search space for SA. This space is the complete set of tests that can be applied in multi-V_{dd} testing of an SoC, when TDM is available. In particular, a set of candidate tests $\tau_{C_iV_jF_k}$ is formed per core - voltage pair, (C_i, V_j) by varying shift frequency F_k. Let us again consider the same SoC as in the previous examples. Then, $S\tau_{CV_1}$, namely the set of candidate tests for testing core C at voltage setting V_1, is defined as $S\tau_{CV_1} = \{\tau_{CV_1F_1},\ \tau_{CV_1F_2},\ \tau_{CV_1F_3}\}$. Similarly, we construct every available set of core - voltage pairs.

In order to fulfil Constraints C-3 and C-4, one test $\tau_{C_iV_jF_k}$ should be selected from each set $S\tau_{C_i,V_j}$ to form a complete set of tests to be scheduled later by RP. This set of tests is referred to hereafter as the state of the SA. If $Size(S\tau_{C_i,V_j})$ represents the number of candidate tests in a set $S\tau_{C_i,V_j}$, then $NS = \prod_{i=1}^{N_c}\prod_{j=1}^{N_v} Size(S\tau_{C_i,V_j})$ different states can be created. Note that the number NS can be very large. Even for the very small SoC of the previous example, there are 648 possible set of tests or *states* and two of them are the sets S_a and S_b reported earlier. Despite the large number of states, SA is able to explore the search space efficiently using metaheuristics.

When SA begins, an initial *state* s_{init} is generated, which includes from each set $S\tau_{C_iV_j}$ the test with the maximum rated scan frequency. In this case, the length of each test is minimum, so the derived state s_{init}, is a good candidate to offer an adequate initial TST that will enable the SA algorithm to move faster to the global optimum. The SA algorithm proceeds with the generation of a neighbouring state s_1 for state s_{init}, then a neighbouring state s_2 for state s_1 etc.

To create a neighbouring state s_{p+1} for state s_p, we consider that r randomly selected candidate tests from state s_p are substituted from new, differentiated candidate tests taken from the correspondent sets $S\tau_{C_iV_j}$. Then, the RP algorithm will undertake the task to exploit any possible correlation among the tests in state s_{p+1}. Even though it is intuitive to select tests that run at high shift frequencies, TDM has shown that this is not always justified for minimizing the overall test time. Therefore, we consider that each new candidate test $\tau_{C_iV_jF_k}$ is selected randomly from the set $S\tau_{C_iV_j}$, without excluding the case of reusing the previously examined test configuration.

The SA algorithm has an inherent dependence on a probability function, which determines if a state s will be accepted. It is called *"acceptance probability function"*,

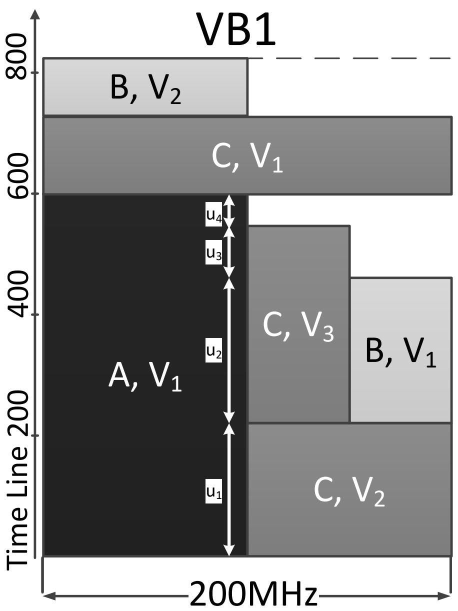


Figure 3.7: Power aware test scheduling using the SA-RP algorithm.

$P(E(s_p), E(s_{p+1}), T)$, and it depends on three variables: a) the energy function $E(s_p)$ of the previous state s_p, b) the energy function of the next state $E(s_{p+1})$, and, c) the temperature T. The energy function for state s_p is simply the test time TST of the set of the tests that comprise state s_p. The temperature T is a variable that determines the likelihood of accepting states that are inferior to the previous states. The acceptance probability function $P(E(s_p), E(s_{p+1}), T)$ of the SA algorithm is derived from the Maxwell-Boltzmann distribution [225], and it is defined as follows

$$P(E(s_p), E(s_{p+1}), T) = \begin{cases} 1 \text{ if } E(s_{p+1}) < E(s_p) \\ e^{-(E(s_{p+1}) - E(s_p))/T} \text{ otherwise} \end{cases} \tag{3.13}$$

The temperature T is calculated through the expression $T = T_{Init} \cdot CR$. T_{Init} is the initial temperature, while CR is the cooling rate ($CR < 1$) that we apply in order to reach a final temperature T_{Final}, where $T_{Init} > T_{Final}$. As shown by equation (3.13), the probability with which SA accepts a worse solution decreases a) with time (cooling process), and b) with the 'distance' between the new (worse) solution and the old one. A new solution is always accepted if it is better than the previous one. When T reaches the value of T_{Final}, the SA algorithm ends and the final state is considered as the one that enables the RP algorithm to provide the lowest possible TST. Note that T_{Init} and CR, determine the maximum number of trial states examined, and thus the running time of the algorithm. In this work, we have set $T_{Init} = 400$, $CR = 0.999$ and $T_{Final} = 0.001$ for our experiments.

In order to ensure that the average power consumption remains under certain bounds, we invoke a checking process before the placement of a rectangle in its bin. Let $ST_{C_iV_j}$ be

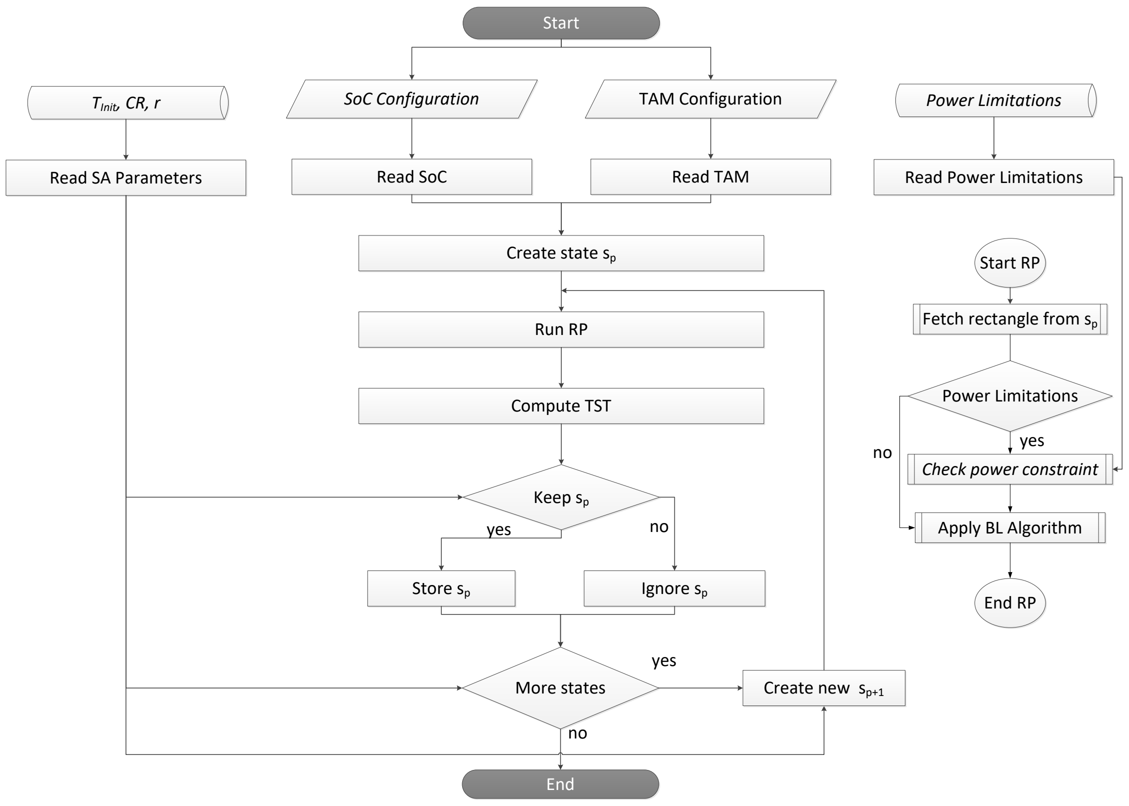


Figure 3.8: Flowchart of test scheduling using the SA-RP method.

the potential start time of test $\tau_{C_iV_jF_k}$. In order to confirm that the period $[ST_{C_iV_j}, ST_{C_iV_j} + \tau_{C_iV_jF_k}]$ does not violate the power consumption limit of the island, we break it into small intervals $u_1, u_2, \ldots u_q$, where $u_1 \cup u_2 \cup \cdots \cup u_q = [ST_{C_iV_j}, ST_{C_iV_j} + \tau_{C_iV_jF_k}]$. To identify every interval $u_r, r = 1, 2, \ldots, q$, we search for scheduled tests that run on island L_i and they are starting or ending in the examined period. For each test that either starts or ends in this period, the start and/or end time of this test defines a point in time that a new interval begins. Each interval has a specific average power consumption, which is equal to the aggregate average power consumption of all tests applied at cores of the same island during this interval. The power consumption of test $\tau_{C_iV_jF_k}$ is added to the power consumption of each interval u_r, and if the power consumption limit of the island is violated, the period $[ST_{C_iV_j}, ST_{C_iV_j} + \tau_{C_iV_jF_k}]$ is rejected for test $\tau_{C_iV_jF_k}$.

Example 6. Let us consider the test schedule in Fig. 3.6(b) of SoC of Fig. 3.3. We assume that tests for core - voltage pairs, $(C, V_1), (C, V_3), (B, V_1)$ have been already scheduled and the next candidate is the core-voltage pair (A, V_1). The later should be scheduled so that the power consumption in island L_1 does not exceed an upper bound P_1. To make a decision about the ability to schedule the test under scope in the period shown by the black rectangle in Fig. 3.7, the power consumptions $P_{u_1}, P_{u_2}, P_{u_3}, P_{u_4}$ in the time intervals u_1, u_2, u_3, u_4 are calculated. if any of the above calculated power consumptions exceeds the upper bound P_1, the proposed period for the test of the core-voltage pair (A, V_1)

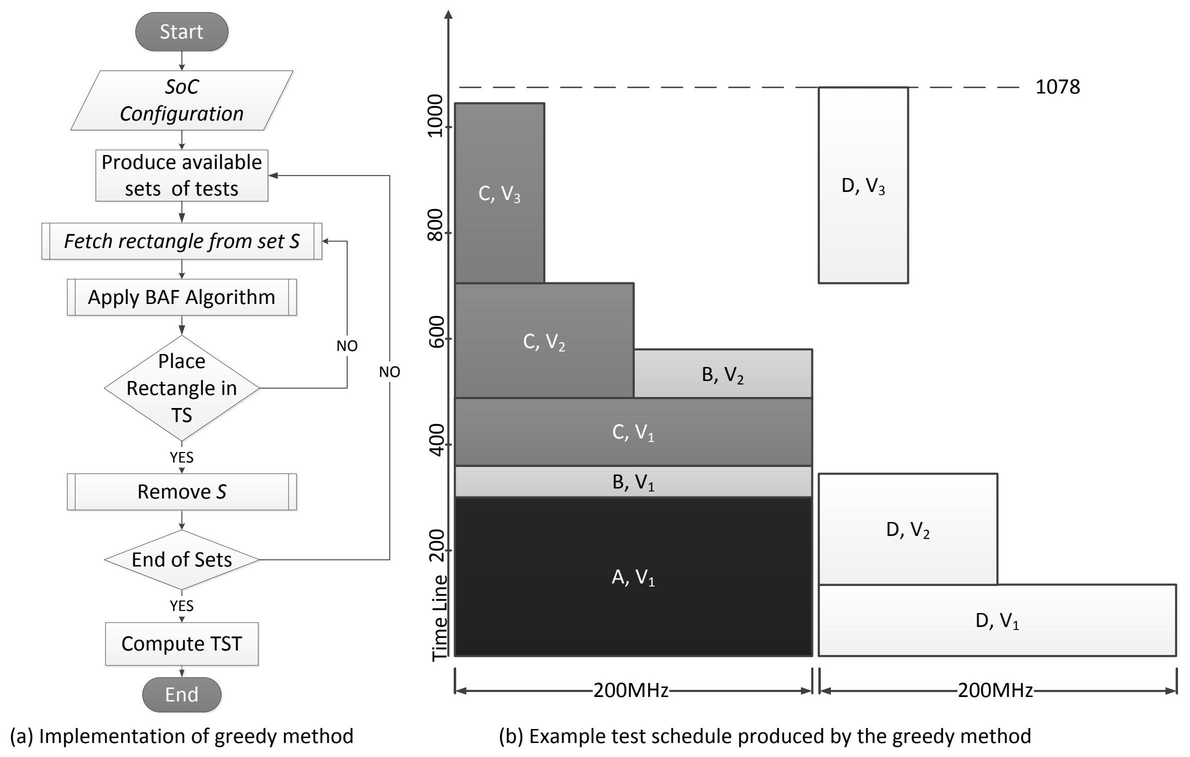


Figure 3.9: Flowchart of test scheduling using the greedy method.

is rejected. Otherwise, it is further evaluated according to the criteria of the SA-RP algorithm. ■

A flow diagram of the SA-RP method is given in Fig. 3.8. The worst-case time complexity of the SA-RP method is $O(F \cdot N_c^3 \cdot N_v^3)$, where F is a function of T_{Init}, and CR. Considering that N_v is usually a small constant number (it is equal to 4 and 6 in our case study), the complexity can be set equal to $O(F \cdot N_c^3)$. Parameter F determines the number of states examined by the algorithm. Therefore, a high (low) number of states examined increases (decreases) the value of F and the run time of the algorithm. It is obvious that the complexity of the SA-RP method is much smaller than the complexity of the ILP method. Therefore, it scales better than ILP. As we show in section 4.3, it also gives very good results, which are comparable to those offered by the ILP method.

3.3.7 Test scheduling based on greedy heuristic

The computational cost of the test scheduling method based on Rectangle Packing and Simulated Annealing, in terms of time and resources, is acceptable for medium to very large SoCs. However, there are cases that fast estimation is required at early design phases, or very limited computational resources and/or time margins are available to effectively run the simulated annealing optimization method. To deal with such cases, we present a test scheduling method based on rectangle packing and a greedy heuristic.

In contrast to the Simulated Annealing method, the greedy approach invokes the

RP only once. Initially we generate the sets $S\tau_{C_iV_j}$ for every core C_i - voltage V_j pair. Recall that each set $S\tau_{C_iV_j}$ consists of the rectangles $R_{C_iV_jF_k}, R_{C_iV_jF_{k+1}} \dots$ where only $F_k, F_{k+1}, \dots$ can be used at voltage V_j. Then, we go through each rectangle and each possible placement of that rectangle in the selected virtual bin. Every rectangle-position pair is assigned a score. Then, we select the pair with the highest score, we place the selected rectangle at the selected position, we remove the set of the selected test and we move to the next packing step. This process is completed when one rectangle per set has been scheduled.

The packing and the score calculation are based on the Best-Area-Fit (BAF) policy. In BAF, we place each rectangle to the position that fulfils the following criteria (a) limitations set by constraints C-2, C-5 are not violated, and (b) the occupied area in the rectangle with lower left corner in bin $(0,0)$ and upper right corner (x_{max}, y_{max}) is maximized (x_{max} is equal to bin width and y_{max} is equal to the highest available y-coordinate among the already placed rectangles and the rectangle to be placed). If there are several such valid positions, then we follow a third criterion (c): we select the position that causes the least fragmentation of the area under scope. Specifically, we calculate the total occupied area in the bin. Then, we divide it by the area of the box specified in criterion (b).

Fig. 3.9b shows the test schedule generated by the Greedy (GRD) method on the SoC of the previous examples. It is obvious that the greedy approach packs very effectively the tests at the beginning (see bottom of the bin), but fails at the later stages (see top of the bin). This is a typical behaviour of the greedy algorithm, which focuses on the local rather than the global optimization each time. The greedy method is described in Fig. 3.9(a) and it has worst-case time complexity equal to $O(N_c^3)$.

3.4 Multi-site test optimization for multi-V_{dd} SoCs using space- and time- division multiplexing

3.4.1 Introduction

Multi-site testing (Fig. 3.10) exploits the available ATE channels to concurrently test multiple chips; therefore, in this case the minimization of the time needed for testing a single chip is not the primary optimization factor [107]. Instead, efficient exploitation of the ATE channels to increase the number of SoC copies that are tested in parallel is more beneficial for an entire production batch of SoCs [108]. The number of SoCs that can be tested in parallel depends, among other parameters, on the availability of ATE channels [96, 97, 100]. For a fixed number of ATE channels, a dual objective must be pursued to optimize multi-site testing, namely the minimization of the number of ATE channels needed per SoC and the minimization of the test time per chip. Broadcasting of test data through the same ATE channels to multiple devices is usually avoided because defective dies may corrupt the test results of good dies.

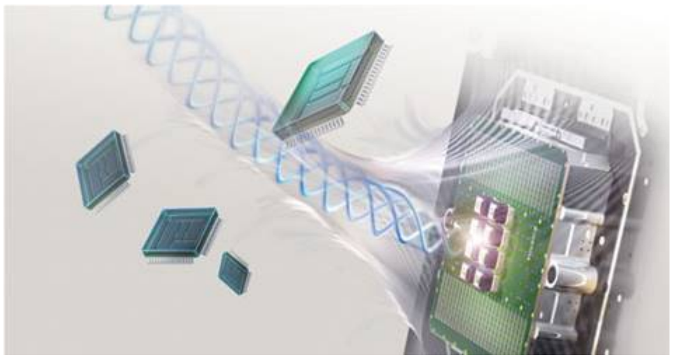


Figure 3.10: Multisite testing [227].

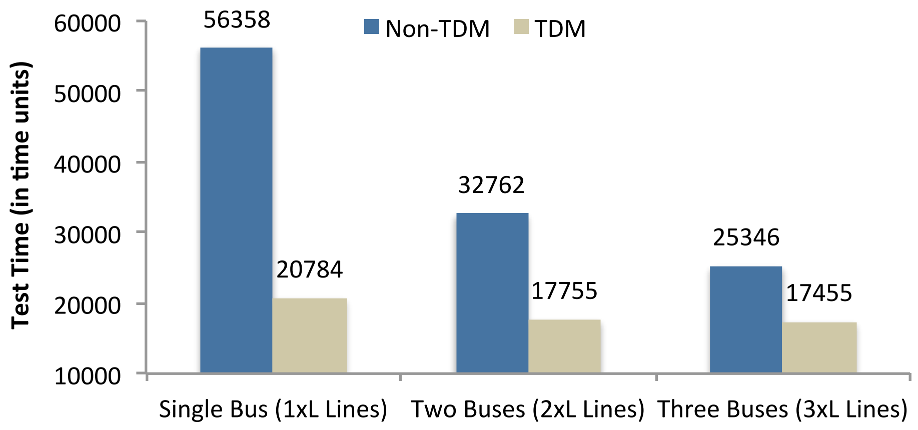


Figure 3.11: Comparisons between TDM and non-TDM scheme.

In order to facilitate efficient multi-site testing, various techniques focus on the optimization of TAM width [108] - [110]. Other techniques reduce the SoC test length through optimization of test scheduling [87], [111] - [113]. Even at the core level, numerous test-resource- partitioning techniques minimize test data volume, test time and the number of ATE channels [106]. Any further reduction in the number of ATE channels to facilitate a higher multi-site efficiency would make the test length per chip to step up in a conversely proportional way, eliminating, thus, any gains due to increased parallelism [108].

We first show that multi-core/multi-V_{dd} SoCs offer increased potential for parallelism, provided that TDM is employed, and a suitable TAM width is used. Specifically, we show that TDM is especially effective for small-bit-width and heavily loaded TAMs, thereby tangibly increasing the effectiveness of multi-site testing. However, inherent limitations of TDM do not allow the fullest possible exploitation of TAM bandwidth. To overcome these limitations, we propose Space-Division-Multiplexing (SDM), which complements TDM. STDM is a unified solution for multi-core/multi-V_{dd} SoCs that combines SDM and TDM and offers very high multi-site test efficiency. It is implemented using a modified version of the SA-RP.

3.4.2 Motivation

TDM is very effective for single-site testing as it minimizes the test time per chip. However, multi-site testing is widely employed in large production lines as it is more efficient than single-site testing [96, 97, 100]. For a given set of ATE channels, the degree of parallelism in multi-site testing is based on the number of ATE channels used per SoC. Therefore, minimization of the number of ATE channels required per SoC directly leads to the maximization of the number of sites tested under certain bounds set by the number of sockets available (in final testing) or by probe card limitations (in wafer testing) within the constraints of the tester load board.

Test data compression, TAM optimization, and test scheduling optimization are used synergistically to reduce both the test length and the required number of ATE channels. Any attempt to further reduce the size of the TAM beyond this point would only lengthen proportionally test time, thus cancelling any parallelization benefits [108]. TDM overcomes this limitation by exploiting the bandwidth of the ATE channels that is left unutilized due to the low shift frequencies used at the lower voltage settings. However, the efficiency of TDM depends on the availability of tests that can be concurrently executed using the same TAM resource. Note that multiple islands and voltage settings impose many constraints that restrict the set of tests that can be concurrently applied [112]. Therefore, in order to boost the performance of TDM, each TAM resource must be connected to a sufficiently large number of cores. Despite being counterintuitive, this unconventional technique is very effective in TDM, as it increases the likelihood that tests can be applied in parallel. Conventional test scheduling techniques, which apply tests in a sequential manner, avoid heavily loaded TAM resources, because they prolong test application time.

In order to highlight this trend, we run simulations using an industrial SoC described in section 4.3. We simulated three different bus architectures for this SoC with one, two and three identical (in size) buses. In the three-buses configuration, each bus was connected to an appropriately selected subset of cores such that the test-time-load for each bus was almost the same. The load of every bus was calculated as the aggregate time of the tests for the cores connected to the bus. The two-bus configuration was generated from the three-bus configuration by removing one randomly selected bus. The set of cores connected to this bus was partitioned into two subsets, such that the aggregate time of the tests for the cores in each subset was almost the same. The cores of one subset were connected to one of the two remaining buses, and the cores of the other subset were connected to the second bus.

Fig. 3.11 presents a comparison between TDM and the test scheduling method of [112] that is also suitable for DVFS-based SoCs with multiple voltage islands (denoted as non-TDM hereafter). The x-axis presents the three configurations, where each bus consists of L TAM lines, and the y-axis presents the test time per device (bus widths and test times are given in normalized units to hide confidential information about this SoC). As we decrease the TAM width, the non-TDM approach worsens considerably in terms

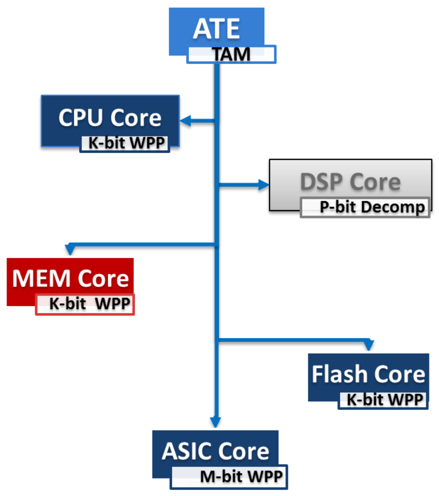


Figure 3.12: An example TAM for core wrappers with different WPP widths.

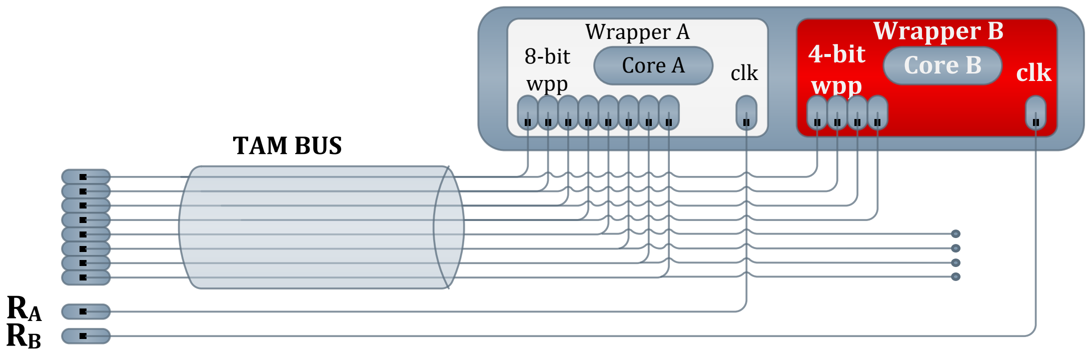
of test time. When the two buses are merged into one, the test time almost doubles, thus offering no gain in multi-site efficiency. In contrast, the test time for the TDM approach increases only slightly as we reduce the number of buses, up to 14.5% per device in the worst case. At the same time, TDM leads to a considerable increase in the number of sites that can be tested in parallel.

Unfortunately, much of the available TAM bandwidth cannot be exploited by TDM, due to the large number of constraints in multi-core/multi-V_{dd} SoCs that prevent tests from being applied in parallel. In addition, as it is illustrated in Fig. 3.12 it is possible that different cores in an SoC will have different WPPs widths. This may happen when hard IP cores are embedded into the SoC or when test decompressors are employed that impose the use of a specific number of inputs channels to provide optimal results (e.g., linear based decompressors fed by external channels in a continuous flow manner [226]). Any bus lines left unutilized by a core cannot be exploited unless fork and join techniques are employed, which introduce additional constraints and increased routing overhead. In order to overcome these limitations, we propose a new DFT architecture, which is referred to as SDM. SDM complements TDM and offers a unified approach for testing multi-core/multi-V_{dd} SoCs.

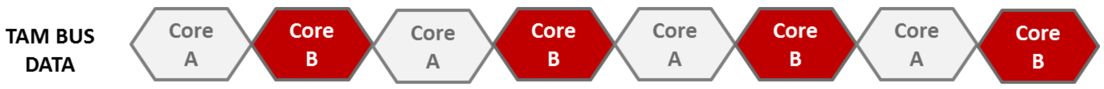
3.4.3 Space-division multiplexing

The objective of SDM, is twofold: (a) enable the use of narrow TAMs to increase multi-site efficiency; (b) overcome mismatch between the WPP size and the size of the bus, and increase the parallelization efficiency of TDM. SDM inserts an interface between the bus and the wrapper whenever their widths do not match. Using SDM, a wide (narrow) bus can be efficiently used to serve a narrow (wide) wrapper interface.

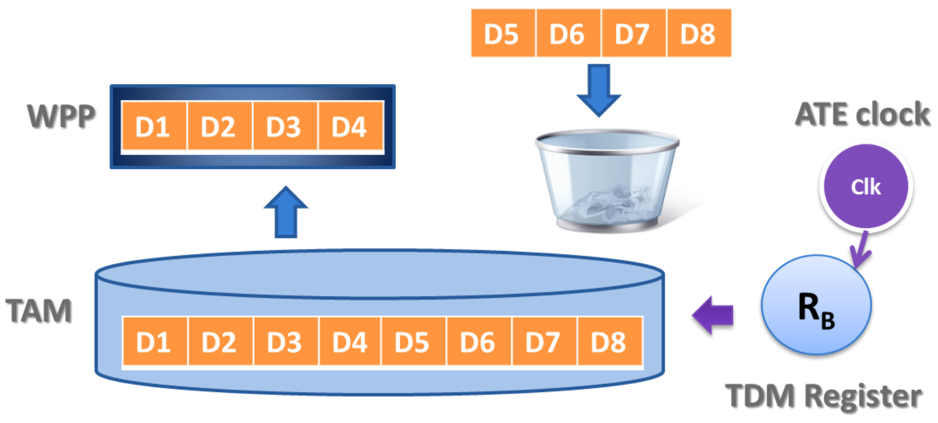
The basic concept of SDM is as follows. Test data at the input of the wrapper are



(c)

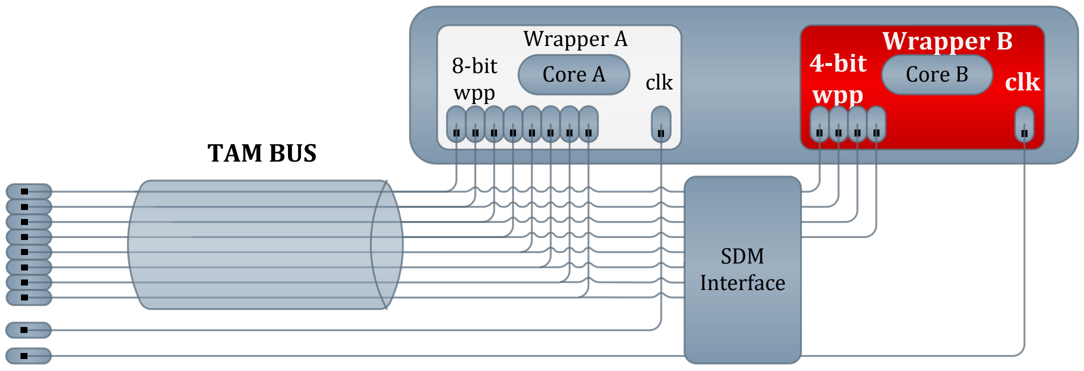


(b)

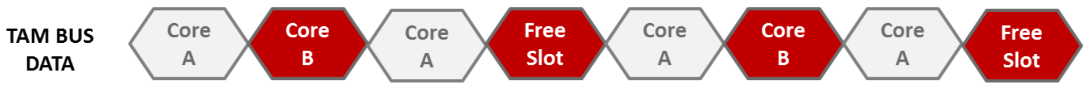


(a)

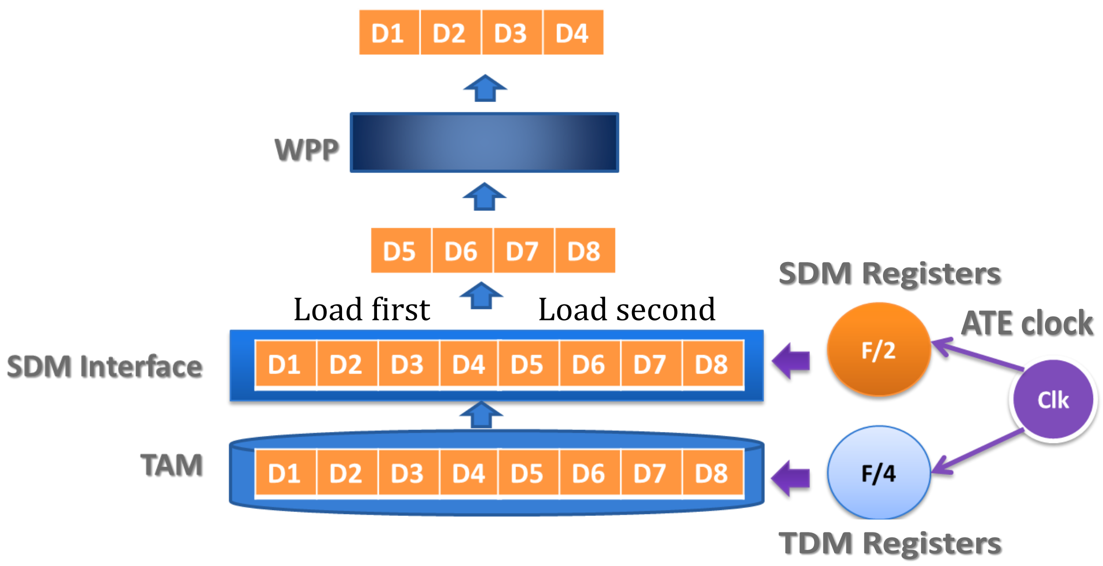
Figure 3.13: An example SoC with a wrapped core that has WPP width less than the TAM bus width using the TDM scheme.



(c)



(b)



(a)

Figure 3.14: An example SoC with a wrapped core that has WPP width less than the TAM bus width using the STDM scheme.

first latched and when they match the width of WPP, they are shifted into the wrapper. SDM considers two cases: when the bus is (a) wider, and (b) narrower than the WPP. In the first case, the test data are latched into an interface register in one clock cycle, and then they are disaggregated and loaded into the wrapper in multiple cycles. In the second case, the test data are latched into the register in multiple cycles, and when they match the width of the WPP, they are transferred to the core in a single cycle.

Example 1. Fig. 3.13c depicts an example multi-V_{dd} SoC with two embedded cores, A and B, that have WPP sizes equal to 8 and 4 bits respectively and an 8-bit TAM bus. We consider that cores A and B are tested in three voltage settings V_1, V_2, V_3. Let the tester provide the ATE_CLK signal and the test data with frequency F. Let the maximum scan frequencies at voltage settings V_1, V_2, V_3 be $F, F/2$ and $F/4$, respectively. Two cyclic registers, R_A and R_B, are responsible to divide the ATE_CLK signal and feed the cores with the required clock signal at each voltage setting. The test data are multiplexed / demultiplexed according to the TDM scheme presented in section 3.3.3. At voltage setting V_2, all cores must be loaded using frequency $F/2$ or lower, when the the test bus supports a maximum shift frequency F. Core A receives the full bandwidth at $F/2$ since the width of the wrapper exactly matches the width of the bus. However, Core B, as shown in 3.13c, can exploit test data only from the half bit-lines of the bus, while the data in the remaining bit-lines become useless. This is illustrated in Fig. 3.13a. Fig. 3.13b demonstrates how the test data for Cores A and B are multiplexed in the TAM bus, when the applied voltage setting and scan frequency for each core is equal to V_2 and $F/2$ correspondingly.

Fig. 3.14c illustrates again the SoC of the previous example (Fig. 3.13c), but in this case an SDM interface has been added to support the TAM bus to load test data in core B. As shown in Fig. 3.14c, the bus is loaded with 8 bits for core B at frequency $F/4$, and the WPP of core B with 4 bits at frequency $F/2$ via an interface provided by the SDM method. Therefore SDM, instead of using the half of the bus at frequency $F/2$, uses the complete bus at frequency $F/4$ and the test length remains the same. Fig. 3.13b and 3.14b illustrates the exploitation of the TAM bandwidth when TDM and STDM are used respectively for testing cores A and B at voltage setting V_2 and scan frequency $F/2$. As shown, in the STDM case, we gain a free slot in the bus for multiplexing test data of an additional core using scan frequency equal to $F/4$. ■

Example 2. Fig. 3.15c depicts another example multi-V_{dd} SoC with two embedded cores, A and B, that have WPP sizes equal to 8 and 16 bits respectively. We consider again that cores A and B are tested in three voltage settings V_1, V_2, V_3 and the tester provide the ATE_CLK signal and the test data with frequency F. The maximum scan frequencies at voltage settings V_1, V_2, V_3 are also considered to be $F, F/2$ and $F/4$, respectively. Two cyclic registers, R_A and R_B, are responsible to divide the ATE_CLK signal and feed the cores with the required clock signal at each voltage setting. The test data are multiplexed / demultiplexed according to the TDM scheme presented in section 3.3.3. At voltage setting V_2, all cores must be loaded using frequency $F/2$ or lower, when the the test bus

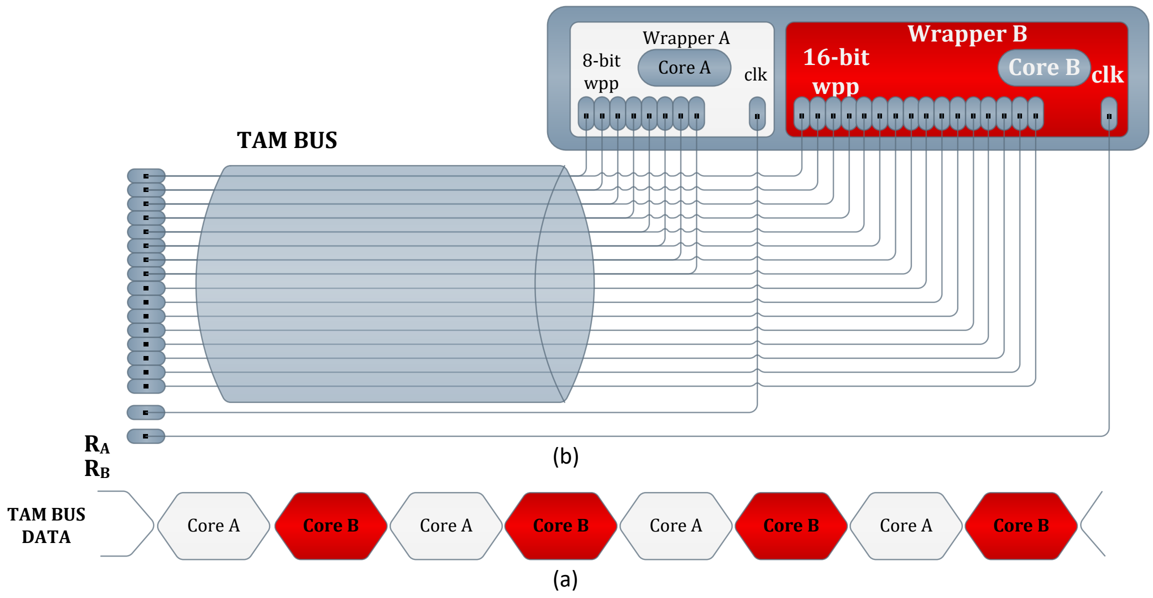


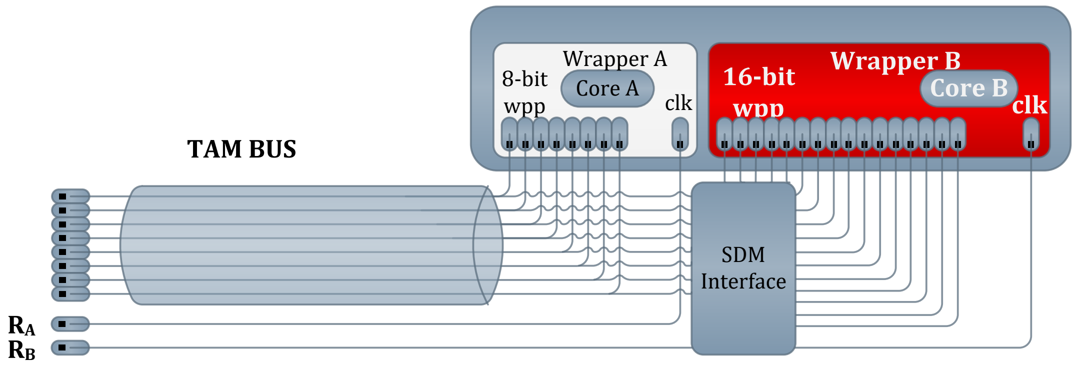
Figure 3.15: An example SoC with a wrapped core that has WPP width greater than the TAM bus width using the TDM scheme.

supports a maximum shift frequency F. As shown in Fig. 3.15b, when the test data are loaded using the TDM scheme, the size of the TAM bus must be 16 bits wide since it is not provided any mechanism able to support wrappers with size of WPP greater than the bus width. Thus, this case is similar to the one in example 1.

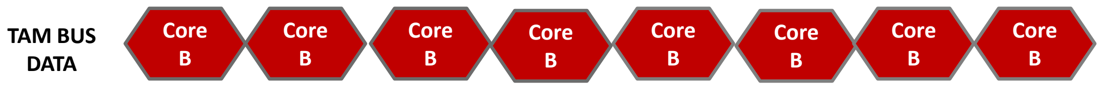
When SDM is applied along with the TDM, an interface mechanism undertakes the task to perform the required matching between the TAM bus and the WPP of the core B wrapper. Then, as shown in Fig. 3.16c, the size of the TAM bus can remain to 8 bits. However, as shown in Fig. 3.16a and Fig. 3.16b, the frequency for loading the TAM bus with test data related to core B increases to F in order to keep the frequency for loading the wrapper of core B at $F/2$. ■

In Case (a) the bus transfers test data at a faster rate than the wrapper can consume. SDM eliminates this gap by enabling low frequency to transmit test data over the bus, and high frequency to transfer the data to the core. Therefore, TAM channels, which would otherwise be left unutilized, are used to reduce the frequency at which test data are transported through the TAM. If TDM is combined with SDM, it can exploit the released frequency to pack more tests in parallel. In other words, STDM reduces test time further, by trading-off the number of ATE channels with the frequency at which test data is transferred. In Case (b), the bus is able to transfer data at a slower rate than the wrapper can consume. In this case, SDM offers the means to increase the frequency at which test data are transferred over the bus in order to shorten the test length.

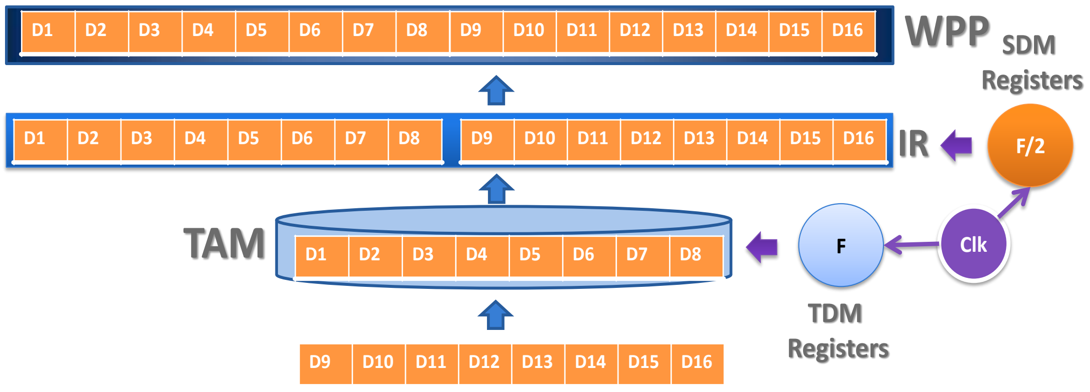
Example 3. Fig. 3.17 depicts an SoC with 3 wrapped cores, A, B, C, and WPP sizes equal to 8, 4, 16 bits for each core respectively. Let the size of the bus be 8 bits and the maximum scan frequencies at voltage settings V_1, V_2, V_3 be $F, F/2$ and $F/4$, respectively.



(c)



(b)



D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15 D16
WPP
SDM
Registers
D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15 D16
IR
F/2
TAM
D1 D2 D3 D4 D5 D6 D7 D8
F
Clk
D9 D10 D11 D12 D13 D14 D15 D16
TDM
Registers

(a)

Figure 3.16: An example SoC with a wrapped core that has WPP width greater than the TAM bus width using the STDM scheme.

At V_2, all cores must be loaded using frequency $F/2$ or lower. Core A receives the full bandwidth at $F/2$ since the width of the wrapper exactly matches the width of the bus. In the case of Core B, the bus is loaded with 8 bits at frequency $F/4$, and the WPP with 4 bits at frequency $F/2$. Therefore, instead of using the half of the bus at frequency $F/2$, SDM uses the complete bus at frequency $F/4$ and the test length remains the same. In case of Core C, the frequency for loading the bus increases to F, while the frequency for loading the wrapper remains at $F/2$. As a result the test length is reduced by half. ■

SDM complements TDM, and the unified STDM scheme fully utilizes the available capacity of the ATE channels. STDM not only exploits unutilized TAM lines, but it also offers high multi-site efficiency even when there are no unutilized TAM lines. This is accomplished by enabling the use of buses that are narrower than the wrappers, with a small additional overhead in test time per device.

The block diagram of STDM for the example SoC is shown in Fig. 3.17. Every core is assigned a small set of registers and a small amount of control logic (not shown in Fig. 3.17 for simplicity and because the hardware overhead is negligible). Three different types of registers are involved:

Interface Register IR: It stores test data from the bus and loads test data into WPP (used only for Core B and Core C that need WPP width adjustment). For Core B, IR_B is 8-bits long and it is loaded in one clock cycle from the bus. The most and least significant nibbles of IR_B are multiplexed at the output of register IR_B and they load WPP in two clock cycles. Register IR_C is 16-bits long; it is loaded in two clock cycles from the bus, and it loads the 16-bit WPP of core C in a single clock cycle.

Bus Control Register BCR: This circular register divides the ATE clock according to a pre-loaded pattern. As the pattern rotates inside BCR, the rightmost cell receives periodically the value of '1' and triggers the loading of the data from the bus directly into WPP (Core A) or into IR (Cores B, C).

Wrapper Control Register WCR: This is similar to BCR and triggers the loading of IR into the wrapper.

Example 4. ATE_CLK, SDM- and TDM- register-oriented and TAM bus waveforms for example 3 are illustrated in Fig. 3.18. Let the frequency of the ATE clock for the SoC of Fig. 3.17 be F. Core A is shifted using frequency $F/2$, as the pattern loaded into BCR_A triggers CLK_A every two clock cycles. Register IR_B is loaded with frequency $F/4$ from the bus. Note that BCR_B triggers IR_B once every four ATE clock cycles. The register IR_B transfers data to the WPP with frequency $F/2$. Note that WCR_B, which controls the loading of IR_B into the WPP, triggers the wrapper once every two clock cycles. Finally, register IR_C is loaded with frequency $F/4$ from the bus — BCR_C triggers IR_C once every four ATE clock cycles. Register IR_C loads the WPP with frequency $F/8$. The patterns loaded into BCR_A, BCR_B, BCR_C have no-overlapping values of '1' to ensure mutually exclusive use of the bus. ■

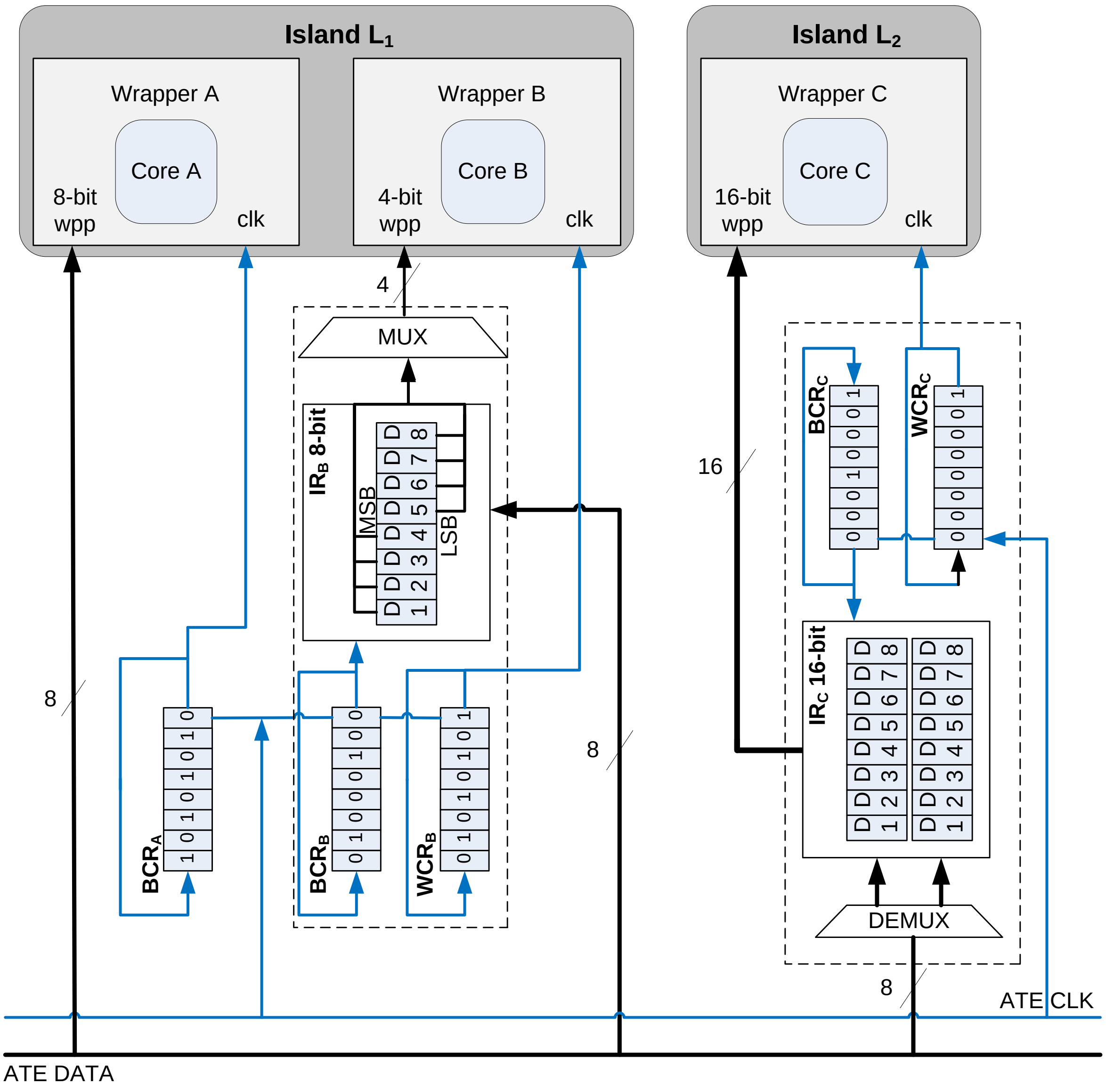


Figure 3.17: Proposed STDM scheme.

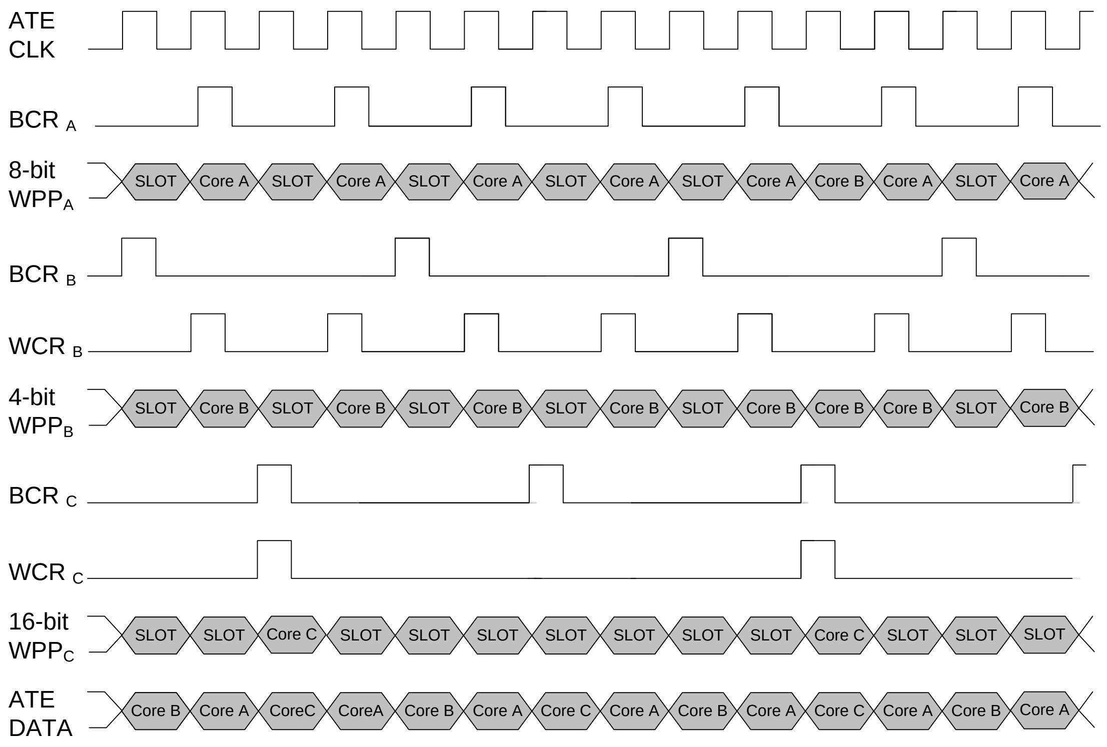


Figure 3.18: ATE_CLK, core-based clock and TAM bus waveforms for the example SoC (Fig.3.17), when STDM is used.

3.4.4 Test scheduling method

We consider a multi-core SoC with I islands $L_1, \ldots, L_I$, N wrapped cores $C_1, \ldots, C_N$ ($N \geq I$), and M voltage levels $V_1, \ldots, V_M$ sorted in descending order (i.e., $V_1 > \cdots > V_M$). Each core (and island) may be tested at all or at a subset of these voltage levels, using one out of S independent buses $B_1, \ldots, B_S$. Each voltage level V_m is associated with one shift frequency F_m that is the maximum rated frequency that can be used for shifting test data at V_m. Note that frequency F_m may differ from core to core and island to island, and it is usually quantized to a pre-specified set of frequencies. Overall we assume that M different shift frequencies $F_1 > \cdots > F_M$ are supported, and every core with maximum (quantized) shift frequency equal to F_m at voltage V_m can use any of the frequencies $F_m, F_{m+1}, \ldots, F_M$ for loading test data.

Let τ_{C_j,V_m,F_i} be the test time needed to test core C_j at voltage V_m using shift frequency F_i ($i \geq m$, where F_m is the highest shift frequency at V_m). τ_{C_j,V_m,F_i} depends on the wrapper depth, $WD(C_j)$, on the number $TP(C_j, V_m)$ of test patterns applied at V_m, and on F_i. When the WPP width W is equal to the size B of the bus, τ_{C_j,V_m,F_i} is approximated using the formula

$$\tau_{C_j,V_m,F_i} = TP(C_j, V_m) \cdot WD(C_j)/F_i \tag{3.14}$$

When the ratio $k = B/W$ of core C_j is equal to $2, 3, \ldots$ or $1/2, 1/3, \ldots$), SDM is

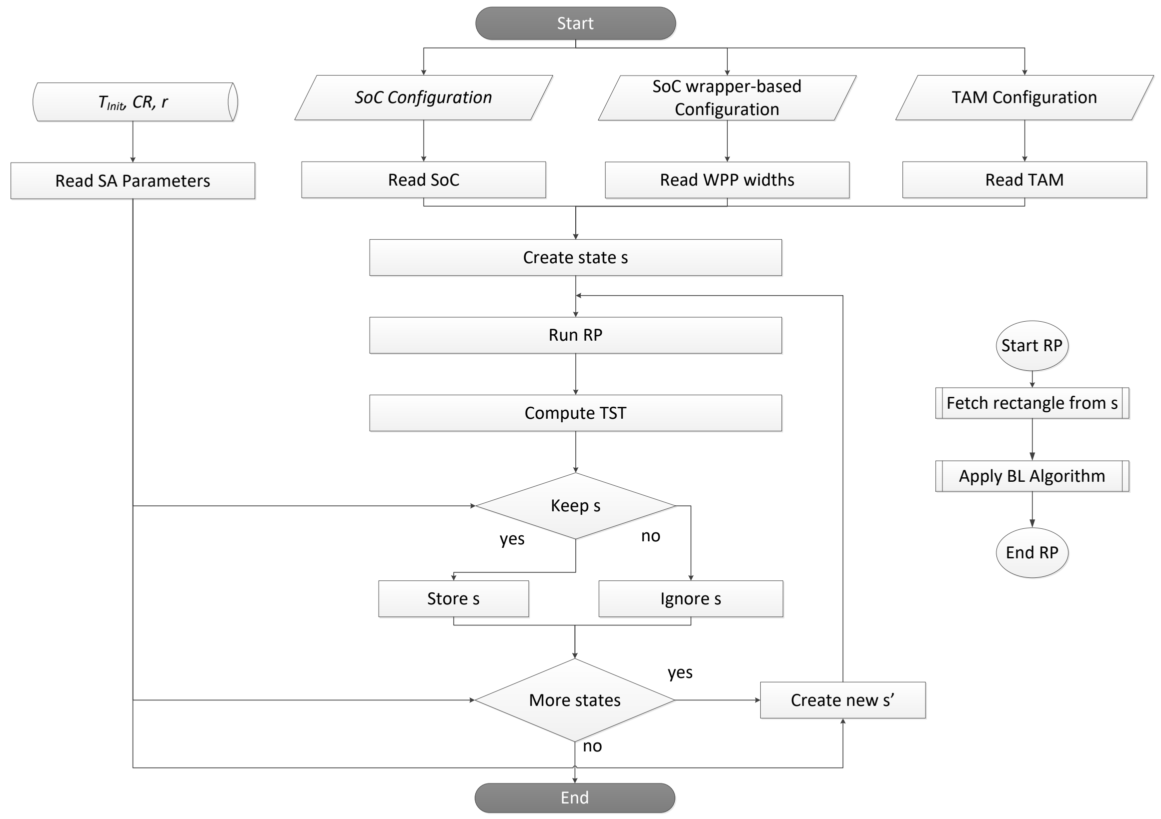


Figure 3.19: STDM test scheduling method flow.

employed, and the test time when the bus is used at frequency F_i is equal to

$$\tau_{C_j,V_m,F_i} = TP(C_j,V_m)\cdot WD(C_j)/(k\cdot F_i). \tag{3.15}$$

The frequency for shifting test data into the core is equal to $k \cdot F_i$, therefore, only the values of i that fulfil the relation $k \cdot F_i \leq F_m$ are used.

Every frequency F_i used for shifting test data into core C_j at any voltage level V_m defines an alternative test, which has a unique length and frequency allocation on the bus. The TDM algorithm selects and schedules the best one for every core-voltage pair, based on the available resources and the inter-core constraints. We have developed a heuristic based on RP and SA. Each test with test time τ_{C_j,V_m,F_i} is modelled as a rectangle with height equal to τ_{C_j,V_m,F_i} and width equal to F_i. Next, a sequence s of such rectangles, one for every core-voltage pair, is packed into Q virtual bins (Q is the number of buses), so that the overall height, i.e. test schedule length TL, is minimum. Each virtual bin has width equal to F_1, which corresponds to the maximum supported scan frequency used for shifting test data.

The heuristic used to implement the packing is based on the Bottom-Left rule [221]. We place each rectangle to the position that (a) fulfills the Multi-V_{dd} testing constraints posed in [95], (b) the y-coordinate of the top side of the rectangle is the smallest. If there are several such valid positions, we select the one that has the smallest x-coordinate value.

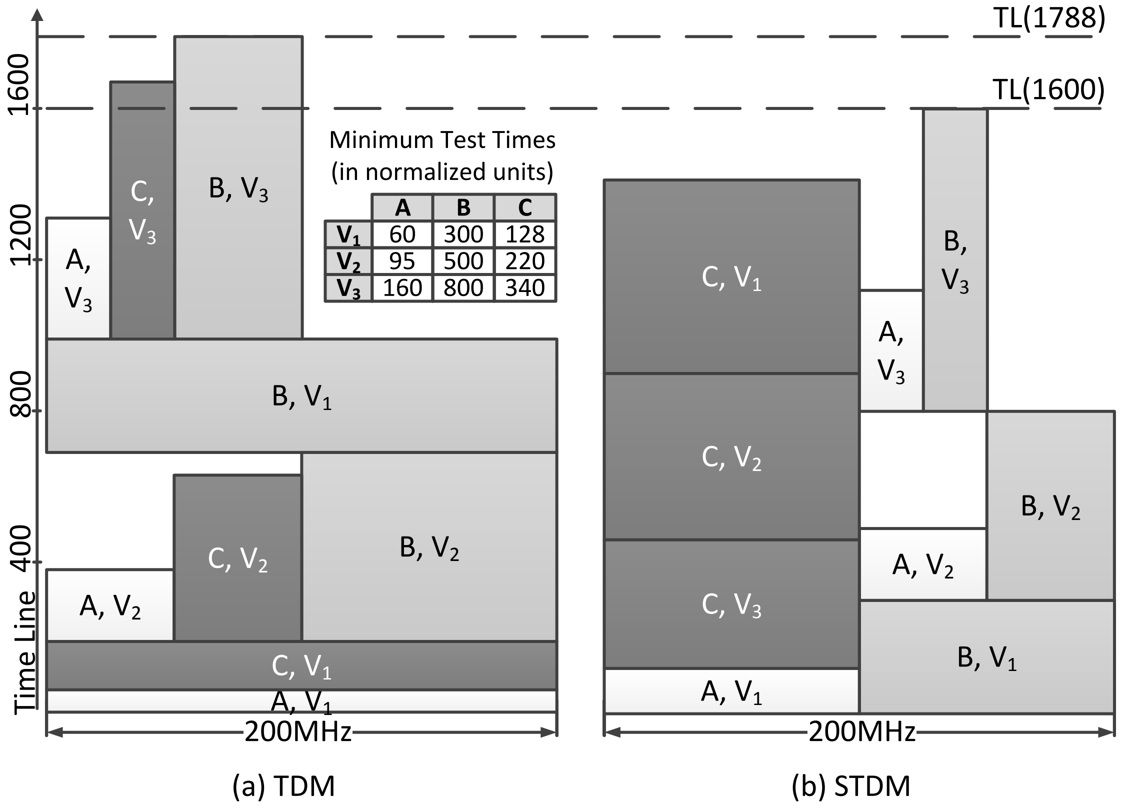
The RP method is combined with an SA heuristic to further optimize its performance. SA uses as energy function $E(s)$, the test schedule length produced by RP. To select the next state in the SA, r randomly selected tests from the current list of scheduled tests are substituted so that a new acceptable schedule of tests can be created. The SA acceptance probability function is derived from the Maxwell-Boltzmann distribution [224]. Both the initial SA temperature T_{init} and cooling rate CR are empirically defined to be a constant value. The flowchart of the proposed method is given in Fig. 3.19.

Example 3. Fig. 3.20 shows the TDM and STDM test schedules for the SoC shown in Fig. 3.17. The x-axis of each bin presents the maximum shift frequency of 200 MHz, and the y-axis presents the test time. The maximum rated shift frequency at V_1, V_2, V_3 is equal to 200, 100 and 50 MHz respectively. The test times at V_1, V_2, V_3 using the maximum rated shift frequency at each voltage setting are shown in the embedded table. The test times at V_1, V_2, V_3 using lower shift frequencies can be derived directly from this table. For example, the test time of Core A at V_1 is $60x1$, $60x2$ and $60x4$ units when 200, 100 and 50 MHz frequency is used respectively. In the TDM case all cores use 200 MHz at voltage level V_1 (all rectangles extend to the full width of the x-axis). At voltage levels V_2, V_3, Core B uses 100 MHz and 50 MHz respectively. At the same voltage levels Core C uses 50 MHz and 25 MHz respectively. In STDM, the tests for Core A retain the same characteristics at V_2 and V_3, while at V_1, the test data are shifted in half of the frequency that was used in TDM. The height of the rectangle is double and the width is half compared to that of TDM. In the case of Core B where width adjustment is used, all tests retain the same time with TDM, even at half the frequency (same height at half the width). In the case of Core C, the frequency used for shifting the test data into the WPP is equal to 50 MHz in all cases, and the frequency for loading the IR_C from the bus is equal to 100 MHz. It is obvious that STDM exploits many different options for sharing the bus and thus it better exploits the bandwidth provided by the ATE. ■

3.5 A branch-&-bound test access mechanism optimization method for multi-V_{dd} SoCs

3.5.1 Introduction

In this work, we propose a Branch-&-Bound (B-&-B) technique to optimize the TAM for minimizing test-time when TDM is used to schedule tests. The proposed technique exploits some unique properties of TDM to set a strict, computationally simple and accurate bounding criterion that enables the B-&-B algorithm to rapidly prune the vast majority of the ineffective TAM configurations. In addition, a fast greedy test-scheduling approach is adopted to evaluate and identify the most effective configurations. The use of the greedy approach is justified by its high correlation (in evaluating TAM designs) with the very effective (but much slower) simulated annealing approach (section 3.3.6). For very large SoCs a global TAM distribution approach is proposed, which optimally



	A	B	C
V_1	60	300	128
V_2	95	500	220
V_3	160	800	340

Figure 3.20: TDM and STDM test schedules.

distributes the TAM lines into multiple SoC areas. To the best of our knowledge, this is the first TAM optimization approach that takes into account the unique characteristics of multi-V_{dd} SoCs and the benefits of the highly effective TDM approach, and offers a complete solution to the test-scheduling problem for multi-V_{dd} SoCs.

The organization of the section is as follows. In section 3.5.2 we motivate this work. section 3.5.3 presents the lower-bound analysis and section 3.5.3 describes the Branch-&-Bound algorithm. Section 3.5.4 presents the global TAM distribution method for very large SoCs.

3.5.2 Motivation

It has been shown in section 3.3 that, independent of the TAM structure of the SoC, the best strategy for testing multi-V_{dd} SoCs is to adopt TDM for test scheduling. However, in order to maximize the benefits offered by TDM, the underlying TAM should be tailored to TDM. To motivate this work and show how important the TAM configuration is when TDM is adopted, we used TDM to schedule the tests for two industrial SoCs [231], SoC-A and SoC-B, for different TAM configurations. Both SoCs consist of digital cores and are targeted for portable wireless applications. They are extremely power conscious, and they have various programmable levels of power control, either through external power supply control or through internal settings. SoC-A has 4 voltage islands $I_1^A, I_2^A, I_3^A, I_4^A$, 9 cores

V_{dd}	I_1^A		I_2^A				I_3^A					I_4^A	
level	C_1^A	F^m	C_2^A	C_3^A	C_4^A	F^m	C_5^A	C_6^A	C_7^A	C_8^A	F^m	C_9^A	F^m
V_1	300	266	60	128	262	200	N/A	N/A	128	600	133	55	200
V_2	500	200	95	220	500	100	475	375	170	950	100	N/A	N/A
V_3	800	100	160	340	700	50	800	600	300	1600	50	N/A	N/A
V_4	1600	50	N/A	N/A	N/A	N/A	1600	1200	600	3200	25	N/A	N/A

Table 3.2: SoC-A Test Times (in Normalized Time Units) At Maximum Scan Frequencies F^m (In MHz) [231].

$C_1^A, \ldots, C_9^A$ and 124 clock domains. SoC-B has 7 voltage islands $I_1^B, \ldots, I_7^B$, 15 cores $C_1^B, \ldots, C_{15}^B$ and 225 clock domains. SoC-A can be set to up to 4 voltage settings, while SoC-B can be set to up to 6 voltage settings. Table 3.2 presents the minimum testing times for all cores C_i^A of SoC-A at the various voltage levels V_j. Correspondingly, tables 3.3 and 3.4 present the minimum testing times for all cores C_i^B of SoC-B at the various voltage levels V_j. The test data for the cores are grouped into columns according to the island they belong to. The test times are calculated using the maximum scan frequencies, denoted as 'F^m', that do not cause scan chain timing violations at any core for shifting at the corresponding voltage level (they are presented in normalized time units, so as not to reveal confidential data). The maximum scan frequencies are presented in the last column of each island (they are reported in MHz and they are different for every voltage setting and every island). The entries denoted as 'N/A' correspond to cores that either do not operate at the respective voltage settings or they are not tested at these voltage settings.

For SoC-A, we generated 37,000 different random TAM configurations, assuming in all cases the same number of ATE channels. In these configurations, we varied the number of buses, their size, and the assignments of cores to buses, and for every configuration we applied the TDM scheme to optimize the test time. The minimum test times found (in normalized time units) in all these cases were in the range [4K, 90K] with average test time μ = 24K and standard deviation σ = 15K (1K=10^3). We repeated the same experiment for SoC-B that is larger than SoC-A, and we applied the TDM-based test-scheduling method on 4.3 million different random configurations by varying the same parameters (again the number of ATE channels was the same in all cases). The minimum test times for SoC-B were found in the range [10K, 330K] as shown in Fig. 3.21, with μ = 85K and σ = 45K.

It is obvious that TDM alone cannot minimize the test time unless the TAM is also optimized. However, the identification of an optimal TAM configuration among millions of possible configurations is a very difficult problem. Moreover, the complexity of scheduling tests for multi-V_{dd} SoCs [88] makes it impractical to optimize both the TAM configuration and the test schedule at the same time. Besides area and routing constraints, which are similar in both single-V_{dd} and multi-V_{dd} SoCs, single-V_{dd} TAM optimization methods do not take into account the specific constraints and limitations of multi-V_{dd} SoCs [88] and

V_{dd}	I_1^B			I_2^B				I_3^B			
level	C_1^B	C_2^B	F^m	C_3^B	C_4^B	C_5^B	F^m	C_6^B	C_7^B	C_8^B	F^m
V_1	900	300	400	700	100	550	200	N/A	188	600	266
V_2	1200	396	300	1400	200	1100	100	475	370	950	200
V_3	1350	450	266	2800	400	2200	50	700	500	1600	100
V_4	1800	600	200	N/A	N/A	N/A	N/A	1400	1000	3200	50
V_5	3600	N/A	100	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
V_6	7200	N/A	50	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 3.3: SoC-B Test Times (in Normalized Time Units) At Maximum Scan Frequencies F^m (in MHz) [231].

V_{dd}	I_4^B			I_5^B			I_6^B			I_7^B	
level	C_9^B	C_{10}^B	F^m	C_{11}^B	C_{12}^B	F^m	C_{13}^B	C_{14}^B	F^m	C_{15}^B	F^m
V_1	700	N/A	200	N/A	165	300	500	125	200	1300	200
V_2	924	198	150	900	192	200	N/A	N/A	N/A	N/A	N/A
V_3	1050	225	133	N/A	250	150	N/A	N/A	N/A	N/A	N/A
V_4	1400	300	100	N/A	500	75	N/A	N/A	N/A	N/A	N/A
V_5	2800	N/A	50	N/A	1000	38	N/A	N/A	N/A	N/A	N/A
V_6	5600	N/A	25	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 3.4: SoC-B Test Times (in Normalized Time Units) At Maximum Scan Frequencies F^m (in MHz) [231].

they do not exploit the benefits of TDM. ILP approaches that have been traditionally used for TAM optimization of single-V_{dd} SoCs are not suitable for the joint optimization of the TAM structure and test-scheduling for the multi-V_{dd} SoCs. As it was shown in section 3.3.5, even for the single task of optimizing the test schedule on a pre-determined TAM structure, ILP requires a prohibitively large number of constraints. Therefore, it is impractical to use ILP modeling to optimize both the test schedule and the TAM structure at the same time, whereas LP-relaxation and branch-and-bound pruning have been already shown to give much inferior results [88]. We propose the first TAM optimization method for multi-V_{dd} SoCs that employ the TDM test mechanism. The main contributions of this work are:

1. We derive a closed-form equation to compute the lower bound on the test time using TDM for any given TAM configuration.

2. We propose a branch-&-bound approach with a very strict bounding criterion that rapidly prunes the vast majority of the ineffective TAM configurations and quickly identifies the most effective ones. This innovative bounding criterion captures the beneficial properties of TDM and it systematically estimates the effectiveness of

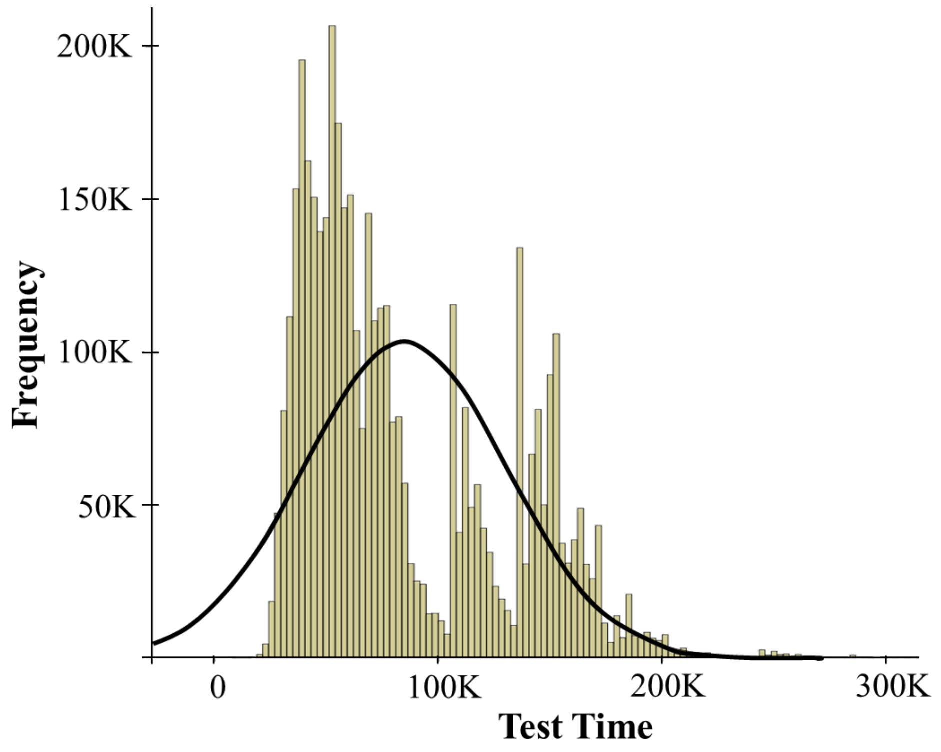


Figure 3.21: Variations in test time of the industrial SoC-B due to TAM configuration.

TDM for any TAM design without actually applying the TDM-based test scheduling algorithm.

3. We show that by evaluating the TAM configurations in a specific order, the lower bound becomes an even more efficient bounding criterion that prunes more than 99% of the possible TAM configurations.

4. For evaluating the TAM configurations that are not pruned by the bounding criterion, we show that a fast greedy test-scheduling approach is equally effective as the slow SA-RP method.

5. For very large SoCs we propose a global distribution scheme, which distributes the TAM lines into multiple areas, and it optimizes the TAM structure at each area separately.

3.5.3 Lower-Bound Analysis

We consider a multi-core SoC with N_I voltage islands $L_1, \ldots, L_{N_I}$, N_c wrapped cores $C_1, \ldots, C_{N_c}$, N_v voltage levels $V_1, \ldots, V_{N_v}$ sorted in descending order (i.e., $V_1 > \cdots > V_{N_v}$) and N_{ATE} tester channels. The TAM consists of N_q test buses $B_1, ..., B_{N_q}$ with sizes (in bit lines) equal to $BS_1, ..., BS_{N_q}$ respectively ($BS_1 + \cdots + BS_{N_q} \leq N_{ATE}$). Each core is tested at a subset of the N_v voltage levels. At each voltage V_j there is a maximum frequency $F_{i,j}^{max}$ that can be used for shifting test data into core C_i (as the voltage level reduces, the maximum shift frequency reduces and the test time increases). $F_{i,j}^{max}$ differs from core to core; thus it is quantized to a pre-specified set of N_f frequencies $F_1 > \cdots > F_{N_f}$ supported by TDM. Every core C_i with maximum (quantized) frequency $F_{i,j}^{max}$ can use frequencies $F_m, F_{m+1}, \ldots, F_{N_f}$ that are lower or equal to $F_{i,j}^{max}$.

The time required for testing core C_i at voltage V_j using shift frequency F_k is equal to $\tau_{C_i V_j F_k}$ and it depends on the scan chain-wrapper depth, WD_{C_i}, the number of test patterns applied TP_{C_i,V_j}, and the shift frequency F_k, according to the equation (in large SoCs capture cycles can be ignored)

$$\tau_{C_i V_j F_k} = TP_{C_i,V_j} \cdot \frac{WD_{C_i}}{F_k} \tag{3.16}$$

For the shake of simplicity in the mathematical analysis, we assume that the wrapper and the internal scan chains of the core are flexible and there is no embedded compression mechanism. Therefore, when the size of the bus of core C_i increases (decreases) the test time $\tau_{C_i V_j F_{i,j}^{max}}$ decreases (increases) proportionally. In addition, when the shift frequency F_k increases (decreases) the test time decreases (increases) proportionally.

When test compression is employed, the test time depends on the number of input channels of the decompressor, and the number of clock cycles needed to load each test pattern into the core. Different bus configurations correspond to different configurations of the decompressor. Therefore, the inverse proportional relationship between the number of input channels and the time for shifting test data into the core does not necessarily hold in all cases. Nevertheless, the exact same analysis holds for cores with or without compression, with the only difference being the way test time is calculated for the various bus and/or decompressor configurations.

A low bound $LB^{(1)}$ on the test time of a multi-V_{dd} SoC for a specific TAM configuration can be calculated if we assume that: a) every bus transfers test data with infinite shift frequency, and b) every core C_i tested at V_j uses the maximum frequency $F_{i,j}^{max}$ permitted by V_j. Both assumptions imply that the bus frequency is as high as it is required for all cores connected to that bus to be tested concurrently at their maximum shift frequencies. Even though most realistic SoCs will not meet these conditions, they are valid for lower-bound analysis as they model the ideal conditions under which the test time is minimized (test time cannot drop below this value without test coverage loss). Therefore, they can be justifiably used for any SoC.

The minimum time $\tau_{L_l V_j}^{min}$ needed for testing all cores in voltage island L_l at voltage V_j is the maximum among the individual test times for any of its cores, that is

$$\tau_{L_l V_j}^{min} \geq \max_{C_i \in L_l} \{\tau_{C_i V_j F_{i,j}^{max}}\} \tag{3.17}$$

Every voltage island L_l is tested at a subset $V_{L_l}^S$ of the voltage levels. Thus, $\tau_{L_l}^{min}$ for L_l is the sum of the minimum test times at every voltage of $V_{L_l}^S$:

$$\tau_{L_l}^{min} = \sum_{V_j \in V_{L_l}^S} \tau_{L_l V_j}^{min} \geq \sum_{V_j \in V_{L_l}^S} \max_{C_i \in L_l} \{\tau_{C_i V_j F_{i,j}^{max}}\} \tag{3.18}$$

The low bound $LB^{(1)}$ on test time at the SoC level equals the highest among the minimum time required for testing any one of its voltage islands:

$$LB^{(1)} = \max\{\tau_{L_1}^{min}, \tau_{L_2}^{min} \ldots \tau_{L_{N_I}}^{min}\} \tag{3.19}$$

The minimum test time of the SoC, τ_{SoC}^{min}, is always higher than $LB^{(1)}$, because the assumption that an infinite frequency can be used at each bus to shift test data into the cores cannot be achieved in practice. However, there are many cases that the shift frequencies used at the core level are much lower than the TAM frequency used at the SoC level due to voltage related and scan-chain related constraints. In these cases the assumption becomes more realistic, and τ_{SoC}^{min} approaches $LB^{(1)}$. In the rest of the cases where the TAM frequency is not very high, the assumption becomes less realistic and $LB^{(1)}$ is much lower than τ_{SoC}^{min}.

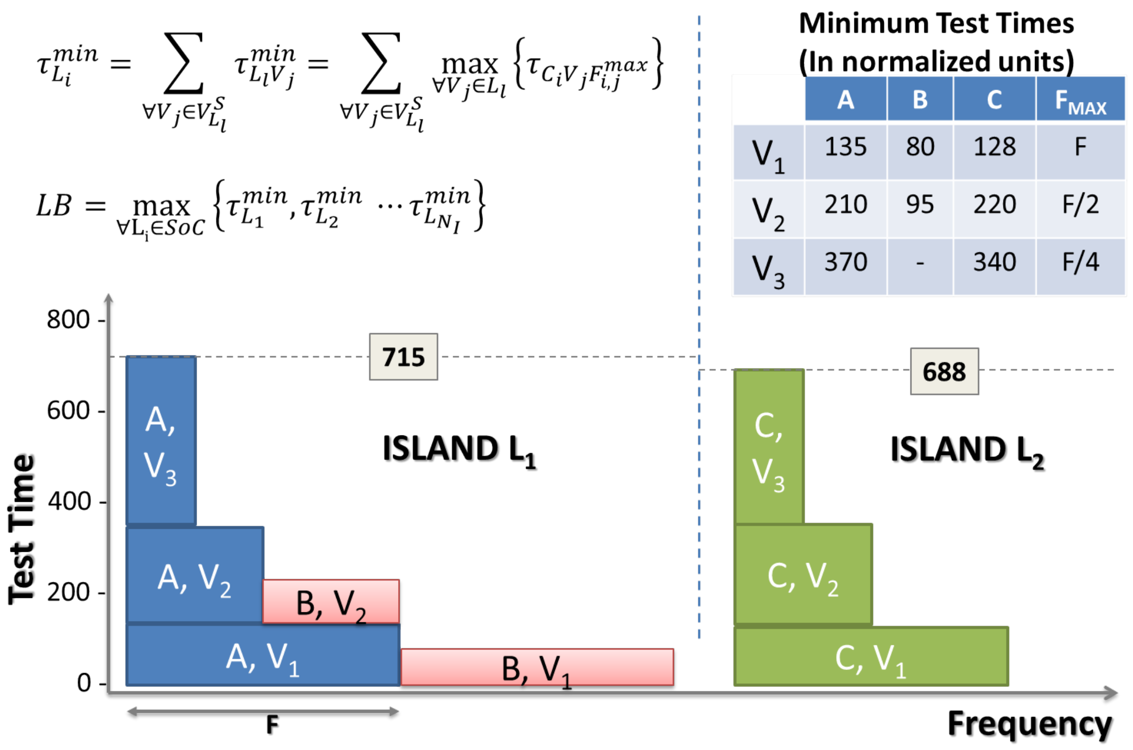
Example 7. Let us again consider the SoC of Fig. 3.1, with voltage levels V_1, V_2, V_3, and nominal scan frequencies at each one of them F, $F/2$ and $F/4$, respectively. Let the tester provide the ATE_CLK signal with unlimited frequency. The embedded table in Fig. 3.22 shows the test times per core at V_1, V_2, V_3, when the maximum rated shift frequency is used at each voltage level (dashes indicate that the corresponding tests are omitted). As shown, the minimum time needed for testing cores A, B in voltage island L_1 is equal to the test time required for testing core A in voltage levels V_1, V_2, V_3, namely 715 time units. The minimum time needed for testing core C in voltage island L_2 is 688 time units. Since islands L_1, L_2 can be tested in parallel, the minimum test time for testing the whole SoC is equal to the test time of the island with the maximum test time, namely the test time of island L_1, which is equal to 715 time units. ■

In order to estimate a low bound which is closer to τ_{SoC}^{min} for those cases, we follow a different approach. Let us consider bus B_m and a number of cores $C_1^m, C_2^m, \ldots$ connected to this bus. Then, the minimum time $T_{B_m}^{min}$ required to schedule the tests of all the cores connected to B_m is given by the following relation

$$\tau_{B_m}^{min} \geq \sum_{i,j} \tau_{C_i^m V_j F_{ATE}} \tag{3.20}$$

where $\tau_{C_i^m V_j F_{ATE}}$ is the time required for testing core C_i^m at voltage level V_j using the highest shift frequency F_{ATE} supported by the bus. We note that this formula is true even when the shift frequency F_{ATE} is not supported by C_i^m at voltage level V_j because $F_{ATE} \geq F_{i,j}^{max}$ and thus $\tau_{C_i^m V_j F_{ATE}} \leq \tau_{C_i^m V_j F_{i,j}^{max}}$.

Intuitively, relation (3.20) provides the minimum time for each bus when the full bandwidth of the bus is exploited. For example, all the rectangles in Fig. 3.2 would be deployed to the full width of the bin, with a proportional reduction of their height (we remind that the area of every rectangle is constant and it does not depend on the shift frequency). If every test occupied the full width of the bin all tests would be scheduled sequentially leaving no empty space in the bin. In that way the test time would be minimum. Even though this is not a realistic scenario due to the low shift frequency



$$\tau_{L_i}^{min} = \sum_{\forall V_j \in V_{L_l}^S} \tau_{L_l V_j}^{min} = \sum_{\forall V_j \in V_{L_l}^S} \max_{\forall V_j \in L_l} \left\{\tau_{C_i V_j F_{i,j}^{max}}\right\}$$

$$LB = \max_{\forall L_i \in SoC} \left\{\tau_{L_1}^{min}, \tau_{L_2}^{min} \cdots \tau_{L_{N_I}}^{min}\right\}$$

Minimum Test Times
(In normalized units)

	A	B	C	F_{MAX}
V_1	135	80	128	F
V_2	210	95	220	F/2
V_3	370	-	340	F/4

Figure 3.22: Lower bound calculation in an example SoC.

imposed by scan-chain-timing violations, it provides a theoretical low bound on the test time of the SoC.

The low bound on the test time for an SoC with N_q test buses is given by the following relation

$$LB^{(2)} = \max\{\tau_{B_1}^{min}, \tau_{B_2}^{min} \dots \tau_{B_{N_q}}^{min}\} \tag{3.21}$$

Based on (3.19), (3.21) the overall lower bound for testing the SoC is given by the following relation:

$$LB = \max\{LB^{(1)}, LB^{(2)}\} \tag{3.22}$$

and obviously $\tau_{SoC}^{min} \geq LB$. Note that this relation is valid for all cores, either they employ test compression or not, since the test data (compressed or not) travel through the same bus lines and have the same shift frequency limitations.

Theorem 3.1. *Let us consider a TAM configuration CNF_a with N_q test buses $B_1, B_2, ..., B_{N_q}$ and lower bound on test time equal to LB_{CNF_a}. Let us also assume a second TAM configuration CNF_b, where one of the buses of CNF_a, say B_m, is split into two buses B_{m_1}, B_{m_2}. Every core C_i^m connected to B_m in CNF_a is connected to either B_{m_1} or B_{m_2} in CNF_b (the remaining buses and the cores connected to these buses remain unaffected in CNF_b). Then we have $LB_{CNF_b} \geq LB_{CNF_a}$.*

Proof. In order to show that $LB_{CNF_b} \geq LB_{CNF_a}$, it suffices to show that the following two conditions are true:

1. $LB^{(1)}_{CNF_b} \geq LB^{(1)}_{CNF_a}$
2. $LB^{(2)}_{CNF_b} \geq LB^{(2)}_{CNF_a}$

Condition (1). We will show that $LB^{(1)}_{CNF_b} \geq LB^{(1)}_{CNF_a}$. The core(s) with the highest test time at each voltage island and voltage level determine(s) the minimum test time of the SoC and thus the value of $LB^{(1)}$. There are two cases:

1. The core(s) with the longest tests are not connected to B_m. Then, the minimum test time(s) for these core(s) remain the same in CNF_b. If they also remain the largest ones among the minimum test times of all cores, then $LB^{(1)}_{CNF_b} = LB^{(1)}_{CNF_a}$; else other tests become the longest ones due to the reduction of the size of B_m in CNF_b, thus $LB^{(1)}_{CNF_b} > LB^{(1)}_{CNF_a}$. Overall we have $LB^{(1)}_{CNF_b} \geq LB^{(1)}_{CNF_a}$.
2. One or more of the core(s) with the highest test time(s) are connected to the bus B_m. In that case the test time(s) for these cores increase as the size of bus B_m decreases, and thus $LB^{(1)}_{CNF_b} > LB^{(1)}_{CNF_a}$

Therefore it is obvious that $LB^{(1)}_{CNF_b} \geq LB^{(1)}_{CNF_a}$.

Condition (2). We will show that $LB^{(2)}_{CNF_b} \geq LB^{(2)}_{CNF_a}$. Let us assume that the bus B_m consists of BS_m bit lines, and buses B_{m_1}, B_{m_2} consist of BS_{m_1} and BS_{m_2} bit lines respectively, with

$$BS_m = BS_{m_1} + BS_{m_2} \tag{3.23}$$

Let C^{m_1}, C^{m_2} be the set of cores connected to buses B_{m_1}, B_{m_2} respectively, which were all connected to bus B_m in CNF_a, i.e., $C^m = C^{m_1} \cup C^{m_2}$. Then we have

$$\begin{aligned} LB^{(2)}_{CNF_a} &= \max\{\tau^{min}_{B_1}, \ldots, \tau^{min}_{B_m}, \ldots, \tau^{min}_{B_{N_q}}\} \\ LB^{(2)}_{CNF_b} &= \max\{\tau^{min}_{B_1}, \ldots, \tau^{min}_{B_{m_1}}, \tau^{min}_{B_{m_2}}, \ldots, \tau^{min}_{B_{N_q}}\} \end{aligned} \tag{3.24}$$

We will examine two different cases.

1. In the first case we assume that bus B_m is the most heavily loaded, and thus $LB^{(2)}_{CNF_a} = \tau^{min}_{B_m}$. Then from (3.20), (3.21) we have

$$LB^{(2)}_{CNF_a} = \sum_{i,j} \tau_{C^m_i V_j F_{ATE}} \tag{3.25}$$

or equivalently

$$LB^{(2)}_{CNF_a} = \sum_{i,j} \tau_{C_i^{m_1} V_j F_{ATE}} + \sum_{i,j} \tau_{C_i^{m_2} V_j F_{ATE}} \tag{3.26}$$

or equivalently

$$\tau^{min}_{B_m} \geq \sum_{i,j} \tau_{C_i^{m_1} V_j F_{ATE}} + \sum_{i,j} \tau_{C_i^{m_2} V_j F_{ATE}} \tag{3.27}$$

In CNF_b the bus B_m of size BS_m is split into the buses B_{m_1}, B_{m_2}, of sizes BS_{m_1}, BS_{m_2} respectively. Therefore the test time of each core $C_i^{m_1}$ ($C_i^{m_2}$) connected to B_{m_1} (B_{m_2}) is increased by a proportion BS_m/BS_{m_1} (BS_m/BS_{m_2}) as compared to the test time of the same core connected to B_m (we note that B_m is wider than B_{m_1}, B_{m_2} by a factor of BS_m/BS_{m_1}, BS_m/BS_{m_2} respectively). Thus we have

$$\begin{aligned} \tau^{min}_{B_{m_1}} &= \sum_{i,j} \frac{BS_m}{BS_{m_1}} \tau_{C_i^{m_1} V_j F_{ATE}} \\ \tau^{min}_{B_{m_2}} &= \sum_{i,j} \frac{BS_m}{BS_{m_2}} \tau_{C_i^{m_2} V_j F_{ATE}} \end{aligned} \tag{3.28}$$

We will show that $LB^{(2)}_{CNF_b} \geq LB^{(2)}_{CNF_a}$ by contradiction. Let us assume that the hypothesis $LB^{(2)}_{CNF_b} < LB^{(2)}_{CNF_a}$ is true. Since $\tau^{min}_{B_m} > tau^{min}_{B_k}$ $\forall k \neq m$ and based on a) the above hypothesis and b) the relation (3.24) we have that

$$\max\{\tau^{min}_{B_1}, \ldots, \tau^{min}_{B_{m_1}}, \tau^{min}_{B_{m_2}}, \ldots, \tau^{min}_{B_{N_q}}\} < \\ \max\{\tau^{min}_{B_1}, \ldots, \tau^{min}_{B_m}, \ldots, tau^{min}_{B_{N_q}}\}$$

or equivalently

$$\tau^{min}_{B_m} > \max\{\tau^{min}_{B_{m_1}}, \tau^{min}_{B_{m_2}}\}$$

Without loss of generality we may assume that

$$\tau^{min}_{B_{m_1}} \geq \tau^{min}_{B_{m_2}} \tag{3.29}$$

and thus we have

$$\tau_{B_m}^{min} > \tau_{B_{m_1}}^{min} \tag{3.30}$$

From (3.28), (3.29) we have that

$$\frac{BS_{m_2}}{BS_{m_1}} \geq \frac{\sum_{i,j} \tau_{C_i^{m_2} V_j F_{ATE}}}{\sum_{i,j} \tau_{C_i^{m_1} V_j F_{ATE}}} \tag{3.31}$$

From (3.27), (3.28), (3.30) we have that

$$\sum_{i,j} \tau_{C_i^{m_1} V_j F_{ATE}} + \sum_{i,j} \tau_{C_i^{m_2} V_j F_{ATE}} > \frac{BS_m}{BS_{m_1}} \sum_{i,j} \tau_{C_i V_j F_{ATE}}^{m_1}$$

or equivalently

$$\frac{\sum_{i,j} \tau_{C_i^{m_2} V_j F_{ATE}}}{\sum_{i,j} \tau_{C_i^{m_1} V_j F_{ATE}}} > \frac{BS_m - BS_{m_1}}{BS_{m_1}}$$

and from 3.23 we have

$$\frac{\sum_{i,j} \tau_{C_i^{m_2} V_j F_{ATE}}}{\sum_{i,j} \tau_{C_i^{m_1} V_j F_{ATE}}} > \frac{BS_{m_2}}{BS_{m_1}} \tag{3.32}$$

From (3.31), (3.32) we have that $BS_{m_2}/BS_{m_1} > BS_{m_2}/BS_{m_1}$ which is a contradiction.

2. In the second case we assume that the bus B_m is not the one with the heaviest load, and let now B_k be the bus with the heaviest load. Since B_k remains unaffected, we can have one of the following two sub-cases:

   (a) $\tau_{B_k}^{min} \geq \max\{\tau_{B_{m_1}}^{min}, \tau_{B_{m_2}}^{min}\}$, and thus $LB_{CNF_b}^{(2)} = LB_{CNF_a}^{(2)}$,

   (b) $\tau_{B_{m_1}}^{min} > T_{B_k}^{min}$ or $\tau_{B_{m_2}}^{min} > \tau_{B_k}^{min}$ and thus $LB_{CNF_a}^{(2)} = \max\{\tau_{B_{m_1}}^{min}, \tau_{B_{m_2}}^{min}\}$ and thus $LB_{CNF_b}^{(2)} > LB_{CNF_a}^{(2)}$

   Therefore it is obvious that $LB_{CNF_b}^{(2)} \geq LB_{CNF_a}^{(2)}$ in all cases.

∎

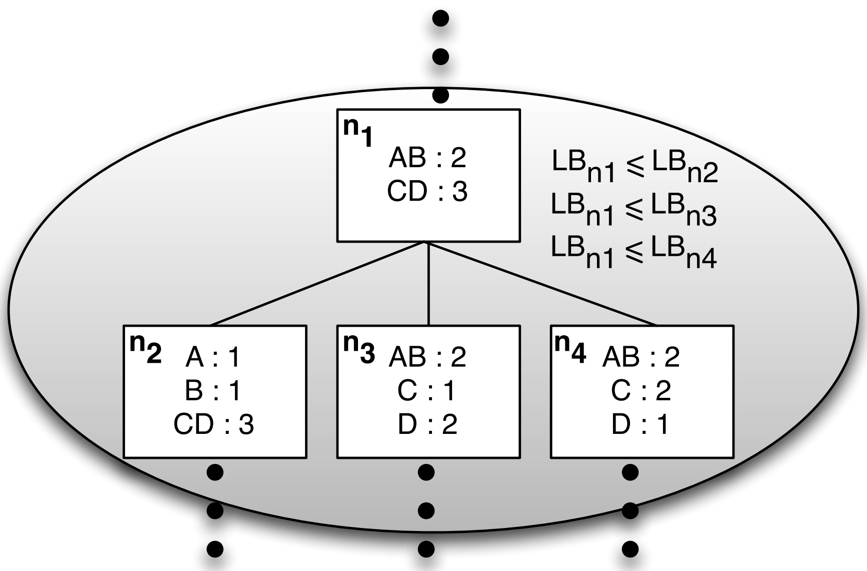


Figure 3.23: A part of an example tree.

According to Theorem 3.1, when any of the buses is split into two or more buses, the value of LB either increases or it remains the same. We exploit this property in order to quickly prune the search space consisting of all the possible TAM configurations. We represent the space of all TAM configurations as a tree, where each node represents one specific TAM configuration. At the root node, all cores are connected at one bus with size equal to N_{ATE} bit lines. The immediate successors of the root are all possible TAM configurations where the initial bus is split into two buses that take all possible sizes. For each combination of bus sizes, we consider all partitions of the cores into two sets that are connected on the two buses. This is recursively applied and every child node is generated from its parent by splitting one bus into two sub-buses where the cores of the initial bus are distributed among the sub-buses in all possible ways.

Example 8. In Fig. 3.23 we present a part of the search tree for an SoC with cores A, B, C, D. Node n_1 represents a configuration of two buses with 2 and 3 lines respectively. Cores A, B are connected to the first bus, while cores C, D are connected to the second bus. Every successor of n corresponds to a configuration of three buses, which is generated by splitting one of the buses of n_1 into two buses of smaller width. At n_2 the first bus of size 2 is split into two buses of sizes 1, 1 and the cores of the first bus are distributed to the cores of the two buses. At n_3, n_4 the bus of size 3 is split into two buses of sizes 1, 2. ■

Theorem 3.2. *The lower bound on test time of node n is lower or equal to the lower bound on test time of any node in the sub-tree with root n.*

Proof. It is a direct consequence of Theorem 3.1 where every parent node corresponds to CNF_a and every child node corresponds by construction to CNF_b. This is iteratively applied from n to the leaves of the sub-tree with root n. ▮

Theorem 3.2 permits the use of an effective branch and bound algorithm. Specifically, the nodes are visited beginning from the root toward the leaves and when the lower bound on test time of any node is higher than the best solution found so far, the whole sub-tree of this node is completely eliminated from the evaluation process. Note that neither this node nor any one of its successors can outperform the best configuration found so far. For example, the lower bounds LB_{n_2}, LB_{n_3}, LB_{n_4} of nodes n_2, n_3, n_4 respectively in Fig. 3.23 are higher or equal to LB_{n_1}. Therefore, if the best configuration found before the algorithm reaches n_1 has test time $\tau \leq LB_{n_1}$, then the sub-tree below n_1 is not further expanded. As we show in Section 4.3, the TDM test-scheduling approach gives results that are close to the lower bound, which makes this bound a strict and effective criterion.

3.5.4 Branch-&-Bound TAM optimization

The TAM configurations are examined by using a tree structure, where each node corresponds to a different configuration. The evaluation begins from the root node, where all N_c cores are connected on a single bus B of size $BS = N_{ATE}$ bit lines. The test time τ_{root} of the root node is calculated using the TDM-based test-scheduling approach described later in this section, and we set $\tau_{best} = \tau_{root}$. Then, the search tree is expanded to the second level, by splitting bus B into two buses, B_1 and B_2 with widths BS_1, BS_2 respectively ($BS_1 + BS_2 = BS$). For every combination of the values BS_1, BS_2 all possible assignments of the N_c cores to buses B_1 and B_2 are generated and each one is assigned to a different child of the root. The lower bound LB_n on the test time of each node n is calculated using Eq. (3.22) and it is compared against τ_{best}. If $LB_n \geq \tau_{best}$ node n is dropped and the tree is not expanded below n, else n is retained and its test time τ_n is computed using the TDM test-scheduling method. If $\tau_n < \tau_{best}$ then τ_{best} is set equal to τ_n and the configuration of node n becomes the best one.

The algorithm, as shown in Fig. 3.24 proceeds iteratively, starting from the leftmost non-rejected node of the second level, and continuously expanding the tree to the third level. Both buses of every node of the second level are selected, one at a time, and they are split into two buses each. New nodes are generated as successors of their parent nodes, and they constitute the third level of the tree. In a similar manner, the fourth, fifth, etc. levels of the tree are generated. The TAM configuration of every node at level L consists of L buses. The LB value of every new node is computed and if $LB \geq T_{best}$ it is rejected and its successors are not generated.

The B-&-B algorithm presented above assumes a predetermined number of ATE channels for testing each die. Even though this is the standard approach in single-site environments, in multi-site test environments the value of N_{ATE} used for testing each die has to be properly set according to the test time per die and the number N_{paral} of dies that can be tested in parallel. To this end, the B-&-B algorithm is repeated ms times, where the initial value of N_{ATE} is decreased in specific steps $s_1, s_2, \ldots s_{ms}$. The values of s_i are selected in such a way as to increase the number of sites N_{paral} (the released ATE channels are allocated to additional dies tested in parallel). The B-&-B algorithm is applied ms times on separate trees, where the root node of each tree corresponds to

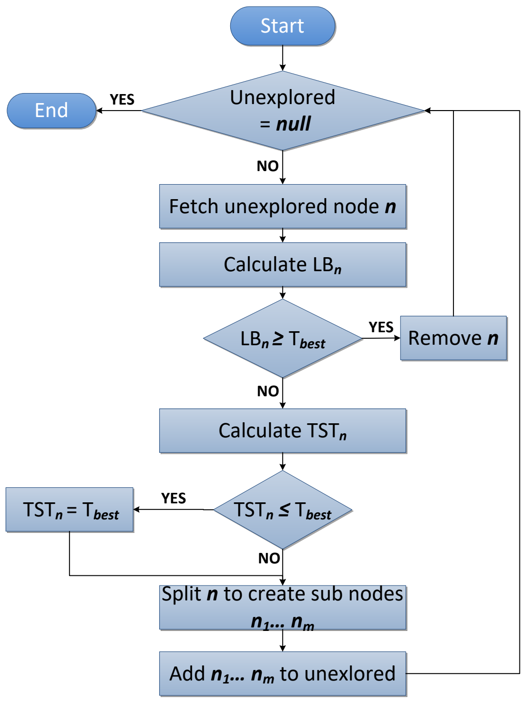


Figure 3.24: Flowchart of the Branch-&-Bound TAM optimization method.

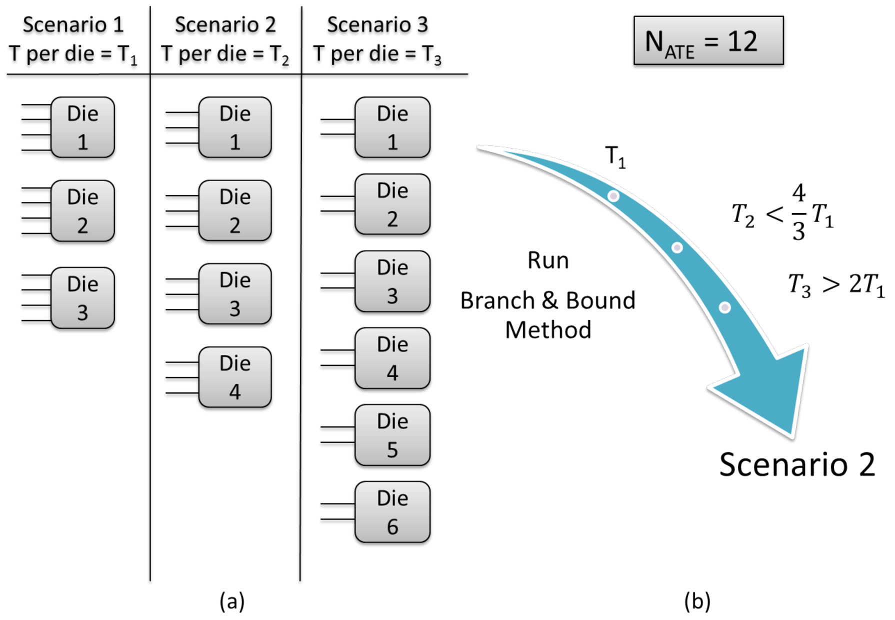


Figure 3.25: Branch-&-Bound TAM optimization method for multisite testing.

an one-bus configuration with $BS = N_{ATE} - s_i$ $(i = 1, 2, \ldots, ms)$. For every such tree the best TAM configuration is found, and among the best configurations found for all ms trees the configuration that minimizes the total time for testing all dies is selected.

The reduction of N_{ATE} by the value of s_i offers test time benefits through the increase of parallelism, but only when the time for testing a single die increases by a proportion that is less than s_i/N_{ATE}. Therefore, in the multi-site environment the B-&-B algorithm begins from the highest values of N_{ATE}, and proceeds by inheriting the best solution from the trees corresponding to the higher values of N_{ATE} to the trees corresponding to the lower values of N_{ATE}. Every node is either rejected or not based on the comparison against the best solution increased by the value of s_i/N_{ATE} when it comes from the previous tree. As a result, more nodes are rejected compared to the independent execution of the B-&-B algorithm on ms trees in the single-site case.

Example 9. Let us consider a tester with 12 test data channels and a multisite capacity of 6 dies. In addition, let us assume a batch of 60 dies that should be tested with minimal cost, i.e. the test time should be minimum. We explore three scenarios, to test in parallel 3, 4 and 6 dies, as shown in Fig. 3.25a. In each scenario, the available test channels per die are 4, 3 and 2 correspondingly. In the first scenario due to the large number of the available channels, the test time per die T_1 is minimum. However, the degree of parallelism is minimum too. On the other hand, in the third scenario the test time per die T_3 is the maximum, but the degree of parallelism is also the maximum. The test time for the whole batch per scenario is $20xT_1$, $15xT_2$ and $10xT_3$ respectively. Let the second scenario be the winning scenario after the execution of the B-&-B algorithm, as shown in 3.25b. In this case $20xT_1 > 15xT_2$ or $T_2 < (4/3) \cdot T_1$ or $T_2 < T_1 + (1/3) \cdot T_1$. Therefore the B-&-B algorithm found that testing the die with $N_{ATE} = 3$ instead of 4 ATE lines, thus $s_i = 1$, increases the test time T_2 by a proportion that is less than $s_i/N_{ATE} = 1/3$ of T_1. Furthermore, since $10xT_3 > 15xT_2$ or $T_3 > T_2 + (1/2) \cdot T_2$, the B-&-B algorithm failed to find a solution that increases the test time T_3 by a proportion that is less than $s_i/N_{ATE} = 1/2$ of T_2. ■

The TDM-based test schedule method applied to evaluate all non-rejected nodes is a simple Greedy Rectangle Packing (GRP) heuristic. According to GRP, a pool of candidate tests is created that consists of all possible tests per core, voltage and frequency. Each bus is represented by one bin, and each test is represented as a rectangle with width equal to the shift frequency used and height equal to the length of the test for the corresponding frequency and bus size (section 3.3.7). Each rectangle and each placement of that rectangle in the corresponding virtual bin is evaluated and the rectangle that can be placed to the lower and leftmost available position of the bin is selected and placed at that position. Every alternative test for the selected core and voltage (i.e., with different shift frequency) is removed from the pool and the same process is repeated for the tests remaining in the pool.

Even though GRP is inferior to SA-RP, they are both equally effective in evaluating different TAM configurations. As we show in Section 5.1, given any two configurations

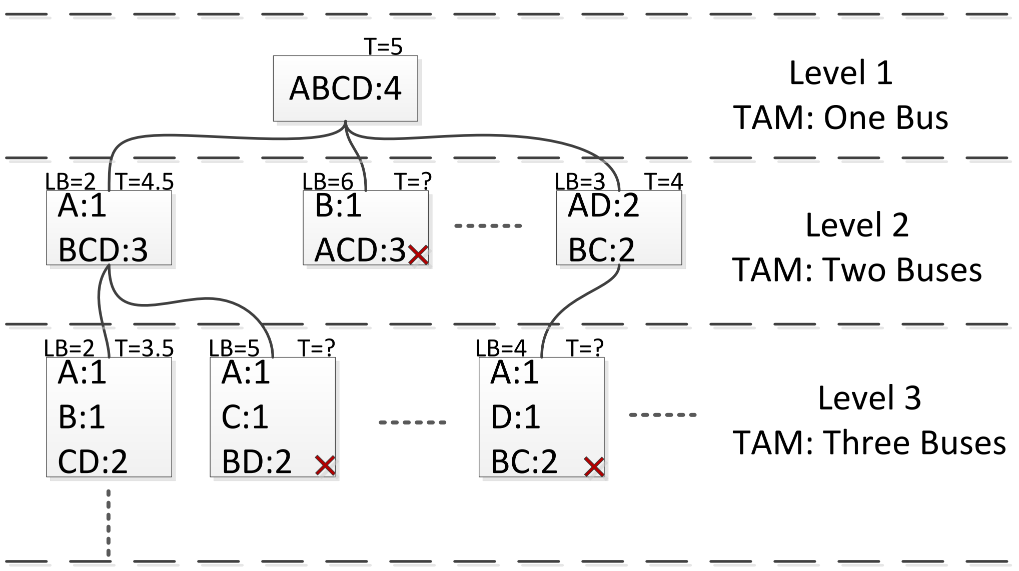


Figure 3.26: Example search tree

CNF_1, CNF_2, SA-RP provides better test schedules than GRP for both of them. However, if the test schedule provided by SA-RP for CNF_1 is better than CNF_2, then the test schedule provided by GRP for CNF_1 is also better than that of CNF_2 (this is confirmed in Section 5.3). Since GRP is much faster than SA-RP, it becomes evident that GRP is more suitable to evaluate the TAM configurations where it has to be applied multiple times. On the other hand, SA-RP is more suitable for generating the final test schedule, after the best configuration has been determined.

Example 10. Let us consider again the multi-V_{dd} SoC shown in Fig. 3.3 where the TAM structure is under optimization this time. Fig. 3.26 shows a part of the search tree. The TAM configurations are reported as sets of cores connected to the buses followed by the size of each bus. At the root node all cores are connected on one test bus with $BS = 4$ ($N_{ATE} = 4$). The test time returned by GRP for this configuration is equal to $T = 5$, and thus $T_{best} = 5$. There are 14 ways to distribute the cores between two buses B_1, B_2 with sizes $(BS_1, BS_2) = (1, 3)$, and another 7 ways to distribute the cores between B_1, B_2 when their sizes are equal to $(2, 2)$ respectively. As a result, 21 new nodes are inserted as direct successors of the root. The leftmost child of the root represents the TAM configuration with $BS_1 = 1$, $BS_2 = 3$. The core A is connected to B_1 and the cores B, C, D are connected to B_2. The lower bound (LB) on test time of every node is reported above the respective node. All the nodes with $LB \geq T_{best}$ (i.e., $LB \geq 5$) are immediately rejected (the X denotes that a node is rejected). Let us assume that only the leftmost and the rightmost nodes of level 2 are not rejected. For these nodes, GRP returns $T = 4.5$ and $T = 4$ respectively. Then $T_{best} = 4$. The tree is further expanded into level 3, by starting only from these two nodes. Since now $T_{best} = 4$, all nodes with $LB \geq 4$ are rejected. The algorithm finishes at the fourth level where each core is assigned a single bus (not shown in Fig. 3.26). ■

Let node n of the tree correspond to a TAM configuration with b buses $B_1, B_2, \ldots, B_b$ of sizes $BS_1, BS_2, \ldots, BS_b$ respectively. The immediate successors of this node correspond to all possible configurations with $b+1$ buses, where the $b-1$ buses among $B_1, B_2, \ldots, B_b$ remain unaffected, and one bus, let it be B_i, is split into two buses B_{i1}, B_{i2}. Let N_i be the number of cores connected to bus B_i. Since the width of bus B_i is equal to BS_i, the width of bus B_{i1} can take any value in the range $1 \ldots BS_i - 1$, and the width of B_{i2} is set equal to $BS_{i2} = BS_i - BS_{i1}$. However, due to the symmetry of the pairs (BS_{i1}, BS_{i2}) only half of them are considered, e.g. the pair $(BS_{i1}, BS_{i2}) = (k, BS_i - k)$ is equivalent to the symmetrical pair $(BS_{i1}, BS_{i2}) = (BS_i - k, k)$. Therefore, the number of different combinations for the different sizes for B_{i1}, B_{i2} is equal to

$$U(B_i) = \frac{BS_i - (BS_i \mod 2)}{2} \tag{3.33}$$

For each of these $U(B_i)$ combinations, there are $Q(B_i)$ different partitions of the N_i cores connected to B_i into two sets connected to B_{i1}, B_{i2}. This number is given by the Stirling number of the second kind formula [228]:

$$Q(B_i) = \frac{1}{2!} \sum_{j=0}^{2} (-1)^j \binom{2}{j} (2-j)^{N_i} \tag{3.34}$$

The total number of combinations of splitting bus B_i into two buses is equal to $Q(B_i) \times U(B_i)$. The total number of immediate successors of any node n is equal to

$$T(n) = \sum_{k=1}^{b} Q(B_k) \times U(B_k) \tag{3.35}$$

As expected, the search tree can become large for realistic problem instances. However, as shown in Section 5.3, more than 99% of the nodes are rapidly eliminated by the bounding criterion, while the number of the remaining nodes is small and manageable. Moreover, as shown in section 3.3, TDM is very effective with a small number of heavily loaded TAM resources, and thus many highly efficient configurations are found at the early stages of the search process at nodes near the root. As a result, many of the sub-trees are pruned by the bounding criterion, and the tree is not expanded towards deep levels. In addition the B-&-B algorithm can be easily implemented as a multi-threaded application, which can further reduce the run time. For very large SoCs where the computational time is still very high, we propose in Section 3.5.4 a distribution scheme, which reduces considerably the computational cost of the TAM optimization process.

3.5.5 Global TAM Distribution Scheme

For very large SoCs that consist of many different cores (identical cores do not constitute a challenge for TDM as they can be concurrently tested using the same TAM resources) the complexity of the problem becomes very large. In these cases it is difficult to enumerate

	$T_{A_i}(N_{A_i})$					L_T		
N_{A_i}	A_1	A_2	A_3	A_4	A_5	A_i	N_{A_i}	T_{A_i}
6	6737	9961	7434	1825	2109	A_2	6	9961
5	7186	9976	7472	1946	2250	A_2	5	9976
4	7200	13219	8962	2085	2410	A_3	2	10172
3	7200	13607	9087	2246	2596	A_3	1	11061
2	7510	15554	10172	2433	2812	A_2	4	13219
1	8261	15854	11061	2654	3068	A_2	3	13607
-	-	-	-	-	-	A_2	2	15554
-	-	-	-	-	-	A_2	1	15854

Table 3.5: Example Distribution Table.

and evaluate every different TAM configuration. However, most of the TAM configurations can be easily ruled out by area constraints. For example, even if the solution with minimum test time requires two cores at two opposite corners of the die to be connected using the same bus, the routing overhead of such a solution will most probably prevent the designer from adopting it.

One more practical design approach is to partition the floorplan of the SoC into w adjacent areas $A_1, A_2, \ldots, A_w$ based on design constraints, and generate a separate TAM structure for each one of them. Besides the low routing overhead, this approach has also the benefit of low TAM optimization complexity, because a) of the smaller problem instance that needs to be solved for each area, and b) the parallel nature of the problem where each area can be optimized independently of the other areas. The only question in this case is how to distribute the N_{ATE} available tester channels among the w areas in order to minimize the test time for the whole SoC.

Provided that the minimum time required for testing every area A_i for every possible number of TAM lines allocated to this area is known, there is an optimal method to distribute the N_{ATE} channels among the w areas of the SoC. Let N_{A_i} be the number of TAM lines of area A_i, and $T_{A_i}(N_{A_i})$ be the test time for area A_i when N_{A_i} TAM lines are used to test this area. Each area A_i is assigned a dedicated TAM structure with N_{A_i} TAM lines ($N_{ATE} = N_{A_1} + N_{A_2} + \cdots + N_{A_w}$) and thus it can be tested independently and in parallel with the others. Therefore, the minimum test time of the SoC equals the maximum test time required for any of the w areas, i.e. $T_{SoC}^{min} = \max\{T_{A_1}(N_{A_1}), T_{A_2}(N_{A_2}), \ldots, T_{A_w}(N_{A_w})\}$. The value of N_{A_i} has to be in the range $[1, N_{ATE} - w + 1]$ because each area A_i must be assigned at least one TAM line. The test times $T_{A_i}(N_{A_i})$ for all values of N_{A_i} constitute the distribution table of the SoC. The test time decreases as the number of TAM lines increases from 1 to $N_{ATE} - w + 1$, therefore the shortest test time required for A_i is equal to $T_{A_i}(N_{ATE} - w + 1)$ and the longest test time is equal to $T_{A_i}(1)$.

The target of the optimization process is to distribute the TAM lines to the areas $A_1, \ldots, A_n$ (i.e., to select the value N_{A_i} for each area A_i), such that the overall time T_{SoC}^{min} is minimized. Initially we calculate $T_{A_i}(1), T_{A_i}(2), \ldots, T_{A_i}(N_{ATE} - w + 1)$ for every $i \in [1, w]$.

The time for testing the SoC is lower bounded by the test time $T_{A_s}(N_{ATE}-w+1)$, which corresponds to the most time-consuming area A_s, i.e.,

$$T_{A_s}(N_{ATE}-w+1) = \max\{T_{A_1}(N_{ATE}-w+1),\\ T_{A_2}(N_{ATE}-w+1),\ldots,T_{A_w}(N_{ATE}-w+1)\}$$

Next, we create a list L_T of all possible SoC times starting from $T_{A_s}(N_{ATE}-w+1)$. In L_T we include all triplets $(A_i, N_{A_i}, T_{A_i}(N_{A_i}))$ with $T_{A_i}(N_{A_i}) \geq T_{A_s}(N_{ATE}-w+1)$ in ascending order of the value $T_{A_i}(N_{A_i})$. Then we begin an iteration and at each step we select the next triplet in the list starting from the topmost one and moving to the bottom of the list. Let the first triplet be $(A_z, N_{A_z}, T_{A_z}(N_{A_z}))$. We assign N_{A_z} TAM lines to the area A_z and for every other area A_i with $i \neq z$ we select the minimum value of N_{A_i} which fulfills the condition $T_{A_i}(N_{A_i}) \leq T_{A_z}(N_{A_z})$ from the distribution table. If the total number of TAM lines required for all areas A_i (including A_z) is less than the value N_{ATE} then this distribution is valid and it is accepted, else the process continues to the next iteration with the next triplet in the list L_T. The computational effort of this method is $O(N_{ATE} \cdot w)$, which is considerably lower than the $O((N_{ATE})^w)$ required by exhaustive search methods.

Example 11. Let us consider a large SoC that is partitioned into five areas A_1 to A_5, and let $N_{ATE} = 10$ test channels that must be distributed into five separate TAM structures, TAM_1 to TAM_5. Each TAM structure can be assigned 1 up to 6 TAM lines. Table 3.5.5 is the distribution table and it shows the test times $T_{A_i}(N_{A_i})$ for each area A_i and every number of test channels N_{A_i}, sorted in descending order of test channels N_{A_i}. The minimum test time for the given SoC is equal to 9961, which is the minimum test time required for area A_2 when the maximum possible number of 6 TAM lines are used for this area. The corresponding triplet $(A_2, 6, 9961)$ is set at the top of the list L_T. The next triplets in the list include all test times that are higher than 9961, i.e., 9976, 10172 etc. In order to test the SoC in 9961 time units we need to set $N_{A_1} = N_{A_4} = N_{A_5} = 1$, $N_{A_2} = 6$ and $N_{A_3} = 3$, which requires 12 TAM lines and it violates the constraint of $N_{ATE} = 10$. The next triplet in the list $(A_2, 5, 9976)$ corresponds to test time equal to 9976. For such a test time we need to set $N_{A_1} = N_{A_4} = N_{A_5} = 1$, $N_{A_2} = 5$ and $N_{A_3} = 3$, which also violates the constraint $N_{ATE} = 10$. Finally, the next triplet in the list $(A_3, 2, 10172)$ corresponds to test time equal to 10172, which requires $N_{A_1} = N_{A_4} = N_{A_5} = 1$, $N_{A_2} = 5$ and $N_{A_3} = 2$. This distribution is selected as the best TAM distribution because it offers the minimum test time and it does not violate the constraint. ■

The aforementioned method optimally distributes the available TAM lines to the areas A_i in negligible time, provided that the optimal TAM configurations for every area A_i and every number of TAM lines are known. Therefore, the execution time of this optimization process depends mainly on the execution time of the B-&-B algorithm, which must be applied $(N_{ATE}-w+1) \times w$ times for calculating the values of $T_{A_i}(N_{A_i})$ for every $i \in [1, w]$ and every $N_{A_i} \in [1, N_{ATE}-w+1]$. Depending on the run time of the B-&-B algorithm for each area and each value of N_{A_i}, the execution time of this optimization process can be excessively long for large areas and wide ranges $[1, N_{ATE}-w+1]$. However, we have to note that for practical SoCs, it is not necessary to examine the whole range $[1, N_{ATE}-w+1]$ due to area constraints. For example, it is highly unlikely that one area will be assigned $N_{ATE}-w+1$ TAM lines, even if this solution provides the lowest test time, to avoid the routing over-congestion of the TAM lines in this area. In most of the cases only a sub-range of $[1, N_{ATE}-w+1]$ need to be examined depending on the routing constraints of each area A_i.

The execution time of the TAM distribution process can be considerably reduced by exploiting the observation that the vast majority of the distribution table entries are not required during the distribution process. Therefore, by using a dynamic process we develop only those parts of the distribution table that are necessary for the optimization process. Specifically, we initially distribute the TAM lines to the various areas using a fast approximate approach. This distribution is usually close to the optimal distribution of the SoC, and it reveals the part of the distribution table where the optimal distribution is included. Therefore, in order to reduce the computational overhead of the distribution process, the distribution table is dynamically developed around this initial distribution.

In the fast approximate approach used for the initial distribution the time consuming B-&-B algorithm is omitted during the calculation of the $T_{A_i}(N_{A_i})$ values. Instead, we apply the GRP method once for each area A_i and each value of N_{A_i} assuming that the TAM structure of this area consists of a single bus with N_{A_i} TAM lines. In other words, an initial approximate TAM distribution table is constructed very fast using values $T_{A_i}(N_{A_i})$ calculated only for the root node instead of expanding the whole tree structure of the B-&-B algorithm. Using this approximate TAM distribution table the TAM lines are initially distributed to the various SoC areas as described before, and the values $N_{A_i}^{init}$ are calculated.

At the next step the initial distribution table is discarded, and a new distribution table is constructed step by step, starting from the values N_{A_i} in the range $[N_{A_i}-Z, N_{A_i}+Z]$, where Z is usually a very small integer parameter ($Z = 2$ in our case). For the new distribution table the time-expensive but accurate B-&-B algorithm is used for calculating every value $T_{A_i}(N_{A_i})$. Then, the optimization process is applied using the distribution table constructed so far, and the TAM lines are partitioned into a new set of values $N_{A_i}^1$. If all the selected values $N_{A_i}^1$ are not at the boundaries of the respective ranges $[N_{A_i}-Z, N_{A_i}+Z]$, i.e. when ($N_{A_i}-Z < N_{A_i}^1 < N_{A_i}+Z$), then the new distribution $N_{A_1}^1, N_{A_2}^1, \ldots, N_{A_w}^1$ is the optimal distribution. If some of the selected values $N_{A_i}^1$ are exactly on the boundaries $[N_{A_i}-Z$ or $N_{A_i}+Z]$ then the boundaries for the respective areas A_i are further expanded to ensure that the optimal solution is not outside the selected ranges. Specifically, for all those values $N_{A_i}^1$ that are equal to the boundary $N_{A_i}-Z$ the respective range is expanded at this boundary with another Z lines, i.e. $[N_{A_i}-2Z, N_{A_i}+Z]$, while for values $N_{A_i}^1$ that are equal to the boundary $N_{A_i}+Z$ the respective range is becomes $[N_{A_i}-Z, N_{A_i}+2Z]$. The optimization process is repeated using the new expanded distribution table.

In the best case the time consuming B-&-B algorithm will be applied only $w \times Z$ times, while in the worst case it will be applied $w \times (N_{ATE} - w + 1)$ times. The effectiveness of this approach to reduce the execution time depends on how close the initial configuration is to the optimal TAM configuration. However, as shown in section 3.3, TDM is very effective for a small number of heavily loaded resources, therefore the root nodes used to construct the initial configuration provide always a good solution that is usually close to the optimal solution. In addition the root node inherits all the factors that characterize uniquely each area A_i, like for example, the test load of each area, the degree of independence among the applied tests in an SoC area and the impact of the test channel bandwidth upon the degree of parallelization succeeded in the test scheduling.

Experimental results reported in Section 5.3, show clearly that only a small portion of the full distribution table is calculated using this approach and thus the optimal configuration is generated in very short execution time. In addition, each value $T_{A_i}(N_{A_i})$ can be calculated independently of the other values, which makes this process inherently parallel that can run very fast on a multiprocessor machine. As shown in Section 5.3, the execution time required for calculating $T_{A_i}(N_{A_i})$ for one area A_i and a single value of N_{A_i} is in the range of a few seconds to a few hours on an i7 processor. Therefore, the total run time of this optimization method on a multiprocessor system for a large SoC is expected to be in the order of a few minutes to a few hours.

3.6 Critical Path - Oriented & Thermal Aware X-Filling for High Un-modeled Defect Coverage

3.6.1 Introduction

Excessive power consumption during test, increases the overall chip temperature, and in many cases, it creates localized overheating. This phenomenon is called hotspot and it causes permanent damage to silicon, reliability failures and eventually yield loss. Thermal aware testing resolves thermal−related issues by reducing the temperature at hotspots in an SoC. Various techniques produce thermal−safe test schedules (see section 2.2.3), other techniques reorder scan vectors in order to maximize the effectiveness of heat dissipation on hotspot [136], while other techniques balance the thermal distribution of the core using layout information and/or weighting mechanisms [161]. By reducing the maximum developed temperature at a hotspot during the application of a specific test, the excessive interconnection delays that may cause a good die to fail testing are reduced, and the targeted areas are protected from effects related to aging.

However, reduction of the excessive delays caused by increased temperatures, is not always directly related to hotspot temperature minimization. Critical paths are not always affected by the hotspots, and thus excessive delays on such paths may remain even after the temperature at the hotspot is minimized. In some cases, the reduction of the temperature at the hotspot may even increase the temperature at other areas of the SoC. In such cases the temperature of critical paths increases, and the probability good dies to fail test increases too. Therefore, methods that are both thermal-aware and delay-aware are needed to avoid unnecessary yield loss.

Since thermal generation is proportional to the average power dissipation, most of the thermal-aware methods focus on the reduction of the power dissipation during testing. To this end, many techniques reduce the switching activity of the core during scan-in/scan-out and/or capture operation [206, 210, 211, 212, 213, 214]. X−fill methods are very effective in reducing shift and/or capture power [205, 206, 216, 217, 218, 208, 214, 215] by manipulating only the unspecified values of test cubes. X-fill methods have negligible impact on the ATPG process, they affect neither the scan chain structure nor the DUT, and they can be easily combined with other test methods. A popular X-fill method for reducing shift power is Fill-Adjacent (FA) technique [205]. Every two complementary consecutive test bits loaded into a scan chain generate switching activity as they travel along the scan chain. The FA technique minimizes the shift power by exploiting the X-bits of the test vectors in order to minimize the volume of the consecutive complementary test bits loaded into the scan chains as well as the distance they travel along the scan chains. For instance, consider a DUT with n scan chains, and assume that a test vector $S_j = XXX1XXX01XX0XXX1$ has to be loaded into scan chain $j(1 < j < n)$ from right to left. By applying FA to fill the Xs, we can get the test vector $T_j = 1111000010001111$. Table 3.6 shows all possible X-Fillings cases produced by the FA technique. The first column shows all possible blocks of test bits comprising any test vector that consists of n ($n \geq 1$) unspecified logic values bounded at the left and/or right by specified logic values. The second column shows the X-filling produced for all these blocks. This technique targets only the scan-in portion of the shift power, but it also reduces the scan-out power, because the scan-in power is highly correlated to the scan-out power [206]. In addition, it can be easily combined with capture-power reduction techniques such as the Preferred-Fill (PF) technique [207][208], to provide an efficient unified power-reduction solution.

Case	**Test vector**	**FA**
i	$0X\cdots X0, 0X\cdots X, X\cdots 0X$	$00\cdots 00$
ii	$1X\cdots X1, 1X\cdots X, X\cdots 1X$	$11\cdots 11$
iii	$0XX\cdots X1$	$011\cdots 11$
iv	$1XX\cdots X0$	$100\cdots 00$

Table 3.6: Fill-Adjacent X-Filling (the rightmost bit is loaded first into the scan chain).

The PF technique is an X-fill technique for reducing the switching activity during capture [133][205]. Let us consider a two-pattern Launch-On-Capture (LOC) test (V_1, V_2) where $V1 = (v_{11}, v_{12}, \cdots v_{1l})$ is the first l-bit vector applied on the DUT and $V2 = (v_{21}, v_{22}, \cdots v_{2l})$ is the response of V_1 which is applied as the second test vector to the DUT. If the logic value of V_1 corresponding to cell i, (i.e., v_{1i}) is unspecified then it should be filled with value $1(0)$ provided that the probability of v_{2i} (i.e., the logic value of

V_2 corresponding to the scan cell i) taking the value 1(0) is higher than taking the value 0(1). In other words, the v_{1i} bit is filled with a value that is more likely to be held after the capture in the i^{th} scan cell.

Since the thermal behaviour of a core depends on both the heat generation and the heat dissipation mechanism, layout information is needed in order to tackle the combined problem of hotspot-temperature and critical-path delay minimization. However, existing X−fill methods are totally unaware of the core layout, and inevitably they reduce the power in the whole area of the core in order to ensure that the temperature of the critical path during testing is also reduced. As a result they consume all 'X' values for reducing power while they neglect other test quality objectives, like the un−modeled defect coverage. For example, the FA technique, which minimizes the number of transitions at the scan chain during the scan−in process, offers very low un−modeled defect coverage [219]. Un-modeled defect coverage is adversely affected by this technique, because the generated test vectors are highly correlated (the major part of each of these vectors consists of runs of 0s and 1s). The method proposed in [219] offers a trade−off between power consumption and un−modeled defect coverage, but it does not consider local thermal and delay effects of the tests.

We propose a thermal and delay aware X−fill method that offers high un−modeled defect coverage. The proposed technique uses layout information to identify the scan cells that are most important for removing hotspots at critical nets, and it creates a thermal safe neighborhood around these nets. The rest of the scan cells are exploited to increase the un−modeled defect coverage of the generated test vectors by the means of an output-deviation based metric [209]. The proposed method reduces the delays at critical paths during testing and avoids thus the unnecessary yield loss, while it considerably increases the quality of the generated test vectors in terms of un-modeled defect coverage.

3.6.2 Motivation

Excessive delays at critical paths cannot be avoided if the critical paths are overheated during testing. In order to avoid overheating of critical paths, the power dissipated at the critical areas, i.e., the areas around the critical paths, must be minimized. At the same time, proper testing conditions must be imposed to ensure that these areas are also protected from heat transferred from other areas of the chip. A major factor affecting the heat-transfer mechanism between two areas is the distance between them. If two areas are geometrically close to each other, then a large thermal gradient can cause significant heat transfer between them and quickly change their temperatures. On the other hand, a large distance between them renders the heat transfer ineffective. Therefore, in order to protect a critical area, a thermal-safe zone must be created around it, that is able to absorb heat from this area and other neighbouring areas that develop high temperatures.

The first step towards this goal is to identify critical paths geometrically, and specify, in a topological manner, thermal-safe zones that surround each path. Then, the heat generated, or equivalently the power dissipated by the circuitry inside these zones must

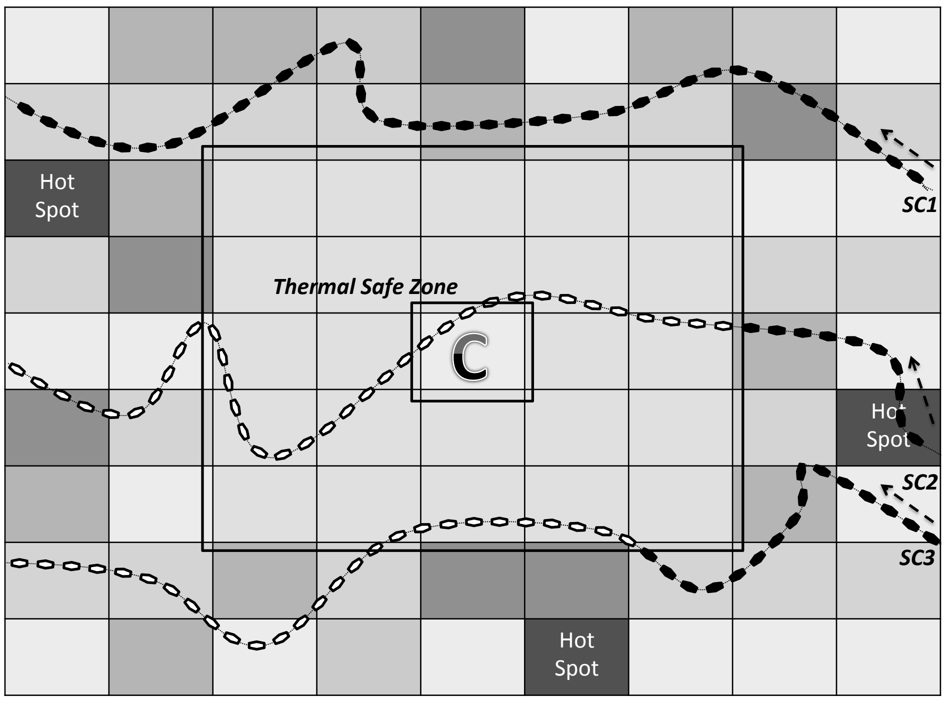


Figure 3.27: Critical Area & Thermal-Safe Zone.

be minimized. Since the largest portion of the power dissipated during testing is dissipated during the scan-in process, the number of transitions occurring inside the thermal-safe zones during scan-in must be minimized. Thermal-safe zones restrict the power dissipation internally, but they permit higher power dissipation externally at the non-critical areas. Therefore, thermal-safe zones reduce considerably the number of scan- cells exploited to reduce the thermal activity of the core, and the remaining scan-cells can be used to increase the quality of the test vectors.

In this work, we propose a novel algorithm that uses topological information to guide the X-fill process. Unspecified bits of scan-cells that affect thermal-safe zones are filled in such a way as to minimize the transitions at the thermal-safe zones during the scan-in process, while the rest of the scan-cells are exploited to maximize the un-modeled defect coverage of the generated test vectors. The un-modeled defect coverage is evaluated using output deviations, which provide an efficient probabilistic means to evaluate test vectors based on their potential for detecting arbitrary defects and, most importantly, without being biased towards any particular fault model [229, 230].

Example 1. Let us consider the CUT shown in Fig. 3.27, which is partitioned into 9×9 equally sized blocks, and let C be the critical block (i.e., the block with the critical paths). Test data are loaded using three scan chains, SC_1, SC_2 and SC_3 from the right to the left (as pointed by the arrows). The color of each block is properly set to reflect the thermal profile of the block during testing (dark colors indicate high temperature). The thermal-safe zone consists of two layers of blocks that surround block C. The objective of the proposed method is to properly fill the unspecified bits of scan-cells in such a way as a) to minimize the heat generation in block C, b) to enable heat transfer from block C to the thermal-safe zone, and c) to create a barrier between block C and the hot spots around the thermal-safe zone. To this end, the scan-cells are divided into two categories: a) the white scan-cells that have their test data pass through the zone during the scan-in process, and depending on their relative position at the scan chain they are filled either using power-oriented or defect-oriented criteria, and, b) the black scan-cells that do not affect the thermal-safe zones and they are filled using solely defect-oriented criteria. ■

3.6.3 Proposed Method

The proposed method consists of two flows, as shown in Fig. 3.28: (a) the area-characterization flow and (b) the X-fill flow. The objective of the area-characterization flow is to identify the critical blocks of the core, and create a thermal-safe zone around each one of them. To this end, the core's floorplan is first partitioned into a given number N of rectangle blocks. Each block occupies a certain area on the die enclosed by the coordinates of the upper left and lower right corner of its rectangle. Then the critical paths are identified using post$-$layout timing analysis and the blocks are separated into critical and non$-$critical blocks depending on whether they contain critical paths or not. Critical blocks are also referred to as blocks of zone Z_0 hereafter.

Every non-critical block that is an immediate neighbour of a critical block is included in the first layer of the thermal-safe zone, namely the Z_1 zone. Every non-critical block that is an immediate neighbour of a block in the Z_1 zone is included in the second layer of the thermal-safe zone, namely the Z_2 zone. Every non-critical block that is included in the Z_1 zone of one critical block, is not included in the Z_2 zone of any other critical block. The thermal activity is minimized at zone Z_0, it is permitted to be a little higher at Z_1 and even higher at Z_2. Outside these zones the thermal activity is left completely unconstrained. Each one of the blocks in zones Z_0, Z_1, Z_2 is assigned a different thermal weight $w_{Z_0} > w_{Z_1} > w_{Z_2}$ in the range [0, 1], which indicates the importance of thermal activity at this block (the higher is the weight, the more strict is the constraint on the thermal activity of the block). The non-critical blocks have no constraints on thermal activity, therefore they are assigned zero weights.

The thermal weight of each block is used to assign a thermal weight at every scan-cell SC_j^i located inside that block. We note that scan-cell SC_j^i, belongs to scan chain $i \in [1 \dots S]$ and occupies position $j \in [1 \dots L]$, where $j = 1$ corresponds to the scan-out cell, and $j = L$ corresponds to the scan-in cell. During the loading of scan chains, test data of scan-cell SC_1^i are shifted first into scan chain i while the test data of scan-cell SC_L^i are the last shifted into scan chain i. Therefore, scan-cell SC_j^i is considered as successor of SC_{j+1}^i and predecessor of SC_{j-1}^i. The thermal weight of each scan-cell SC_j^i depends on the block it belongs to, as well as on its relative position j in the scan chain i. The position j of SC_j^i in the scan chain i is used to measure the impact of SC_j^i on

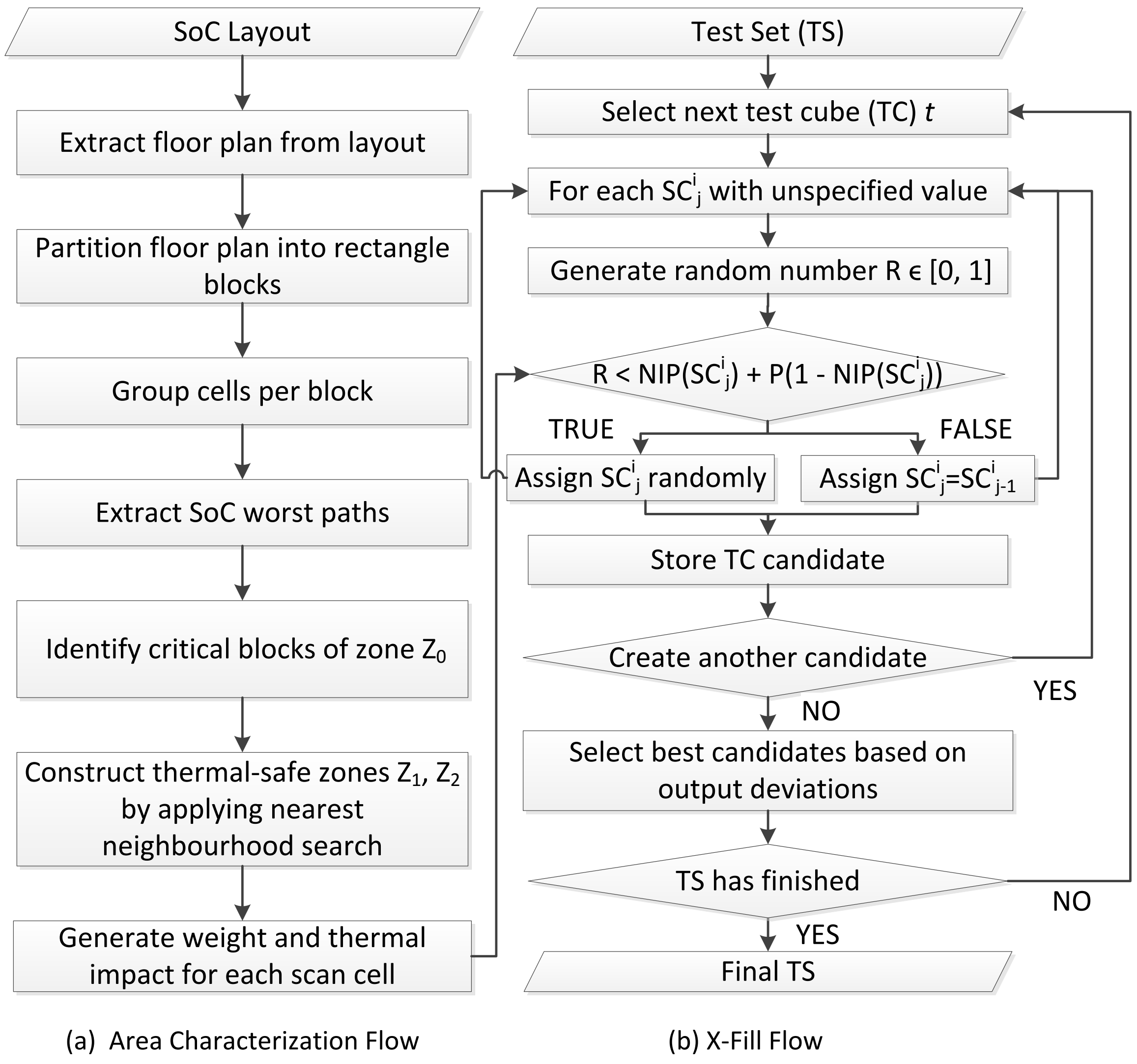

SoC Layout
Extract floor plan from layout
Partition floor plan into rectangle blocks
Group cells per block
Extract SoC worst paths
Identify critical blocks of zone Z_0
Construct thermal-safe zones Z_1, Z_2 by applying nearest neighbourhood search
Generate weight and thermal impact for each scan cell
(a) Area Characterization Flow
Test Set (TS)
Select next test cube (TC) *t*
For each SC^i_j with unspecified value
Generate random number R ϵ [0, 1]
$R < NIP(SC^i_j) + P(1 - NIP(SC^i_j))$
TRUE
FALSE
Assign SC^i_j randomly
Assign $SC^i_j=SC^i_{j-1}$
Store TC candidate
Create another candidate
YES
NO
Select best candidates based on output deviations
TS has finished
NO
YES
Final TS
(b) X-Fill Flow

Figure 3.28: Flow of Proposed Method.

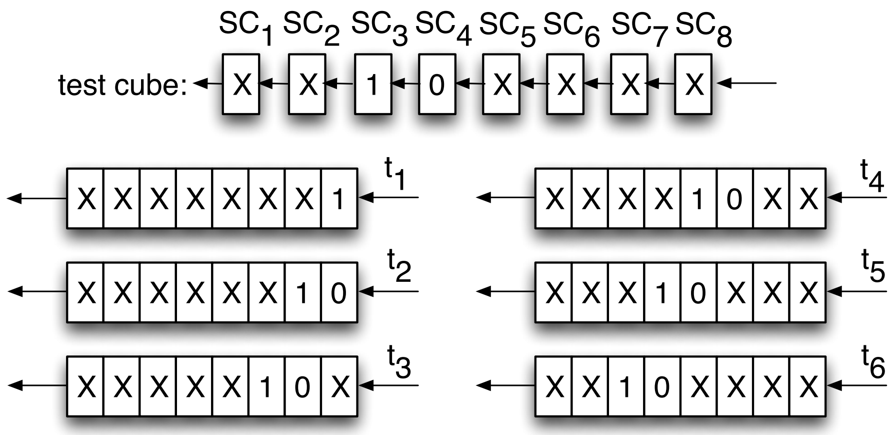


Figure 3.29: Scan-In Transitions.

the overall number of transitions caused by SC_j^i during the scan-in operation. Scan cells that are located closer to the scan output (i.e. those with low values of j) receive higher weights than those located closer to scan inputs (i.e., those with high values of j). This is done because the test data of scan-cells located close to the outputs need to travel long distances in the scan chains to reach the scan-cells, and thus they affect the heat generation at the core during testing more than the scan-cells located close to the scan inputs. Therefore, the thermal weight of scan-cell SC_j^i that belongs to thermal zone Z_k is calculated as $W(SC_j^i) = w_{Z_k} \times \frac{1}{j}$ for $k = 0, 1, 2$.

After the scan-cells are assigned weights, the unspecified bits of the test cubes (i.e., test vectors with 'X' values) are filled in such a way as to reduce the number of transitions inside the safe-zones. We note that every pair of complementary test bits at two successive scan-cells causes transitions at all the predecessor scan-cells during the scan-in process. This is shown in Fig. 3.29, where the complementary test bits at scan-cells SC_3 and SC_4 cause transitions at scan-cells $SC_3 - SC_8$ during scan chain loading. Therefore, even when a scan-cell SC_j^i is located inside a non-critical block, it impacts the thermal activity of other critical blocks or blocks of thermal-safe zones, when they contain scan-cells that precede SC_j^i in scan chain i.

The impact IP of scan-cell SC_j^i to the power dissipation of the die can be quantified by summing the weights of SC_j^i and its predecessor scan-cells SC_{j+1}^i, SC_{j+2}^i, … using the following formula:

$$IP(SC_j^i) = \sum_{m=j \ldots L} W(SC_m^i) \tag{3.36}$$

A large value of $IP(SC_j^i)$ provides an indication that scan-cell SC_j^i should be set to the same logic value with scan-cell SC_{j-1}^i, else a logic transition will be introduced to the scan-in process, with a large impact on the thermal activity of the critical areas of the core. The value of $IP(SC_j^i)$ is used to calculate the normalized impact of scan-cell SC_j^i, $NIP(SC_j^i)$, which is a value in the range $[0, 1]$, according to the formula

$$NIP(SC_j^i) = \frac{IP(SC_j^i)}{IP_{max}} \tag{3.37}$$

where $IP_{max} = max\{IP(SC_j^i)\}, \forall i, j$. $NIP(SC_j^i)$ is used to fill the unspecified bits of each test cube in a probabilistic manner, starting from the scan-out cells SC_1^i and moving towards the scan-in cells SC_L^i. Specifically, let R be a random number generated in the range $[0, 1]$. If $R \leq NIP(SC_j^i)$ the unspecified value of scan-cell SC_j^i is set equal to the specified value of scan-cell SC_{j-1}^i, else it is set randomly to either logic value 0 or 1. In the case that SC_j^i does not affect much the thermal-safe zones the value of $NIP(SC_j^i)$ is very low, and the probability that SC_j^i is set randomly is very high. However, when SC_j^i affects thermal-safe zones then the value of $NIP(SC_j^i)$ is high and there is a high probability that it is assigned the same logic value with its neighbour scan-cell SC_{j-1}^i.

Even though this formula achieves the thermal objectives set earlier, there are many cases that further bias towards more power friendly test vectors is required. To this end, a parameter P is introduced by the designer in the range $[0, 1]$, and the formula becomes

$$R \leq NIP(SC_j^i) + P(1 - NIP(SC_j^i)) \tag{3.38}$$

As the value of P increases from 0 to 1, the right part of (3.38) increases from $NIP(SC_j^i)$ towards 1, and thus the probability that this formula is true increases. As a result, more scan-cells are assigned the same logic values with their neighbour cells, and the power dissipation of the test vectors decreases even more.

Large values of P tend to generate test vectors with very low power dissipation. However, the test vectors are highly correlated due to the biased filling of their unspecified bits, and thus the un-modeled defect coverage drops. Low values of P generate test vectors with many unspecified bits filled randomly, which increases the un-modeled defect coverage. Moreover, due to the probabilistic nature of the proposed method, a number of test vectors can be generated for each test cube, by repeatedly applying this X-fill technique on every test cube. Even though all the test vectors generated for one test cube are equally effective in term of power dissipation at the critical blocks and the thermal-safe zones for a given value of P, they offer different un-modeled defect coverage. Therefore, in order to select the best one among them, they are evaluated using the output-deviation metric proposed in [209]. Output deviations are probability measures at primary outputs and pseudo-outputs that indicate the likelihood of error detection. Specifically, a probability map (referred to as confidence-level vector) is assigned to every logic gate in the circuit, and signal probabilities are assigned at each internal line of the circuit. The output deviation for input pattern tp and an output/pseudo-output w is defined as the probability output w to receive the opposite than the error-free logic value. Each test vector is applied with two capture cycles r_1 and r_2 (i.e., we assume the Launch-On-Capture technique as it is common in industry). For each output w, the generated test vectors are partitioned into four groups: those producing fault-free response 0 and 1 at capture cycles r_1 and r_2. The output-deviation values of all generated test vectors are calculated and the largest value for every output w and for each fault-free response v at capture cycle r_k are used to evaluate the test vectors. The selection process ensures that one test vector is selected for every test cube, and the final set of selected test vectors maximizes the output deviation of every output. The X-fill flow is shown in Fig. 3.28(b).

Chapter 4

Simulation Framework

4.1 Introduction

The simulation framework provides with an integrated IC design and test environment. A comprehensive set of script- and program- based flows for commercial and custom-made tools enable the efficient and reliable deployment and execution of experiments upon the test methods discussed in chapter 3. The simulation framework consists of four main flows, as shown in Fig. 4.1,

- the design flow for benchmark cores, that processes ISCAS and IWLS [232] cores and produce the required core- design and test data for the needs of our experiments,
- the SoC design flow, that allows for the creation of artificial SoC and TAM configurations, which can be used to evaluate the proposed test methods,
- the execution flow, that includes implementations of the proposed test methods and allows for experimentation upon artificial and industrial SoCs,
- the presentation flow, that stores and present in an efficient and user friendly way the results of the experiments for debugging and evaluation purposes.

A short description of these flows is given in the next subsections.

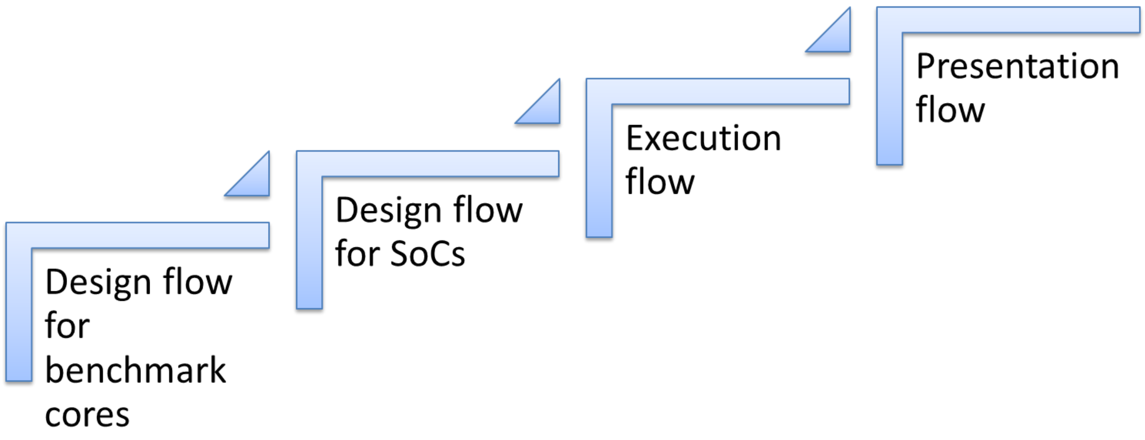


Figure 4.1: Flows in technological framework.

4.2 Design flow for benchmark cores

A design flow, in the domain of the VLSI IC, is a set of procedures that allows designers to progress from chip specification to chip implementation. A general design flow is shown in Fig. 4.2. Design starts at the behavioral level and then proceeds to the structural level. This step is called Register Transfer Level (RTL) synthesis. In RTL level the designs are captured in a Hardware Description Language (HDL). The HDL description is then transformed to a physical description suitable for chip fabrication. This step is called physical synthesis or layout generation. In Fig. 4.2, the design has been partitioned into the front end stage at the RTL level and the back end at the structural and physical levels. This is important because it illustrates a partitioning that is used to build ASIC [9]. In this case, the design can be developed at the HDL level in a fabless, hardware-design-oriented enterprise (or department of an enterprise) and then passed to a manufacturing enterprise (or department) that implements the actual chip. The basic design flow, shown in Fig. 4.2, applies to SoC design too, in the sense that the entire system needs to be specified, debugged, modified for testability, validated, and mapped to a technology, but in this case the whole flow needs to be done in an integrated framework.

The design flow for benchmark cores constitutes a subset of the general design flow. In the case of a benchmark core, the behavioural level has already implemented, tested and verified. Thus, the front-end flow starts at the point of RTL synthesis, where specialized tools are used to directly transform the behavioural RTL description to a structural gate-level net-list. The tools that have been used in this research for RTL synthesis were provided from Synopsys and Cadence Design Systems via the Europractice software service [114].

The imlemented RTL flow is shown in Fig. 4.3. As it is illustrated, using an RTL synthesizer such as the Design Compiler from Synopsys or the RTL Compiler from Cadence, the verified RTL description of an ISCAS or IWLS core is transformed initially

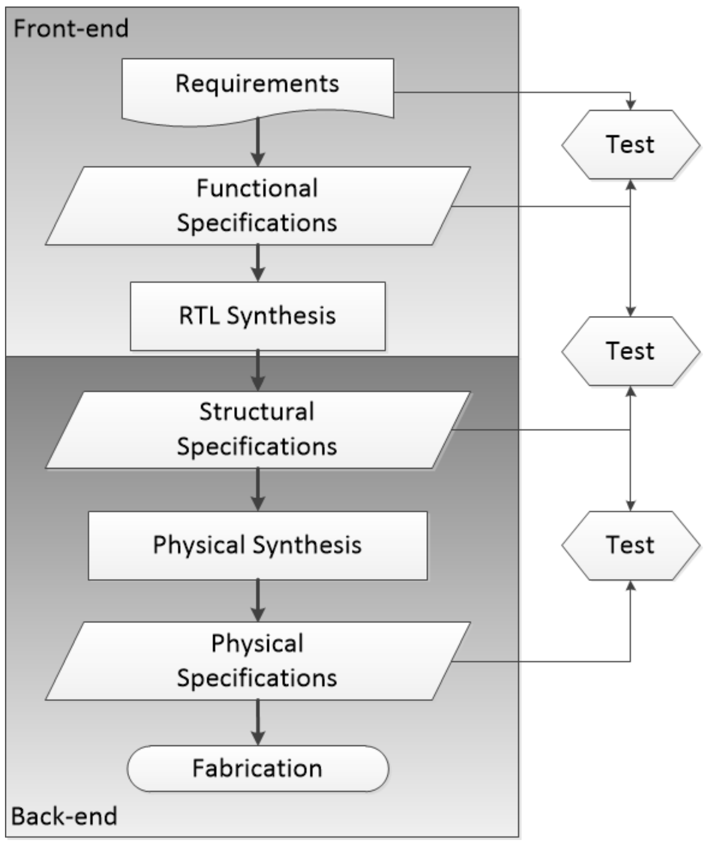


Figure 4.2: General Design Flow.

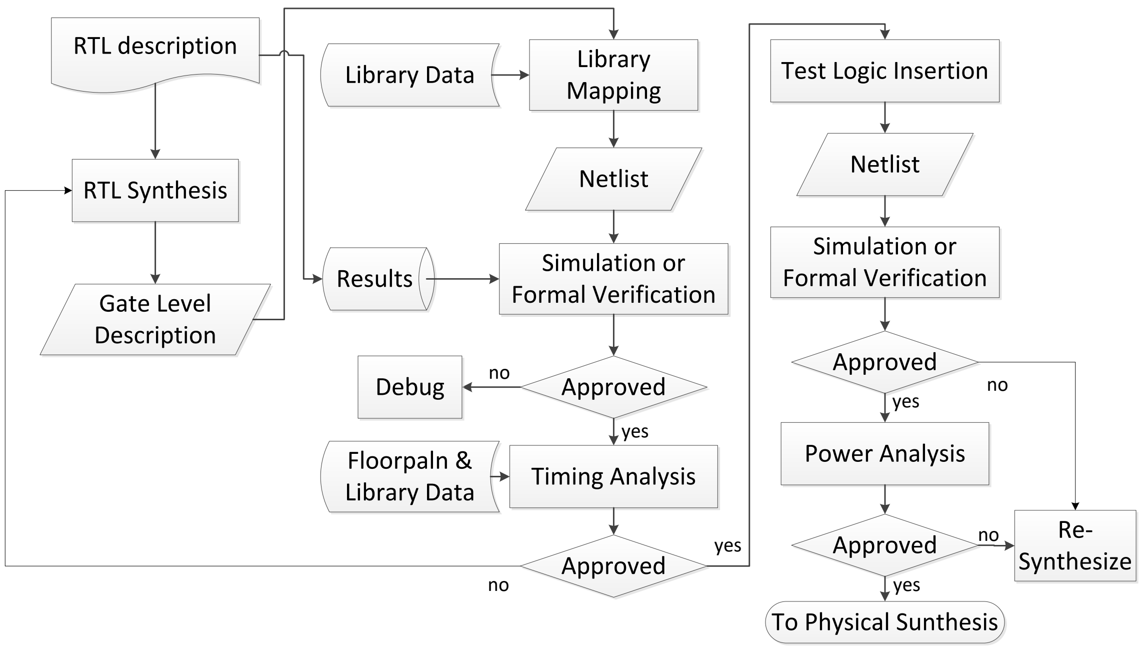


Figure 4.3: Example RTL flow.

to a generic circuit of gates and registers, optimized in terms of speed and area. Then, the generic gates are mapped one-to-one to pre-designed cells of a standard cell library. In this research, the 45nm process nangate standard cell library [115] is used. The final result is a structural net-list of the benchmark core.

There are two main approaches to verify that the structural net-list performs the same function as the verified HDL product, the functional and the formal verification. In the first approach, a logic test bench is used to verify that exactly the same output is produced for the behavioural and structural descriptions. In the second approach, it is created a formal verification program that compares the logical equivalence of the two descriptions. Formal verification mathematically proves that both descriptions have exactly the same Boolean functions [116], [117]. In contrast, functional verification is based on the efficiency of the test vectors. Formality from Synopsys and Incisive Conformal from Cadence are examples of formal verifiers. Other types of verification that can be run are semantic and structural checks on the HDL. For example, in a semantic check, it should be ensured that all bus assignments match in bit width, while in a structural check, it should be checked that all outputs are connected.

The next step in the flow is a static timing analysis that evaluates all timing paths in the core under scope. The inputs to the timing analyser are derived from the basic timing of the library gates, due to intrinsic gate delays, and routing loads. The routing loads are estimated statistically or they are derived from floor-planning data. In this research, the Encounter from Cadence has been used for timing analysis. The final result is a report that includes timing information for the worst paths in the core.

At this point, to allow for efficient test, the DFT is implemented using specialized procedures either in the Synopsys Design Compiler or the Cadence RTL Compiler. Specifically, scannable registers are inserted and / or existed registers are modified so that the state of the design can be set and monitored. Then, ATPG is performed to generate tests for the scannable design. In this research, the required core test sets are generated using the Synopsys Tetramax and the Cadence ET ATPG tools.

The RTL flow concludes with the estimation of the design's normal and test power consumption. Commercial power analysis tools that have been used are the PrimePower from Synopsys and the EPS from Cadence.

Layout generation is the process of turning a design into a manufacturable database. Namely, it transforms a design from the structural to the physical domain. This process is sometimes called physical synthesis. Fig. 4.4 shows a standard place and route layout generation design flow used in most ASICs and in this research too. The commercial tool that has been used in this research for the layout generation of the ISCAS and IWLS cores is the Encounter from Cadence.

The layout flow starts with the structural net-list describing gates, flip-flops, and their interconnections. The net-list is provided in the Design Exchange Format (DEF) as a Verilog netlist. Then, a semi-automated floor-planning step is performed. The result of this procedure is a file that describes the floor-plan of the design. The next step is the

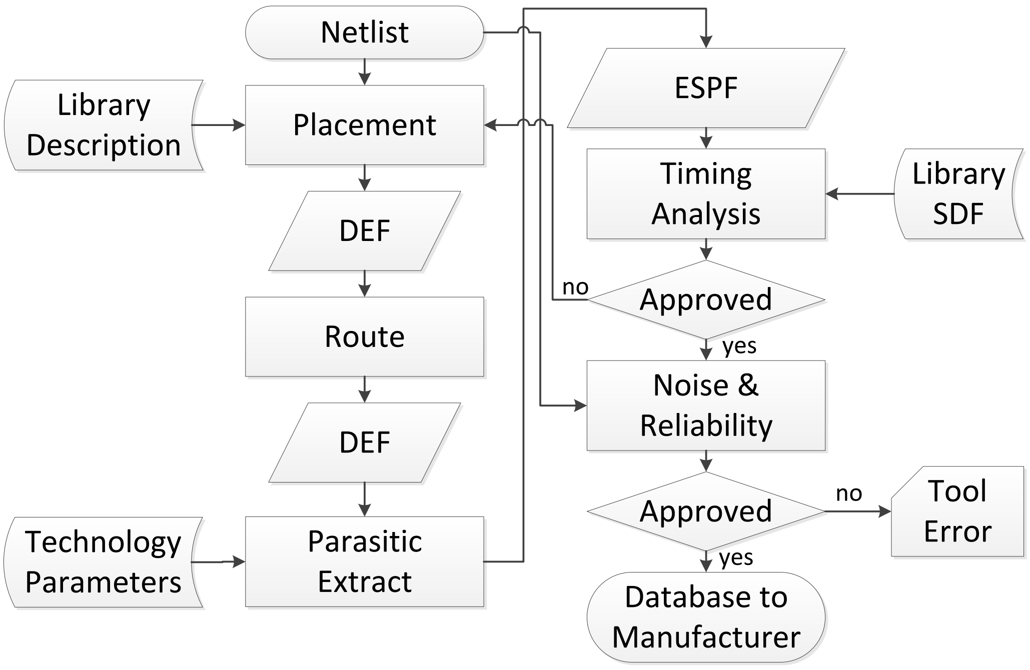


Figure 4.4: Automated layout generation.

placement process, where standard cells of constant-height and variable-width are arrayed in rows across a chip, as shown in Fig. 4.5. A standard cell library definition describing cell dimensions and port locations, in the Library Exchange Format (LEF), summarizes the salient physical details of cells. A simple placement algorithm is used to minimize the length of wires. At the end of the placement phase, the used cells have been fixed in position in the overall array. The placed design is saved in a standard format (e.g., DEF) for routing.

After placement of standard cells, the signal nets in the design need to be routed. Routing is divided into two phases: global routing and detailed routing. A global router creates routes through channels according to a cost function. Wires can be changed from channel to channel if the density of wires in a channel becomes too high. The detailed router places the actual geometry required to complete signal connections. The results are written to another DEF file. The picture of a design after routing is shown in Fig. 4.6.

The placed and routed design is then passed to the circuit parasitic extractor. The placed and routed design is provided to the extractor in DEF format and the output is a file in the Standard Parasitic Exchange Format (SPEF) that describes the R's and C's associated with all nets in the layout. The extractor uses another technology file defining the interlayer capacitances and layer resistances.

Static timing analysis is rerun with the actual routing loads placed on the gates. Multiple iterations of synthesis and placement/routing are usually necessary to converge

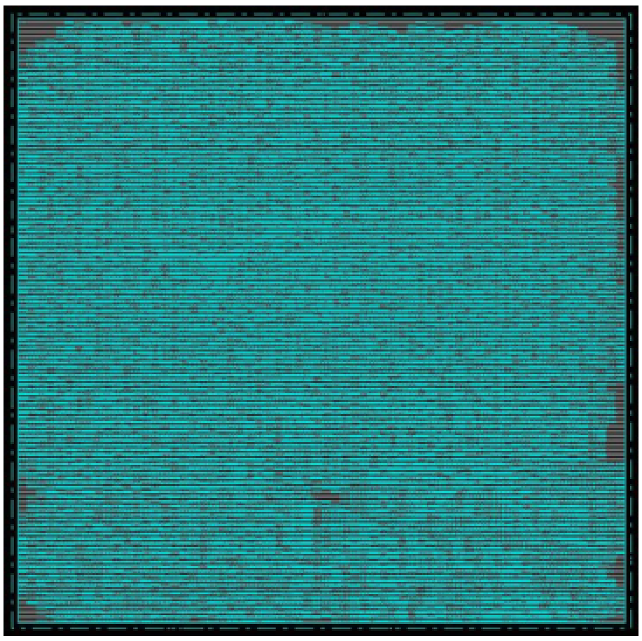


Figure 4.5: Floor-plan anf placement of standard cells (Cadence Encounter).

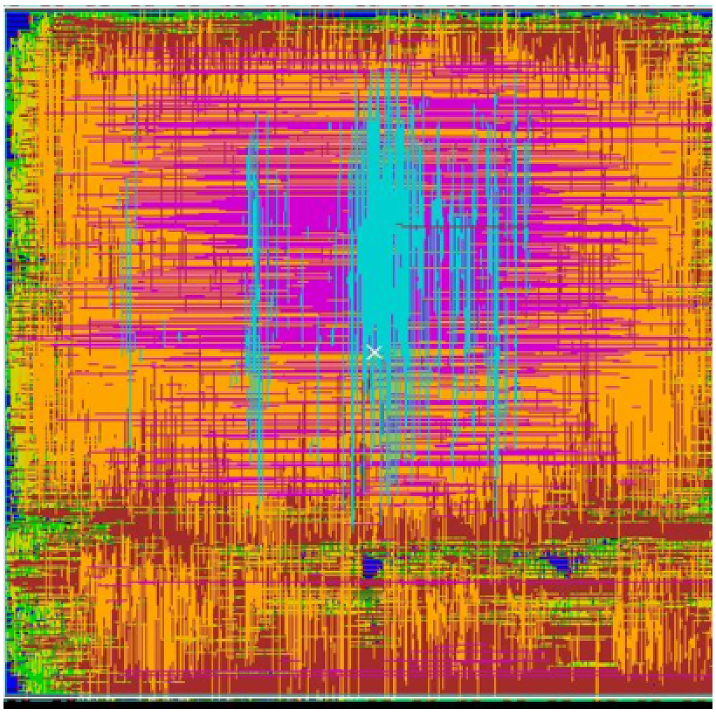


Figure 4.6: A routed design (Cadence Encounter).

on timing requirements. Central to modern high-speed designs is the clock distribution strategy. To minimize skew, the clock and its buffers are pre-route before the main logic placement and routing is completed. This task is performed with a clock tree router. Normal and test power estimation is repeated for the complete design. Similar tools and techniques to those in the RTL level are used. Finally, the thermal profile of a given benchmark core during test is estimated using the hotspot tool [204].

4.3 Design flow for SoCs

The design flow for SoCs consists of an industrial-SoC- and an artificial-SoC- oriented process. In the first process, the SoC design comes directly from an industry source. Specifically, an SoC configuration file is provided that includes

- the voltage islands and the cores in the SoC,
- the voltage settings that are used in each island,
- the maximum scan frequency that is allowed in each voltage setting per island,
- the test time of each core per voltage setting when the maximum allowed scan frequency is used,
- the power consumption of each core in each voltage setting when the maximum allowed scan frequency is used.

Next, the given SoC configuration file is further processed so that information related to the test environment and the test method can be incorporated. Specifically, this step defines

- the maximum ATE_clock frequency,
- the scan frequencies provided by the TDM scheme,
- the TDM-based TAM configuration, that is the number of the TAM buses, their bit-length and the set of the cores that are connected to each bus.
- Optionally, the WPP width of each wrapped core.

The above described data are included in two configuration files, an SoC test- and a TAM- configuration file, that constitute the input in the execution flow. Example SoC test- and a TAM- configuration files can be found in the appendix A. In the artificial-SoC-oriented process, a custom-made tool, designed by the author and called 'SoCTDM-Scheduler', is used to define the configuration file of an artificial SoC. A designer, using a either a graphical user interface or a descriptive text file (see appendix A), can

- select the embedded cores in an artificial SoC from a set of available benchmark cores and/or user-defined cores,
- define voltage islands and assign cores in each island. The core assignment can be implemented either manually or automatically. In the manual case, the designer predefines the cores per island. Then, the SoCTDMScheduler using an external tool, the 'hotfloorplanner' [204], derives the floor-plan of the artificial SoC taking into account proximity relations among the embedded cores that belong to the same voltage island. In the automated case, the floor-plan of the artificial SoC is derived first and then the embedded cores, according to their position in the floor-plan, are clustered in voltage islands,
- define the voltage settings per island from an available set. The voltage settings that can be used in each island are dependent on the cores of the island and the available data for these cores. For example, up to three predefined voltage settings can be supported in the case of an island that consists of ISCAS and IWLS cores.

When the designer concludes the aforementioned definitions, a similar to the industry-based SoC configuration file is produced automatically. Then, the designer can proceed in the definition of the SoC test and TAM configuration files as described above.

4.4 Execution flow

The execution flow is implemented in the custom-made tool SoCTDMScheduler and consists of a test method configuration process and an execution process. During the test method configuration process, a test designer can choose between

- a fixed TAM configuration or a TAM optimization process,
- the TDM and the STDM scheme,
- power-aware and power-unconstrained test scheduling,
- execution or not of thermal simulations for the derived test schedule.

In the case of a fixed TAM configuration, only the test scheduling method needs to be defined along with its parameters. A designer can select among the TDM-based GRD, the TDM-based SA-RP and the STDM-based SA-RP method. If TAM optimization is required, the test designer can select the number of the available ATE test channels and the maximum number of buses that should be used in the TAM. When the test method is fully defined the execution process can take place in order to derive the test schedule and, if it is required, the optimal TAM configuration for the SoC under test.

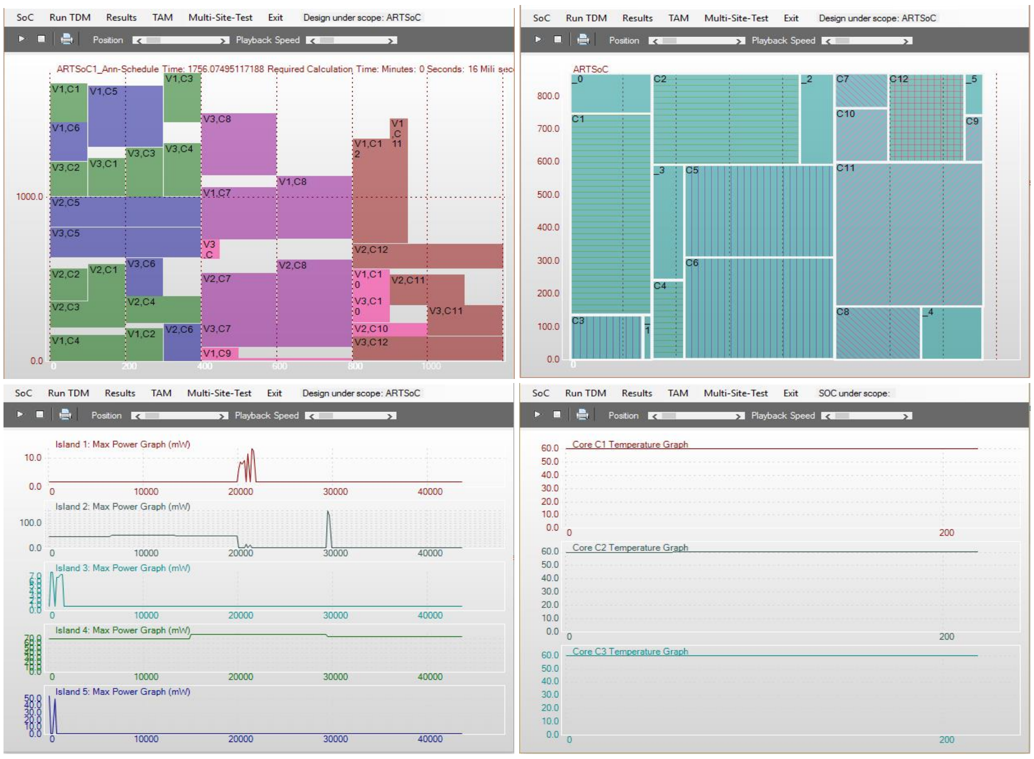

SoC
Run TDM
Results
TAM
Multi-Site-Test
Exit
Design under scope: ARTSoC
Position
Playback Speed
ARTSoC1_Ann-Schedule Time: 1756.07495117188 Required Calculation Time: Minutes: 0 Seconds: 16 Mili sec
ARTSoC
Island 1: Max Power Graph (mW)
Island 2: Max Power Graph (mW)
Island 3: Max Power Graph (mW)
Island 4: Max Power Graph (mW)
Island 5: Max Power Graph (mW)
Core C1 Temperature Graph
Core C2 Temperature Graph
Core C3 Temperature Graph
SOC under scope:

Figure 4.7: SoCTDMScheduler presentation interface.

4.5 Presentation flow

The presentation flow consists of a set of procedures that allows for user-friendly, graphical representation of the derived results. These results can be used for debugging or evaluation of the test methods. The SoCTDMScheduler, as shown in Fig. 4.7, can represent graphically

- SoC test schedules,
- SoC floor-plans,
- power graphs of SoCs and their embedded cores,
- temperature graphs of SoCs and their embedded cores.

Chapter 5

Evaluation of the research work

5.1 Evaluation of the TDM-based test methods

V_{dd}	I_1^A		I_2^A				I_3^A					I_4^A	
level	C_1^A	F^m	C_2^A	C_3^A	C_4^A	F^m	C_5^A	C_6^A	C_7^A	C_8^A	F^m	C_9^A	F^m
V_1	300	266	60	128	262	200	N/A	N/A	128	600	133	55	200
V_2	500	200	95	220	500	100	475	375	170	950	100	N/A	N/A
V_3	800	100	160	340	700	50	800	600	300	1600	50	N/A	N/A
V_4	1600	50	N/A	N/A	N/A	N/A	1600	1200	600	3200	25	N/A	N/A

Table 5.1: SoC-A Test Times (in Normalized Time Units) At Maximum Scan Frequencies F^m (In MHz) [231].

In this section, we present an evaluation of the proposed TDM-test-scheduling methods based upon experimental results that come from two industrial SoCs, hereafter referred to as SoC-A and SoC-B, respectively [231]. Both SoCs consist of digital cores and are targeted for portable wireless applications. They are extremely power conscious, and they have various programmable levels of power control, either through external power supply control or through internal settings. SoC-A has 4 voltage islands $I_1^A, I_2^A, I_3^A, I_4^A$, 9 cores $C_1^A, \ldots, C_9^A$ and 124 clock domains. SoC-B has 7 voltage islands $I_1^B, \ldots, I_7^B$, 15 cores $C_1^B, \ldots, C_{15}^B$ and 225 clock domains. SoC-A can be set to up to 4 voltage settings, while SoC-B can be set to up to 6 voltage settings. While the test environment for these SoCs has several more constraints than modelled in this section, e.g., in terms of available test pins, tester resources such as power supplies and clocks, compatibility amongst different test modes, sequencing amongst test modes, etc., these examples illustrate the richness that exists in industrial designs, the test of which can benefit from effective test-scheduling methods.

Table 5.1 presents the minimum testing times for all cores C_i^A of SoC-A at the various voltage levels V_j. Correspondingly, tables 5.2 and 5.2 present the minimum testing times for all cores C_i^B of SoC-B at the various voltage levels V_j. The test data for the cores are grouped into columns according to the island they belong to. The test times are calculated using the maximum scan frequencies, denoted as 'F^m', that do not cause scan chain timing violations at any core for shifting at the corresponding voltage level (they are presented in normalized time units, so as not to reveal confidential data). The maximum scan frequencies are presented in the last column of each island (they are reported in MHz and they are different for every voltage setting and every island). The entries denoted as 'N/A' correspond to cores that either do not operate at the respective voltage settings or they are not tested at these voltage settings.

V_{dd}	I_1^B			I_2^B				I_3^B			
level	C_1^B	C_2^B	F^m	C_3^B	C_4^B	C_5^B	F^m	C_6^B	C_7^B	C_8^B	F^m
V_1	900	300	400	700	100	550	200	N/A	188	600	266
V_2	1200	396	300	1400	200	1100	100	475	370	950	200
V_3	1350	450	266	2800	400	2200	50	700	500	1600	100
V_4	1800	600	200	N/A	N/A	N/A	N/A	1400	1000	3200	50
V_5	3600	N/A	100	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
V_6	7200	N/A	50	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 5.2: SoC-B Test Times (in Normalized Time Units) At Maximum Scan Frequencies F^m (in MHz) [231].

In the rest of this section, we present test-time results for SoC-A and SoC-B at all supported voltage levels. In particular, we report results for the following test-scheduling approaches:

1. Shortest-Job-First (SJF): This method is very effective for scheduling tests at single-V_{dd} designs [234]. We use it as a good representative of single-V_{dd} test scheduling approaches; we adapted SJF to multi-V_{dd} designs by appending the required constraints described in section 3.3.4.

V_{dd}	I_4^B			I_5^B			I_6^B			I_7^B	
level	C_9^B	C_{10}^B	F^m	C_{11}^B	C_{12}^B	F^m	C_{13}^B	C_{14}^B	F^m	C_{15}^B	F^m
V_1	700	N/A	200	N/A	165	300	500	125	200	1300	200
V_2	924	198	150	900	192	200	N/A	N/A	N/A	N/A	N/A
V_3	1050	225	133	N/A	250	150	N/A	N/A	N/A	N/A	N/A
V_4	1400	300	100	N/A	500	75	N/A	N/A	N/A	N/A	N/A
V_5	2800	N/A	50	N/A	1000	38	N/A	N/A	N/A	N/A	N/A
V_6	5600	N/A	25	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 5.3: SoC-B Test Times (in Normalized Time Units) At Maximum Scan Frequencies F^m (in MHz) [231].

2. Session-Based Scheduling (SBS): This is the first method proposed for multi-V_{dd} SoCs [88].

3. TDM-based Scheduling: This is the TDM scheduling method proposed in this work. It is implemented using three approaches: the ILP formulation presented in section 3.3.5 (TDM-ILP), the rectangle-packing/simulated annealing approach presented in section 3.3.6 (TDM-SA-RP), and the greedy approach presented in Section 3.3.7 (TDM-GRD). The TDM-ILP approach was implemented using FICO XPress-MP Solver [235]. Both TDM-SA-RP, TDM-GRD approaches were implemented using the SoCTDMScheduler tool.

The baseline SJF approach is adapted to the particular requirements of multi-V_{dd} designs. Recall that besides the additional constraints set by the use of multiple voltage settings, SJF is unaware of the different shift frequencies required at different voltage levels. As a result, SJF often schedules in parallel different tests at different voltage settings and different islands that require different shift frequencies, provided of course that no constraints are violated. However, even if constraints are not violated, testers in industrial multi-site testing environments usually provide a single clock for every SoC; hence it is likely that all scan partitions will have to be shifted at the same rate. If one shift frequency has to be used at a certain time, that frequency should be equal to the lower frequency that does not violate the scan-chain timing for any of the cores being tested at that time.

Since the exact number of shift frequencies that can be used concurrently for testing an SoC depends on the available tester channels and the available SoC pins, we consider two bounding scenarios for the SJF method. In the worst-case scenario (denoted as 'WC'), we assume that there is only one ATE channel for providing the clock signal to every core. In that case, the lowest frequency has to be used, which is equal to 25 MHz as it is reported in Tables 5.1, 5.2, 5.3. In the best-case scenario (denoted as 'BC'), we assume that for each core/island a separate ATE channel is available for providing a separate clock signal, and thus the highest possible scan frequency can be used for loading the test data at any voltage setting. Any single-V_{dd} method cannot achieve better test time than this scenario, therefore it corresponds to the best-case in terms of test time for single-V_{dd} methods.

The BC scenario is overly optimistic due to the very high cost associated with such an approach. In addition, usually only a very small set of shift frequencies are supported by the tester. Nevertheless, we consider it here to show the relative effectiveness of the proposed method against any other conventional single-V_{dd} technique that could potentially reach the BC scenario. The WC scenario is rather pessimistic, even though it is closer to the reality than the BC scenario. Recall that, as shown in Tables 5.1, 5.2, 5.3, the test times at the lower voltage settings dominate the time for testing the SoC. Therefore, it is very likely that the lower scan frequencies will have to be used most of the time in a pin/ATE-channel-limited environment. In the average case, any single-V_{dd} method is expected to achieve a test time that is between the best and the worst case.

For the TDM methods, we assume that the tester provides a clock signal with frequency that is close to the highest scan frequency of any core of the SoC (200 MHz for SoC-A and 400 MHz for SoC-B). For SoC-A, this frequency is divided on-chip using the proposed TDM scheme by 8, 4, 2 and 1, which corresponds to scan frequencies 25, 50, 100 and 200 MHz. For SoC-B, it is divided by 16, 8, 4, 2 and 1, and the available scan frequencies are 25, 50, 100, 200 and 400 MHz, respectively. Each core C_i is tested at a voltage V_j using any of the TDM frequencies that are smaller or equal to the corresponding highest nominal scan frequency of C_i at V_j. For example, as reported in Table 5.1, for testing core C_8^A at V_1, the highest scan frequency that can be used is 133 MHz. If we consider a TDM scheme that provides frequencies of 200, 100, 50 and 25 MHz, then only the frequencies 100, 50, and 25 MHz, can be used for testing C_8^A at V_1, and using equation (3.1), the respective test times are equal to 800, 1600 and 3200, respectively. Note that when the scan frequency is divided by two, the completion time for the task doubles. However, at the same time, the use of a smaller frequency permits a higher level of parallelization between the various tasks, which offsets the increase in test time.

In order to show the potential of TDM to decrease testing time for multi-core/multi-V_{dd} SoCs, we consider that SoC-A is tested using a TAM configuration that employs two test data buses, BUS-A and BUS-B. Initially, BUS-A transfers test data only to the wrapper of the core C_8^A and BUS-B loads the rest of the cores. Then, we gradually increase the test load of BUS-A by moving the cores of BUS-B to BUS-A, until all cores are connected to BUS-A and BUS-B is completely removed (all core wrappers have the same width and thus the size of BUS-A remains the same). The results are shown in Fig. 5.1. The x-axis shows the number of additional cores connected to BUS-A at each step. The y-axis shows the test time achieved by both TDM-ILP and the SBS test scheduling methods. The TDM-ILP method offers higher parallelism than the SBS method, and thus the total test time only slightly increases as the number of cores connected to BUS-A increases. On the other hand, the SBS method can only aggregate the additional test load to the existing load, thus, forcing the overall test time to increase accordingly. The ratio of the test time of SBS over TDM is reported above each pair of columns.

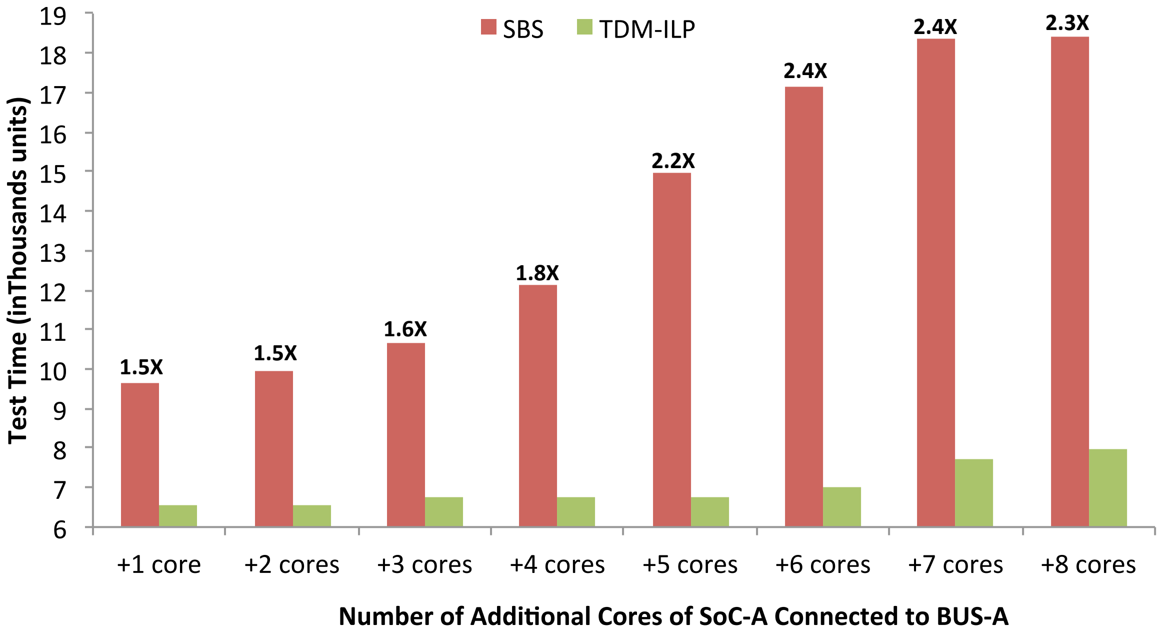


Figure 5.1: Test time obtained using TDM for various numbers of cores connected to a bus.

TDM uses less TAM resources and achieves, at the same time, better TAM utilization than the non-TDM methods. However, in order to keep the total test time low, there is an upper bound on the number of cores that can be connected to every bus. Due to the ability of TDM to schedule tests in parallel, this bound is much higher than the bound of a non-TDM scenario. Nevertheless, when this bound is reached, more buses can be added in the design, and the cores can be connected to these buses according to area constraints and performance optimization objectives. Even in that case, the number of buses is expected to be much smaller in the TDM approach than in any non-TDM approach.

In order to evaluate all the test-scheduling methods, we used various TAM configurations for the two industrial SoCs. We assumed a pre-determined TAM architecture in each case (the problem of TAM optimization is not yet considered). Specifically, to avoid any bias due to the assignment of cores to TAMs, for SoC-A (SoC-B), we considered three TAM configurations, using 2, 3 and 4 (3, 4 and 5) test data buses respectively, where we connected three different randomly selected sets of cores. The results for SoC-A and SoC-B are shown in tables 5.4 and 5.5 respectively. For both tables, the first row presents the name of each SoC, as well as the theoretical lower bound in test time calculated according to Equation (3.17). The first two columns of each part present the number of buses and the configuration index number while the next columns present the results for each considered method for both SoCs. The TDM-based test scheduling methods achieve remarkably high reduction in test time compared to non-TDM methods, especially when the tester has limited resources (low number of test data buses). The test time of the TDM - based methods is considerably lower, when the worst-case scenario is considered,

SoC-A ($T^{min}_{SoC-A} = 6540$)							
No of Buses	Cnf	SJF		SBS	TDM		
		WC	BC		ILP	SA-RP	GRD
2	1	37196	14055	12894	**6548**	6603	8177
	2	28060	12489	12200	**6548**	6767	7815
	3	36652	14489	13527	**6548**	6746	8737
3	1	22264	9540	7603	**6548**	6548	7782
	2	26752	11139	7624	**6548**	6548	9870
	3	28060	12350	12200	**6548**	6788	7815
4	1	24720	12598	6775	**6548**	6748	9648
	2	25380	12886	7867	**6548**	6548	6848
	3	24720	10740	6625	**6548**	6548	6848

Table 5.4: Test Time Comparisons (test times are shown in clock cycles).

and also much lower than the best-case scenario for most of the SJF test cases. Both SJF and SBS methods are competitive only in the best-case scenario, and only when there are enough resources to counterbalance the parallelization efficiency of TDM.

There are cases in tables 5.4 and 5.5 where the ILP method provided optimal results (boldfaced entries). In the rest of the cases, the solver was terminated after a certain time limit. In the particular case of SoC-B where the complexity of the ILP model is very high, there are cases that the SA-RP method performs better than the ILP. This is due to the running-time limit given to the ILP solver. Interestingly, even in the cases that ILP did not terminate (and thus provided sub-optimal results), both ILP and SA-RP methods achieved test times that are very close, if not equal, to the lower bound provided by Equation 3.17. The TDM-GRD approach offers less compelling test schedules, which are however superior to those given by the SBS and SJF methods in most cases.

Based on the test-time results, the ILP-TDM method is the most attractive choice when enough computational resources are available. Tables 5.6 and 5.7 present the complexity of the ILP models, and the computation times that are required for each method to derive the test schedules reported in tables 5.4 and 5.5 respectively. For both tables, the first column presents the number of buses and the configuration index as a pair (B,C). The next three columns present the complexity of the ILP model in terms of number of relations (Rel.), number of integer (Int.) and binary (Bin.) variables. The next three columns present the computation time (in seconds) for the ILP, the SA-RP, and the GRD method.

Fig. 5.2 reports the CPU times for all intermediate solutions generated by the ILP solver for SoC-B and the configuration (B,C) = (3,1). The dashed line shows the test time of the competing SBS approach. Very high test time improvements are achieved and the proposed method quickly outperforms the SBS approach. Further improvements are achieved when more CPU time is given to the ILP solver. The results for the rest of the

SoC-B ($T^{min}_{SoC-B} = 17095$)							
No of Buses	Cnf	SJF		SBS	TDM		
		WC	BC		ILP	SA-RP	GRD
3	1	169840	39939	38749	17095	17095	19420
	2	124093	25603	24275	17095	17095	17095
	3	134413	25848	25532	17910	17095	17760
4	1	128607	30477	30447	19537	18095	22480
	2	122032	20396	21321	17095	17095	17095
	3	123952	22846	22846	18594	17395	17993
5	1	108740	22246	23796	17671	17345	18215
	2	119013	22696	21075	17095	17095	17095
	3	133740	23396	23396	17993	17395	17993

Table 5.5: Test Time Comparisons (test times are shown in clock cycles).

configurations are similar.

The heuristic approaches, namely the TDM-SA-RP and the TDM-GRD, drastically reduce the required computation time. Specifically, the TDM-SA-RP method manages to produce much faster than TDM-ILP, very efficient test schedules that are close to the schedules produced by the TDM-ILP method. The TDM-GRD method has negligible computation time, but it delivers inferior results than the TDM-ILP and the TDM-SA-RP methods. Therefore, TDM-GRD can serve as a good estimation method (e.g. for cases that two different test architectures must be compared at an early design phase), or even as the final scheduling approach in environments with very limited CPU resources and/or very large SoCs under test. On the other hand, ILP is very advantageous for small to moderate SoCs, where it can offer optimal results, while it serves as a good heuristic-like approach for large SoCs.

Fig. 5.3 shows a typical run of the TDM-SA-RP and the TDM-GRD algorithm for SoC-B. The x-axis shows the CPU time of the TDM-SA-RP algorithm in milli-seconds, and the y-axis shows the test time in normalized time units. The TDM-GRD algorithm runs in 69 msec and provides the test time shown as a straight line in Fig. 5.3. The SA offers initially some fair solutions, which are close to the solution given by the TDM-GRD approach. However, this solution is gradually improved as the optimization proceeds. At a certain point after 22 seconds, the test time is considerably reduced and reaches its lowest value, which is much better than the solution given by TDM-GRD. Beyond that time, there is no further improvement and the SA-RP terminates. It is worth noting that for the same result the ILP solver had to run for one day.

Even though testing at multiple voltage levels is a very promising technique to reduce test escapes, the current practice of industry is to test at a selected subset of voltage levels. This subset may be limited to only two extreme voltage levels (the operating corners) to reduce the test cost. To show the benefits of the proposed TDM method in

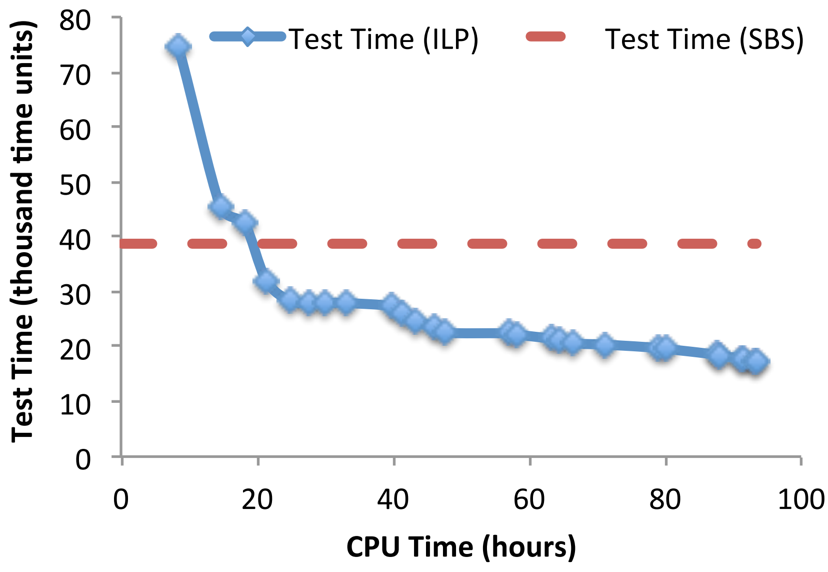


Figure 5.2: Test and CPU times for ILP.

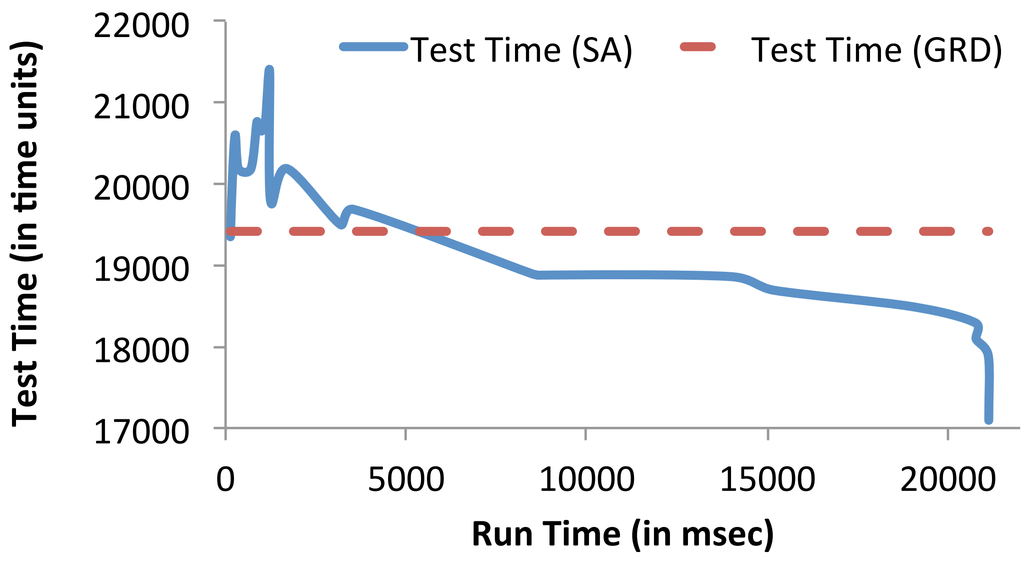


Figure 5.3: Results obtained using SA-RP and comparison with the GRD method.

SoC-A						
(B,C)	ILP				SA-RP	GRD
	Rel.	Int.	Bin.	CPU Time (s).	CPU Time (s)	CPU Time (s)
(2,1)	23190	1969	8754	24624	15.26	0.08
(2,2)	18454	1969	6450	8856	15.69	0.1
(2,3)	25742	1969	9986	30762	18.07	0.13
(3,1)	10601	1969	2665	13788	18.84	0.07
(3,2)	13456	1969	4046	22284	22.18	0.09
(3,3)	14874	1969	4722	22932	18.66	0.1
(4,1)	8404	1969	1601	26856	29.31	0.03
(4,2)	9502	1969	2138	5328	27.54	0.04
(4,3)	8872	1969	1835	1512	26.23	0.06

Table 5.6: Model complexity for ILP method and CPU times for the various optimization methods.

this case, we consider SoC-B with one bus, and we apply the SBS, the TDM-GRD and the TDM-SA-RP methods. The test time in each case is equal to 33401, 17600 and 9694 time units respectively. Therefore, even in this case, the benefits of TDM remain high.

In order to show the scalability of the SA-RP and GRD approaches, we conducted an experiment with a hypothetical large many-core SoC that consists of many identical copies of SoC-A and SoC-B. In total, the large SoC consists of 7 islands and the number of voltage settings was set equal to 6. The number of buses was set equal to 10 in all cases. Table 5.8 presents the results. The first column presents the number of the cores and the next two pairs of columns present the test time and the running time for the TDM-SA-RP and the TDM-GRD approaches (the ILP approach is not scalable to SoCs of that size due to computational complexity and thus it is excluded from this experiment). TDM-SA-RP offers much better results than TDM-GRD, but requires considerably more CPU time. Some non-monotonicity in the CPU time comparison can be attributed to the heuristic nature of both approaches. However, it is obvious that the CPU times for both methods are realistic, even for the large SoCs with 500 cores.

In the next experiment, we examine the effect of power constraints on the test time of TDM. Table 5.9 shows for SoC-A the power consumption for every core at every voltage setting when the highest shift frequency supported by this voltage setting is used (the maximum shift frequencies are reported in table 5.1. Correspondingly, tables 5.10 and 5.11 shows in similar way the power consumption of SoC-B. The power consumption for lower frequencies can be calculated for each core using equation (3.3). The power limit for each island was set equal to 120 units for I_4^A, 400 units for I_1^A, I_2^A, 750 units for I_3^B, I_4^B, 800 units for I_3^A, 1000 units for I_2^B, I_5^B, I_6^B, 2000 units for I_1^B and 300 units for I_7^B. Table 5.12 presents the power-aware results. As expected, the power consumption limits affect the test times generated by the proposed methods. However, TDM adjusts the shift frequency

SoC-B						
(B,C)	ILP				SA-RP	GRD
	Rel.	Int.	Bin.	CPU Time (s).	CPU Time (s)	CPU Time (s)
(3,1)	89199	4101	38570	336240	71.57	0.07
(3,2)	123042	4101	55111	421920	73.96	0.15
(3,3)	102673	4101	45169	227880	145.462	0.24
(4,1)	46572	4101	17795	436320	220.71	0.1
(4,2)	32129	4101	10756	479160	67.1	0.14
(4,3)	29937	4101	9677	258840	139.56	0.19
(5,1)	24473	4101	7036	297360	216.03	0.1
(5,2)	20683	4101	5183	266400	65.89	0.17
(5,3)	22511	4101	6059	245520	132.572	0.23

Table 5.7: Model complexity for ILP method and CPU times for the various optimization methods.

Cores	Test Time		Run Time	
	SA-RP	GRD	SA-RP	GRD
100	21195	31681	5h	2 sec
200	28601	44127	60h	11 sec
300	33846	47289	55h	37 sec
400	40308	95989	71h	316 sec
500	50066	74728	79h	1024 sec

Table 5.8: Test time and CPU time for many-core SoCs.

(and thus the power consumption) as necessary, and exploits parallelism to minimize the overall test time. This is shown in Table 5.12, where in many cases, the proposed TDM methods manage to compensate effectively the power constraint and derive test schedules that are close or equal to efficiency to test schedules that are derived without taking into account power limitations. Note that in many cases, TDM has no other option than to apply the test using shift frequencies that are lower than the maximum ones, because the maximum shift frequencies violate the power constraints of the respective islands.

In order to show how the power limits affect test time, we gradually increased the power limits reported above for each island in steps of 10%. Using each new set of power limits, we run the TDM-SA-RP algorithm for SoC-B, using the configuration $(B, C) = (3, 1)$. The increase of the test time at each step is shown in Fig. 5.4. The x-axis shows the percentage increase of the power limit for each step, and the y-axis shows the corresponding test-time increase. As shown by the curve's slope, the TDM-SA-RP method can reschedule effectively the tests for SoC-B even if we reduce by half the initial power limits per island. Any further decrease of these limits causes a higher increase to

V_{dd}	C_1^A	C_2^A	C_3^A	C_4^A	C_5^A	C_6^A	C_7^A	C_8^A	C_9^A
V_1	798	120	256	524	N/A	N/A	170	800	110
V_2	486	49	104	212	475	375	104	583	N/A
V_3	197	20	42	86	192	152	42	425	N/A
V_4	80	N/A	N/A	N/A	78	62	17	310	N/A

Table 5.9: SoC-A Power Consumption (in Normalized Time Units) [231].

V_{dd}	C_1^B	C_2^B	C_3^B	C_4^B	C_5^B	C_6^B	C_7^B	C_8^B
V_1	3600	1200	1400	200	1100	N/A	188	600
V_2	2437	812	632	94	496	950	128	407
V_3	1950	650	285	41	224	429	58	184
V_4	1323	441	N/A	N/A	N/A	193	26	83
V_5	597	N/A	N/A	N/A	N/A	N/A	N/A	N/A
V_6	269	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 5.10: SoC-B Power Consumption (in Normalized Time Units) [231].

the test schedule time, as it is expected.

5.2 Evaluation of the STDM-based test methods

In this section, we present an evaluation of the proposed STDM-test-scheduling methods based upon experimental results that come from the industrial SoC SoC-B, described in section 5.1. We assume that the tester provides a clock signal with frequency equal to the highest shift frequency reported in tables 5.2 and 5.3, i.e., 400 MHz. The TDM scheme supports frequencies 25, 50, 100, 200 and 400 MHz. For every core-voltage pair, every frequency in this set that is lower or equal to the corresponding F^m reported in tables 5.2 and 5.3 can be used. For example, for C_1 at V_3 with $F^m = 266$ MHz only 200 MHz, 100 MHz, 50 MHz and 25 MHz are supported and the respective test times are equal to 1795, 3591, 7182 and 14364.

Throughout this section, we consider WPP widths that are up to 4x smaller than the size of the bus. Since STDM reduces the frequency on the bus to compensate for wrappers that are narrower than the bus, the set of frequencies supported by STDM is extended to include also the values of 12.5 MHz and 6.25 MHz, which are 2x and 4x smaller than the lowest frequency supported by TDM. Finally, based on the observations in section 3.4.2, we used the single bus configuration, and we varied the size of the bus in the range $\{L/4, L/2, L\}$.

In the first experiment, we compare STDM against the TDM and non-TDM approaches. We consider first the case that wrappers are flexible and thus their parallel

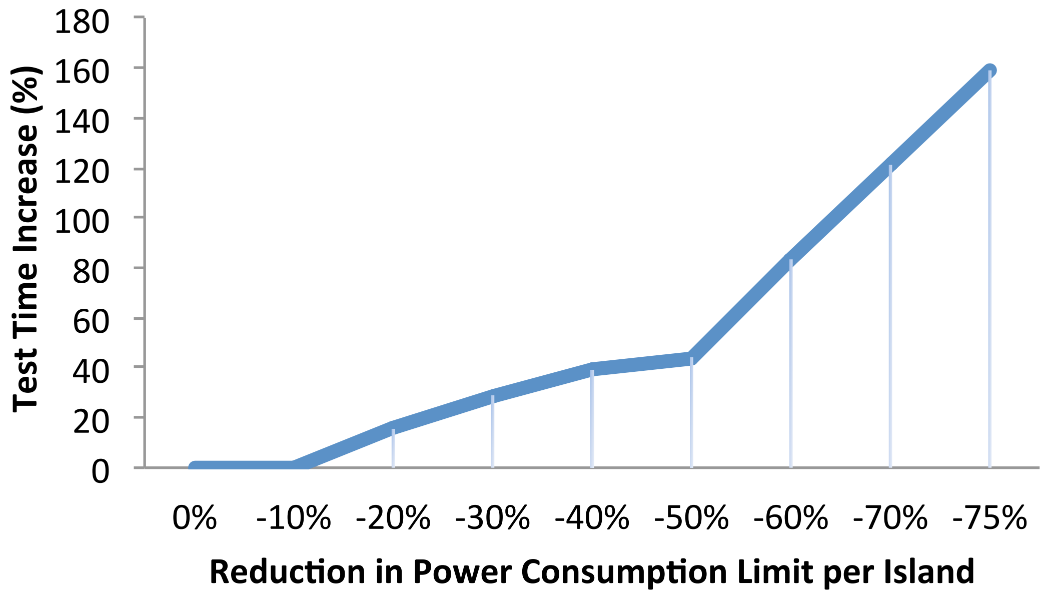


Figure 5.4: Effect of power consumption limit on test time.

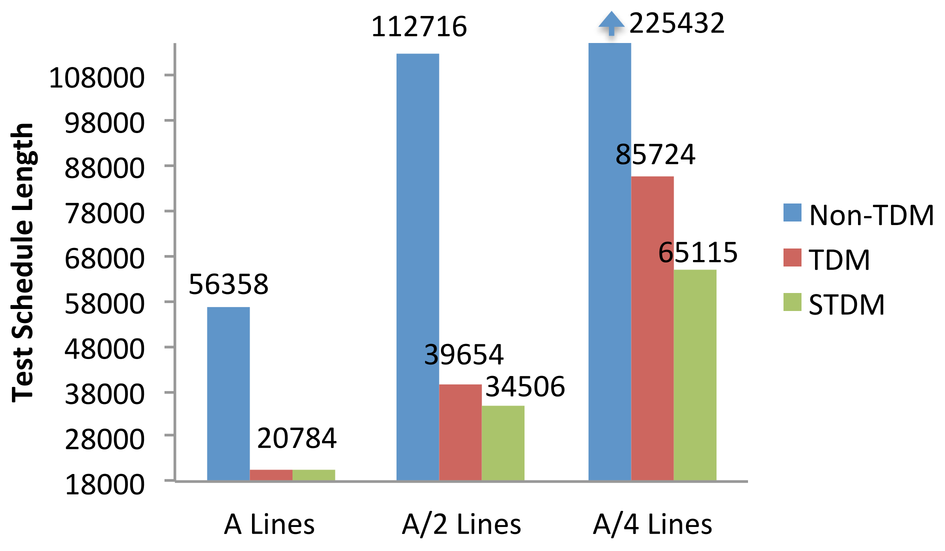


Figure 5.5: STDM vs TDM.

V_{dd}	C_9^B	C_{10}^B	C_{11}^B	C_{12}^B	C_{13}^B	C_{14}^B	C_{15}^B
V_1	1400	N/A	N/A	495	1000	250	2600
V_2	948	297	900	297	N/A	N/A	N/A
V_3	758	238	N/A	202	N/A	N/A	N/A
V_4	515	161	N/A	91	N/A	N/A	N/A
V_5	232	N/A	N/A	42	N/A	N/A	N/A
V_6	104	N/A	N/A	N/A	N/A	N/A	N/A

Table 5.11: SoC-B Power Consumption (in Normalized Time Units) [231].

(B,C)	SoC-A			(B,C)	SoC-B		
	ILP	SA-RP	GRD		ILP	SA-RP	GRD
(2,1)	7023	7193	7698	(3,1)	38377	19295	24993
(2,2)	7023	7023	8198	(3,2)	37936	18995	19645
(2,3)	7023	7226	7981	(3,3)	25707	18495	20180
(3,1)	6548	8098	8093	(4,1)	17995	18995	26344
(3,2)	6548	8828	7568	(4,2)	17694	18370	19115
(3,3)	7023	9018	8838	(4,3)	19795	18295	20395
(4,1)	6548	9698	9638	(5,1)	19270	18495	19200
(4,2)	6548	9942	8415	(5,2)	17095	18995	19600
(4,3)	6548	9518	8260	(5,3)	17395	18295	20387

Table 5.12: Test time per proposed method with power constraints.

ports can be set to the same bitwidth with the bus. Since all wrappers were initially designed to be L bits wide, in order to consider buses of bitwidth smaller than L, the wrapper configuration of every core had to be adjusted in TDM and non-TDM schemes. In STDM, all wrappers retain their original width L. The reshaping of the wrapper has a proportional effect on the length of the tests loaded into the wrapper; they are proportionally increased (every 2x reduction of the WPP bitwidth corresponds to a 2x increase of test time for shifting test data into the core).

The results for bus sizes equal to $L, L/2, L/4$ are shown in Fig. 5.5. STDM outperforms both TDM and non-TDM methods by a considerable margin. Note that the test time for the non-TDM scheme extends above the chart in the case of $L/4$ bits. When the size of the bus is equal to L bits, SDM is not required, since the bus and the wrappers have the same bitwidth, and STDM degenerates to TDM. As the bus bitwidth decreases to $L/2$ and $L/4$, TDM cannot further exploit the released TAM lines and the test time doubles in both cases. In contrast, SDM exploits the released TAM lines and test time does not scale proportionally, thereby offering higher multi-site efficiency.

Adjustments in the WPP size require adjustments of the scan chains and thus they are not always possible, like for example in the case of hard IP - cores or when decompressor

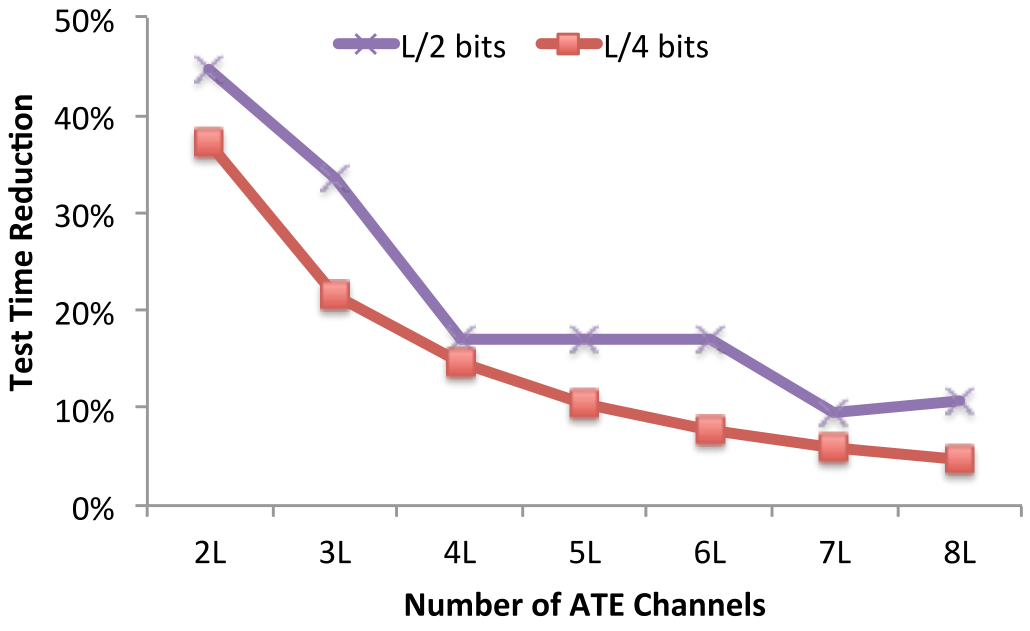


Figure 5.6: Percentage test-time reduction offered by STDM for multi-site testing.

constraints prevent them. At the same time, TDM and non-TDM cannot be applied when the bus is smaller than the WPP width. To show the advantages of STDM in this case, we compare the overall test time needed for testing one million devices using STDM with bus bitwidth equal to $L/2$ and $L/4$ against the overall test time needed for the same number of devices using TDM and non-TDM with bus and WPP bitwidth equal to L. Hence, only the effect of ATE-channel count is considered to evaluate multi-site efficiency. Fig. 5.6 shows the improvement offered by STDM over TDM for various ATE channel counts in the range $[2 \cdot L, 8 \cdot L]$. Note that this count includes ATE channels reserved for control signals. The improvement of STDM over non-TDM (not shown in Fig. 5.6) is in the range of $[65\%, 80\%]$. It is obvious that STDM offers considerable test time savings, especially when the number of available ATE channels is small, because it achieves higher parallelism than the other methods. In practical scenarios, ATE channels constitute an expensive resource and only a small number of channels is available per chip in multi-site testing. Hence, we conclude that STDM is a very efficient approach.

Finally, we examine the case that the different cores have different WPP bitwidths, and they cannot all perfectly match the size of the shared bus. Specifically, we examine the case that the WPPs of cores $C_2, C_3, C_5, C_8, C_{12}, C_{15}$ are $L/2$ bits wide, the WPPs of cores C_1, C_9 are $L/4$ bits wide and the rest of the WPPs are A bits wide. In TDM, the bus is equal in bitwidth to the widest among the WPPs (i.e., L bits), and the smaller wrappers use only part of the bus. In the case of STDM, three different bus bitwidths were considered: L, $L/2$, $L/4$. The overall time improvements of STDM over TDM for testing 1 million devices were equal to 1.1x, 2.7x and 3.6x, respectively. STDM is better suited for buses of small bitwidth and outperforms TDM in this case. It is also important to note that when the bus width reduces by the factor of 2 from L to $L/2$ bits, STDM

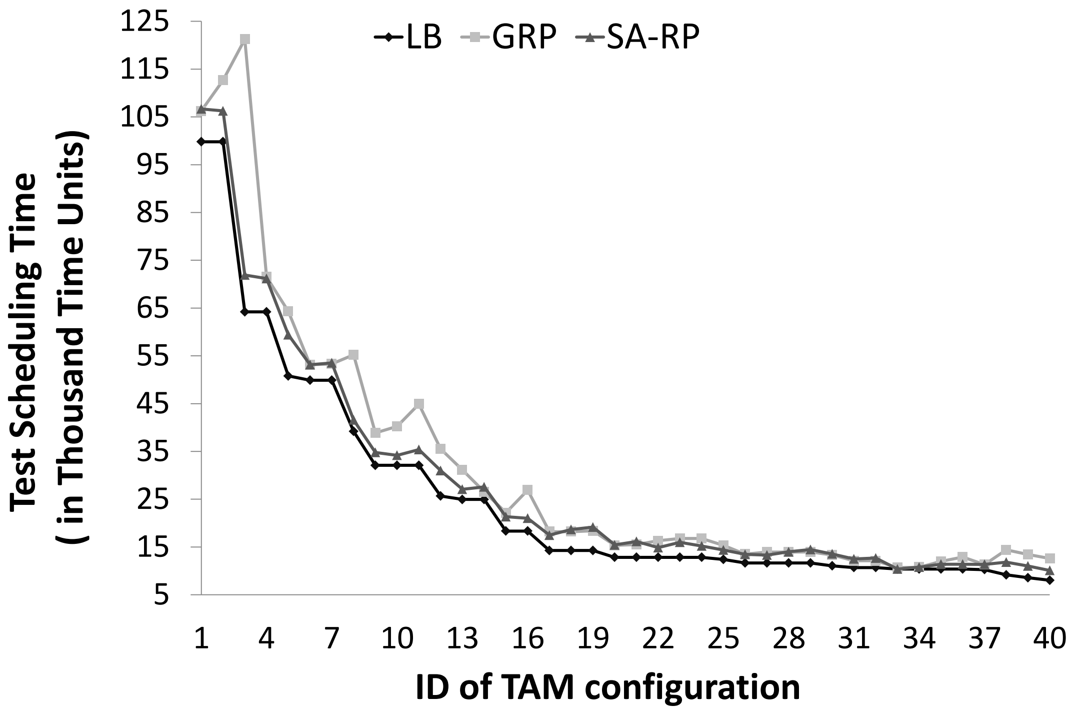


Figure 5.7: Test-scheduling times for TAM configurations of SoC-B.

tests more than double the number of sites. Therefore, there are cases in which STDM not only increases the multi-site factor, but it also offers shorter test time per device than TDM.

5.3 Evaluation of the B-&-B optimization approach

For evaluating the B-&-B optimization approach, initially, we used again the two industrial SoCs, SoC-A and SoC-B, described in section 5.1. Tables 5.1, 5.2 and 5.3 present the test times and the highest rated shift frequencies F_{ij}^{max} for every pair C_i, V_j assuming that all wrapper parallel ports are L_a-bits wide (SoC-A) and L_b-bits wide (SoC-B). Remember that in order to conceal confidential data, test times for the industrial SoCs are presented in normalized units. We assume that the tester provides a clock signal with frequency close to the highest shift frequency reported in Tables 5.1, 5.2 and 5.3, i.e., 200 MHz for SoC-A and 400 MHz for SoC-B. The TDM scheme supports frequencies 25, 50, 100, 200 and 400 MHz.

First we study the evaluation accuracy of GRP on different TAM configurations. This is very critical because GRP is used to guide the B-&-B towards the best TAM configuration. To this end we applied the B-&-B method on SoC-B and we selected 40 representative TAM configurations with the same number of ATE channels (the rest of the configurations exhibit similar behaviour too). For each one of them we generated the test schedules using both the SA-RP and the GRP methods, and we calculated their lower bounds using Eq. (3.22). The test times of the TAM configurations sorted in decreasing order of their LB values are depicted in the graph of Fig. 5.7. Note that the higher the index, the better is the TAM configuration depicted in Fig. 5.7. Even though SA-RP method provides lower test times than the GRP method, they both yield the same results when they are used to compare different TAM configurations. Specifically, as we move from configuration 1 to 40, the test application time of both methods decreases in the same way in the vast majority of the cases. The correlation between GRP and SA-RP using the Kendall's Tau-b coefficient [233] is equal to 0.9, therefore it is obvious that they both drive the B-&-B algorithm towards the same configurations. For SoC-B, the CPU time of GRP is 0.2 seconds for each TAM configuration, while the running time of the SA-RP method is 86 seconds. Therefore, GRP is suitable for the time intensive process of evaluating the nodes of the tree, while SA-RP is suitable for generating the final test schedule for the selected configuration.

$SoC - A$					
N_{ATE}	N_{gen}	P_{rej}	Run Time	Test Time	CNF
α	57.3K	99.6%	7.4sec	4,310	α
$\alpha - 1$	52.2K	99.5%	8.9sec	4,664	1, $\alpha - 1$
$\alpha - 2$	36.5K	99.7%	5.3sec	4,925	$\alpha - 2$
$\alpha - 3$	59.3K	99.0%	9.9sec	5,803	1, α-4
$\alpha - 4$	32.8K	99.5%	4.8sec	5,746	α-4
$\alpha - 5$	43.4K	99.2%	6.9sec	6,740	1, α-6
$\alpha - 6$	20.8K	99.7%	2.9sec	6,896	α-6
$\alpha - 7$	17.0K	99.6%	2.6sec	7,662	α-7
$\alpha - 8$	15.8K	99.6%	2.3sec	8,620	α-8
$\alpha - 9$	18.7K	99.0%	3.3sec	10,388	α-9
$\alpha - 10$	5.6K	99.7%	0.9sec	11,493	α-10
$\alpha - 11$	7.6K	99.6%	1.1sec	13,792	α-11
$\alpha - 12$	4.3K	99.3%	0.7sec	17,240	α-12
$\alpha \ldots \alpha - 12$	212.8K	99.7%	27sec	$4310 - 7489$	-

Table 5.13: Branch-&-Bound Results.

Tables 5.13 and 5.14 show the results of the B-&-B approach on SoC-A and SoC-B respectively. For single-site test applications we run the B-&-B algorithm for all values of N_{ATE} in the range $[\alpha, \alpha - 12]$ for SoC-A and $[\beta, \beta - 12]$ for SoC-B (we note that $\alpha = 2L_a$ and $\beta = 2L_b$). For multi-site test applications we run a single experiment for all values of N_{ATE} in the above ranges (the value of N_{ATE} and the test times are reported as normalized units to conceal confidential information). For the given SoCs, we generated TAM configurations with up to three buses, because more than three buses increase the area overhead without any significant benefit in test time. The columns

$SoC - B$					
N_{ATE}	N_{gen}	P_{rej}	Run Time	Test Time	CNF
β	20.6M	99.9%	2.0h	10,689	1, $\beta - 1$
$\beta - 1$	49.0M	99.9%	5.0h	11,991	1, $\beta - 2$
$\beta - 2$	58.6M	99.9%	6.4h	12,913	1, $\beta - 3$
$\beta - 3$	30.0M	99.9%	3.0h	13,943	2, $\beta - 5$
$\beta - 4$	25.4M	99.9%	2.8h	15,334	1, $\beta - 5$
$\beta - 5$	27.3M	99.8%	3.2h	16,316	1, $\beta - 6$
$\beta - 6$	14.4M	99.8%	3.7h	18,170	2, $\beta - 8$
$\beta - 7$	16.3M	99.8%	2.1h	20,491	1, $\beta - 8$
$\beta - 8$	17.0M	99.8%	2.3h	23,063	2, $\beta - 10$
$\beta - 9$	18.2M	99.8%	2.1h	27,627	1, $\beta - 10$
$\beta - 10$	3.2M	99.7%	0.5h	30,675	1, $\beta - 11$
$\beta - 11$	6.4M	99.8%	0.7h	36,340	1, $\beta - 12$
$\beta - 12$	4.3M	99.9%	0.4h	46,127	1, $\beta - 13$
$\beta \ldots \beta - 12$	81.1M	99.9%	7.6h	$10,689 - 46,127$	-

Table 5.14: Branch-&-Bound Results.

labelled N_{gen} present the total number of TAM configurations that were generated by the B-&-B algorithm, and the columns labelled P_{rej} present the percentage of the N_{gen} nodes that were rejected by the bounding criterion (we report only the number of nodes which are visited by the B-&-B algorithm and not the nodes in the sub-trees eliminated by the bounding criterion). It is obvious that only a very small percentage of the TAM configurations were actually evaluated by the GRP method. In practical applications additional nodes will be rejected due to violation of area constraints. The columns with the labels 'Run Time' and 'Test Time' present the run time of the algorithm and the test time of the best TAM configuration found in each case. The columns with labels 'CNF' present the size of the buses of the best configurations separated with commas. As the value of N_{ATE} decreases, the run time decreases because the number of possible TAM configurations decreases (N_{gen} decreases too). As a result, the size of the tree becomes smaller and the B-&-B algorithm finishes in a smaller amount of time. At the same time, test time increases due to the use of a smaller number of ATE channels. We note that the run time can be further reduced by using multi-threading programming, which can be easily applied in the B-&-B algorithm.

In the last row of table 5.13 and 5.14, we present the results of the multi-site experiment. The number of nodes generated in the multi-site experiment and the run time of both SoCs is much lower than the aggregate number of the generated nodes and the aggregate run time in the standalone experiments respectively. Note that the best multi-site TAM configurations belong in the intervals $N_{ATE} \in [\alpha - 6, \alpha]$ for SoC-A and $N_{ATE} \in [\beta - 6, \beta]$ for SoC-B, as the rest of the N_{ATE} values did not offer any benefits.

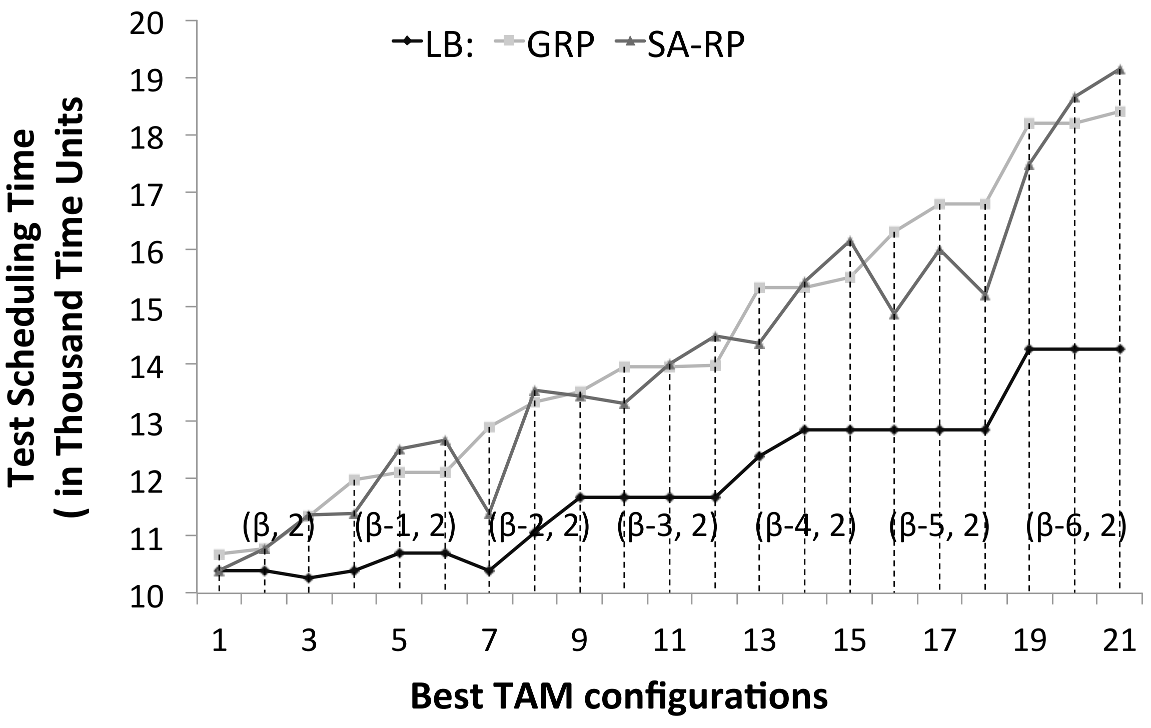


Figure 5.8: Best test times for various values of N_{ATE}.

Fig. 5.8 presents the test times of the three best TAM configurations of SoC-B found for $N_{ATE} \in [\beta - 6, \beta]$. For each triplet we report the test time provided by GRP and SA-RP, as well as their lower bounds. The x-axis shows the various configurations, and the y-axis shows the test times. The pair of values in each parenthesis denotes the number of ATE channels followed by the number of the test buses used (it is the same for each triplet). As the value of N_{ATE} increases (from the right to the left) the test time drops and it approaches the value of LB. This was expected as the increased number of ATE channels increases also the parallelization efficiency of the TDM scheme (we remind that the lower bound calculation was based on the infinite parallelism assumption for the buses). There are cases that the reduction of ATE channels is very beneficial, like the 7^{th} TAM configuration with $N_{ATE} = \beta - 2$ that offers almost the same test time with the $N_{ATE} = \beta$ case. Such cases provide benefits in multi-site test environments and the B-&-B approach is a very effective way to identify them.

In the next experiment we compare the proposed method against other TAM optimization methods. One intuitive method is to partition the set of available TAM lines into a number N_q of equally sized buses $B_1, B_2, \ldots, B_{N_q}$, and connect the cores to the buses in a balanced way as far as the time required for testing the cores of every bus is concerned. Specifically, the set of the cores is partitioned into N_q subsets $S_1, S_2, \ldots S_{N_q}$, such that the aggregate time for testing all the cores of any subset at every voltage level is similar for all subsets. Then, the cores of each subset $S_1, S_2, \ldots S_{N_q}$ are connected to bus $B_1, B_2, \ldots, B_{N_q}$ respectively. We applied this method on SoC-A and SoC-B for $N_q = 2, 3$ using α and β ATE channels respectively. For $N_q = 2$ the test time for SoC-A (SoC-B) using this approach was found to be equal to 7510 (17095) time units, which is 1.7x (1.6x) times higher than the test time of the configuration reported in table 5.13 (table 5.14) for α (β) TAM lines. For $N_q = 3$ the test time for SoC-A (SoC-B) was found to be 2.5x (2.4x) higher than the test time reported in table 5.13 (table 5.14) for α (β) TAM lines.

$SoC-A$				$SoC-B$			
N_{ATE}	[236]	Prop.	Impr.	N_{ATE}	[236]	Prop.	Impr.
α	9.5K	4.3K	2.2x	β	29.0K	10.7K	2.7x
$\alpha-1$	10.3K	4.7K	2.2x	$\beta-1$	32.8K	12.0K	2.7x
$\alpha-2$	11.5K	4.9K	2.3x	$\beta-2$	35.0K	12.9K	2.7x
$\alpha-3$	12.1K	5.8K	2.1x	$\beta-3$	35.5K	13.9K	2.5x
$\alpha-4$	12.4K	5.7K	2.2x	$\beta-4$	38.2K	15.3K	2.5x
$\alpha-5$	13.8K	6.7K	2.1x	$\beta-5$	42.5K	16.3K	2.6x

Table 5.15: Test time comparisons against [236].

Another very efficient optimization technique for single-V_{dd} SoCs that targets both the TAM structure and the test schedule was proposed in [236]. We modified this method to be applied on multi-V_{dd} SoCs as follows: it begins scheduling tests for the first voltage level of every island, and when all tests of an island have been scheduled, it proceed to the next set of tests for this island at the next voltage level. The tests of every island are scheduled independently of the tests for the rest of the islands. For every test the highest possible frequency at its voltage level was used. The test time comparisons with this method are presented in Table 5.15. The proposed method offers 2.1 times to 2.7 times shorter test schedules than [236]. Therefore, it is obvious that the proposed method is much more effective than traditional TAM optimization methods tailored to single-V_{dd} SoCs.

In the following experiment, we optimize the TAM for a large artificial SoC consisting of 33 islands and 105 cores (we distributed multiple copies of the cores of SoC-A and SoC-B at the voltage islands). This SoC was partitioned into 11 areas $A_0, A_1, \ldots, A_{10}$, and the proposed method was applied for each area A_i and every value of $N_{ATE}^{A_i}$ in the range $\alpha \ldots \alpha - 6$ separately. The proposed method took from 2 seconds up to 6 hours to calculate the best TAM configuration per area A_i and value of $N_{ATE}^{A_i}$. Table 5.16 shows the configurations that provide the minimum test time for this SoC, which is equal to 11669. This time is determined by area A_7, which allocates the highest number α of TAM lines. For the other areas the minimum number of TAM lines that did not increase the test time of the SoC were allocated. The test time given by the method proposed in [236] for this large SoC and the same number of ATE lines was equal to 32500, which is 2.8 times higher than the test time of the proposed method. We note that the test schedule provided by [236] does not consider area restrictions, which are expected to further increase the test schedule length. Nevertheless, the proposed method outperforms this single-V_{dd}-based approach for large SoCs too.

Area	CNF	Test Time	Area	CNF	Test Time
A_0	1, $\alpha - 7$	9.18K	A_6	1, $\alpha - 7$	6.45K
A_1	2, $\alpha - 3$	9.96K	A_7	1, $\alpha - 1$	11.67K
A_2	2, 2, $\alpha - 7$	9.09K	A_8	1, 1, $\alpha - 3$	11.07K
A_3	$\alpha - 6$	2.92K	A_9	$\alpha - 6$	9.41K
A_4	$\alpha - 6$	3.38K	A_{10}	$\alpha - 11$, $\alpha - 11$	7.76K
A_5	$\alpha - 6$	6.35K			

Table 5.16: Results for large SoC.

5.4 Evaluation of the TAM distribution approach

In order to show the effectiveness and the scalability of the proposed global TAM distribution approach for very large SoCs (section refsec:3.5.5), we used an artificial SoC consisting of 79 islands and 218 benchmark cores. The benchmark cores and the artificial SoC were designed using the design flow for benchmark cores and artificial SoCs respectively, that are described in section 4.2. The cores were wrapped using IEEE 1500 Std. wrappers with 8 wrapper chains and 8 scan chains. For each core, test vectors targeting transition faults were generated using the Launch-on-Capture scheme. The test vectors were rated using timing simulation for increasing shift frequencies at three different voltage levels, 0.95V, 1.10V and 1.25V. Tables 5.17, 5.18 and 5.19 present the test times of the cores per voltage setting and the highest rated shift frequencies $F_{i,j}^{max}$. The SoC was partitioned into 22 areas $A_1, A_2, \ldots, A_{22}$. Each area includes 8 - 13 cores, and 2 - 5 islands. The number of TAM lines used for testing the artificial SoC were set equal to $N_{ATE} = 200, 100$ and 50.

V_{dd}	$ISCAS$	
level	s38417	s38584
V_1	238	202
V_2	228	200
V_3	236	213
$F_{i,j}^{max}$	190	190

Table 5.17: Test Times (in μs) At Maximum Scan Frequencies $F_{i,j}^{max}$ (in MHz) for ISCAS cores.

The TAM lines were distributed to the 22 areas using the following two approaches:

1. *Baseline Distribution*: each area was assigned a number of TAM lines proportional to the total test data volume of the cores in this area. This is an intuitive design decision driven by test cost related constraints.

V_{dd}	*IWLS*					
level	ac 97	aes core	ethernet	mem ctrl	tv80s	vga lcd
V_1	151	160	6621	386	82	15025
V_2	161	165	6706	389	83	14401
V_3	151	163	6791	382	81	14401
$F_{i,j}^{max}$	200	200	200	200	200	200

Table 5.18: Test Times (in μs) At Maximum Scan Frequencies $F_{i,j}^{max}$ (in MHz) for IWLS cores.

V_{dd}	*IWLS*			
level	des perf	wb conmax	pci bridge32	usb func
V_1	188	160	16	95
V_2	185	153	17	95
V_3	186	148	15	95
$F_{i,j}^{max}$	400	400	400	400

Table 5.19: Test Times (in μs) At Maximum Scan Frequencies $F_{i,j}^{max}$ (in MHz) for IWLS cores.

2. *Proposed Distribution.* The global TAM distribution approach proposed in Section 3.5.5 was applied in this case. Specifically, the distribution table was developed, and the best configuration was selected for each area in order to minimize the test time for the SoC.

In both of these cases, the TDM approach was used to schedule the test data on the selected configuration.

Table 5.20 shows the detailed distribution of the 200 TAM lines among the 22 areas, as well as the overall SoC test times for 200, 100 and 50 TAM lines. The first column presents the 22 areas of the SoC. Columns 2, 3 present the baseline distribution of the 200 TAM lines to the 22 areas, and the test time (in msec) for each area A_i. Columns 4, 5 present the proposed distribution of TAM lines and the test time for each area A_i. The total test time of the baseline method for $N_{ATE} = 200$ was equal to 6.37 msec, while the test time provided by the proposed method was equal to 5.09 msec (note that areas $A_1 \ldots A_{22}$ are tested in parallel). The test times for $N_{ATE} = 100, 50$ are shown in the last two rows of table 5.20. It is obvious that in all cases the proposed distribution method outperforms the baseline approach.

Columns 6, 7 of table 5.16 present the range of TAM lines examined by the global TAM distribution method for each area, and the total run time for the respective area. According to the distribution algorithm presented in Section 3.5.5, in the worst case the range $[1, 179]$ of TAM lines has to be evaluated for each area A_i, in order to optimize the test time for the whole SoC. However, as it is shown in table 5.20, the initial range

	Baseline Distribution		Proposed Distribution			
Area	TAM Lines	Test Time (msec)	TAM Lines	Test Time (msec)	Range 1-179	Run Time (mins)
A_1	8	3.54	6	4.85	5-9	50.53
A_2	7	4.01	6	4.62	4-8	60.53
A_3	7	4.13	6	4.62	4-8	39.21
A_4	7	3.86	6	4.44	4-8	29.68
A_5	8	3.72	7	4.15	5-9	25.17
A_6	4	3.72	4	3.72	2-6	4.49
A_7	7	4.03	5	4.94	3-7	16.74
A_8	6	4.03	5	4.62	3-7	0.85
A_9	6	3.62	5	4.35	3-7	0.63
A_10	5	3.87	4	4.82	2-6	0.96
A_{11}	6	3.91	5	4.64	3-7	2.99
A_{12}	6	4.01	5	4.50	3-7	1.49
A_{13}	5	4.83	5	4.83	3-7	21.62
A_{14}	5	3.95	4	4.97	2-6	4.00
A_{15}	6	4.04	5	4.68	3-7	2.81
A_{16}	6	4.04	5	4.46	3-7	3.75
A_{17}	6	3.54	5	4.18	3-7	1.62
A_{18}	5	4.14	4	4.98	2-6	0.10
A_{19}	4	3.91	4	3.91	2-6	0.06
A_{20}	4	3.87	3	4.79	1-5	0.79
A_{21}	27	6.03	32	5.09	30-34	0.50
A_{22}	55	6.37	69	5.08	66-70	1.00
SoC	200	6.37	200	5.09	-	269.52
SoC	100	12.99	100	10.8	-	78.32
SoC	50	25.04	50	23.37	-	42.48

Table 5.20: Results for large SoC with benchmark cores.

$[1 \ldots 179]$ of TAM lines was considerably reduced in all areas, and no more that 5 different values of the number of TAM lines were evaluated for each area (the value of Z was set equal to 2 and the optimal distribution was found in the first repetition of the distribution algorithm).

The last three rows present the total run times of the proposed method for $N_{ATE} = 200, 100, 50$. The total run time reported in each case includes also the calculation of the initial distribution table using the GRP approach. As the number of TAM lines increases from 50 to 200, the size of the search tree increases exponentially. However, the high rejection rate offered by the low bound criterion prevents the exponential increase of the run time of the B-&-B algorithm.

In order to show the effect of $LB^{(1)}$ and $LB^{(2)}$ on the rejection rate of the B-&-B approach, we applied this approach on each one of the 22 areas of the large SoC, to find the best configuration for 16 TAM lines and 1-3 buses. The total number of TAM configurations with 1-3 buses for each area was equal to 1.76x10^8. The total number of configurations that were not rejected by $LB^{(1)}$ was equal to 2.12x10^7 (rejection rate 87.9%), while the total number of configurations that were not rejected by both $LB^{(1)}$ and $LB^{(2)}$ was equal to 5.50x10^5 (rejection rate 99.7%). It is obvious that the use of both $LB^{(1)}$ and $LB^{(2)}$ reduces considerably the computational effort and thus the run time of the B-&-B algorithm. Further reduction of the run time can be achieved by exploiting the parallel nature of the TAM optimization method.

In the last experiment we study the effect of power constraints on the optimization of the TAM structure. The most common design practice is to consider power constraints during the generation of the final test schedule. The proposed TDM method can ensure that the final test schedule complies with the power constraints of the SoC, as shown in section 5.1. However, power constraints can be also considered during the optimization of the TAM structure to boost the effectiveness of TDM in reducing the test time. To show this potential we optimized the TAM structure of the artificial SoC for $N_{ATE} = 100$ using the following three approaches:

1. *No Power Constraints.* In this case the proposed method was applied without any power constraints.

2. *Power Constraints on Final Test Schedule.* In this case the TAM structure was optimized without considering any power constraints, but the final test schedule was derived by setting specific power constraints to the TDM process. These constraints were set in such a way as to reduce the peak power of every area by a percentage between 5%-15%.

3. *Power Constraints on both TAM Optimization and Final Test Schedule.* The same power constraints with case (2) were set a) during the evaluation of every different configuration in the B-&-B approach, and b) during the generation of the final test schedule. As a result, both the generated TAM structure and the final test schedule were optimized taking into account these constraints too.

The results are shown in Fig. 5.9. As it was expected, power constraints increase the test time of the SoC. However, when power constraints are consider during both the TAM optimization process and the final test schedule generation process the test time drops. Therefore, we conclude that if power constraints are considered early during the TAM optimization process, the test time can be further reduced when the final test schedule is generated.

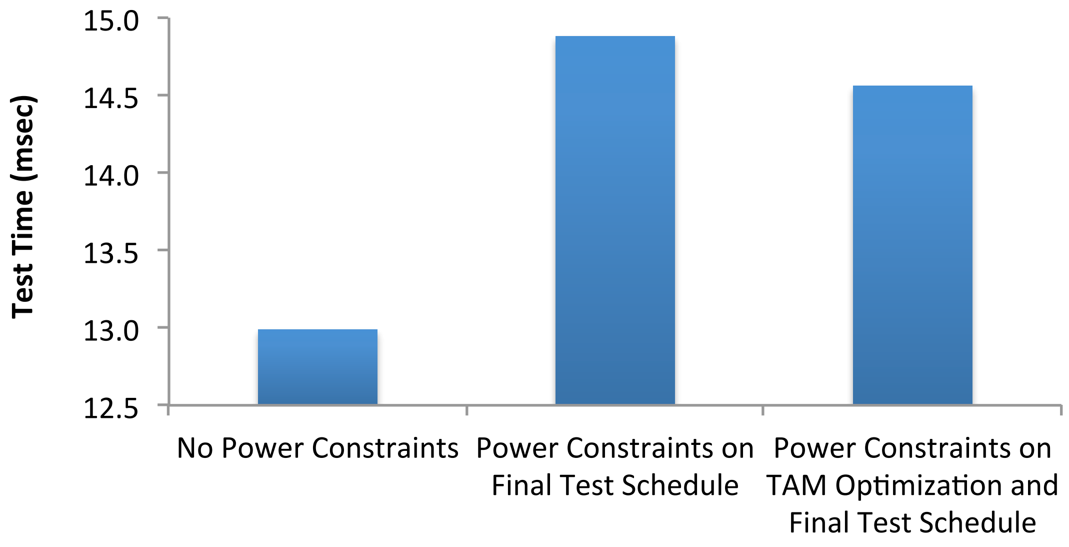


Figure 5.9: Test time under power constraints.

5.5 Evaluation of the critical path-oriented and thermal aware X-fill method for high un-modelled coverage

Benchmark Cores	IN	OUT	FFs	Scan Chains	Number of Test Vectors
ethernet	161	179	10544	64	1395
des3	267	96	8808	32	130
aes_cipher	268	137	530	8	524
wb_conmax	1139	1424	770	8	122
tv80s	23	40	359	8	404
usbf	137	129	1746	8	184
s38417	47	122	1564	16	254
s38584	31	294	1166	16	60

Table 5.21: Benchmark cores and experimental data.

In order to evaluate the proposed methodology, we run experiments on the largest *ISCAS* and *IWLS* benchmark cores shown in Table 5.21. The cores were designed using the design flow for benchmark cores presented in section 4.2. However, due to the requirements of the proposed method, the floor-plan of each core was partitioned into a number of blocks, which was determined based on the area of the core and the number of scan cells inside each block. In addition, the critical paths were identified using post-layout timing analysis based upon standard operating condition (i.e., process variation $P = 1$, power supply voltage $V = 1.1V$ and temperature $T = 25^oC$). Specifically, all paths with delays within a margin of 90% of the worst path delay were classified as critical paths. Every

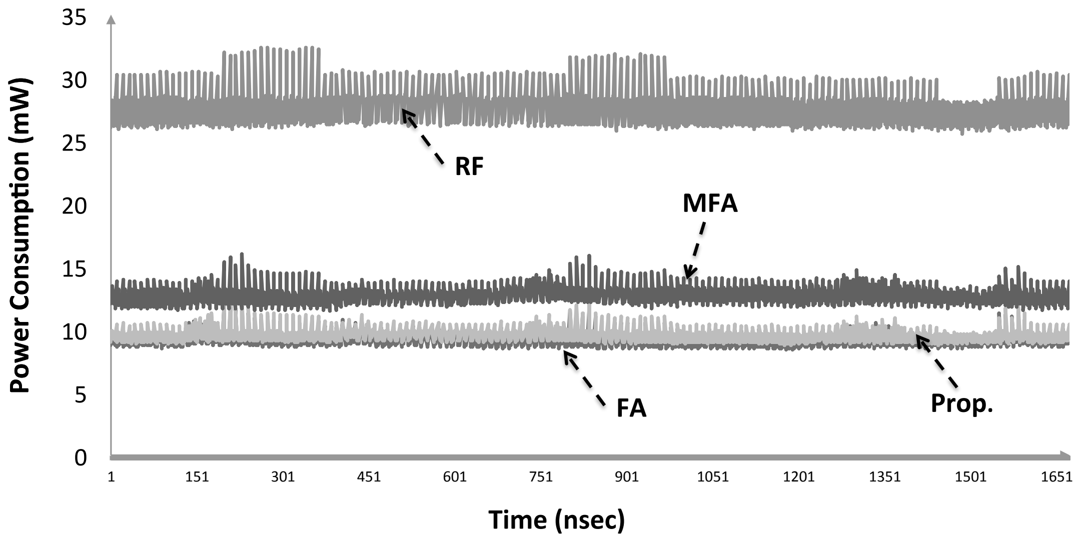


Figure 5.10: Power consumption of Ethernet core.

block that includes at least a part of a critical path is considered as critical block of zone Z_0. A nearest-neighbourhood search was applied to determine the blocks of thermal-safe zones Z_0 and Z_1. The weights w_{Z_0}, w_{Z_1} and w_{Z_2} were set equal to 3, 1.5 and 1 respectively.

The Random-Fill method (RF), the FA method proposed in [205] and Modified-Fill-Adjacent (MFA) method proposed in [219] were also implemented for comparison purposes. All these methods were applied on compacted test sets generated for complete coverage of stuck−at faults. Similar to [219], [229], [230] these test sets were evaluated for un-modeled defect coverage by using the transition-fault model as surrogate fault model (i.e., a fault model that is not targeted by the generated test sets). In the proposed method and the MFA, the same output-deviation based metric proposed in [209] was used, and 30 test vector candidates were generated for each test cube.

The power profile and the floorplan of every core and every test set were given as input to the *Hotspot* tool [204], in order to produce the static and dynamic temperature profile for each block of the core. Then, in order to measure the impact of the evaluated methods on the delay of critical paths, especially under harsh operating conditions and large process variations, we followed a worst-case approach. Therefore, we used the steady profile generated by *Hotspot* for each core - test set pair, to determine the operating condition of each block and every scan cell in the block. This operation condition was provided to Cadence Encounter, in order to perform timing analysis using on-chip variations and the two slow-corner libraries of the 45nm Nangate technology, namely the worst-low library and the slow library. Both libraries use low power supply voltage equal to 0.95V, and temperatures set at −40 and 125 grades in Celsius scale respectively. Then, the exact path delays at the given operating condition of every block were generated by using the Encounter static analyzer, which interpolated the library−based timing information to calculate the delays of the standard cells.

Fig. 5.10 presents the power consumption of the Ethernet core of the *IWLS* suite.

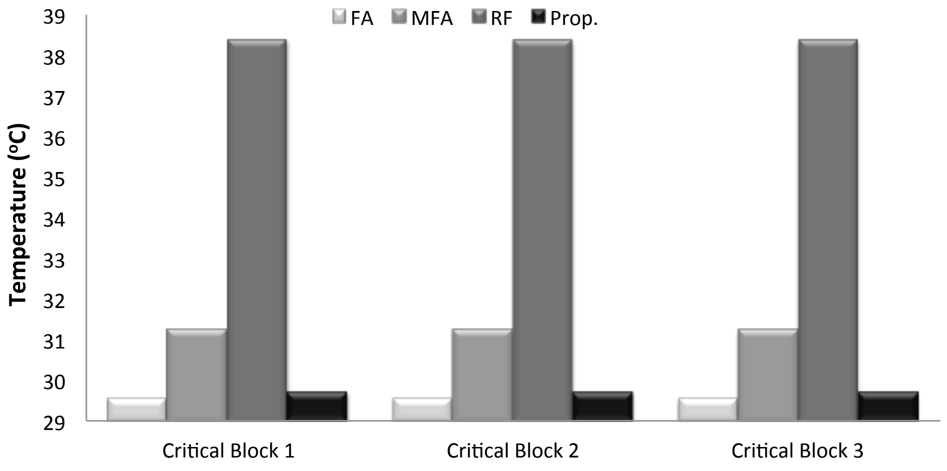


Figure 5.11: Temperature at critical areas of Ethernet core.

This core was partitioned into 100 blocks and the value of R was set equal to 0.85. The horizontal axis presents the test time (nsec) and the vertical axis presents the power consumption of the core. The average power consumption of the proposed method is equal to 9.54mW, and it is slightly higher than the 9.20mW of average power consumption of the most power-oriented method, the FA method. The power consumption of the MFA method is equal to 12.73mW and it is clearly higher than both the proposed and the FA methods, while the RF method consumes almost three times higher average power consumption, that is 27.65mW.

In the next experiment we compare the four methods in respect with the temperature developed at the critical blocks of the Ethernet core. Specifically, with the post-layout timing analysis on the Ethernet core we identified 3 critical blocks , 7 blocks at the Z_0 zone, 11 blocks at the Z_1 zone and the rest 79 blocks were left un-constrained. The 'steady' temperature at each one of the critical blocks is depicted in Fig. 5.11 for FA, MFA, RF and the proposed method. It is obvious that the temperature at the critical blocks for both the proposed method and FA is the lowest one, and it is equal to 29.5^oC. MFA increases the temperature by 2 degrees, i.e., at 31.2^oC. Finally RF develops significant higher temperature at 38.4^oC, which is almost 10 grades higher than that of the proposed method.

Remember that the temperature that is developed upon each cell in a pathway affects the overall delay in this path. Specifically, when they are kept in low temperature, the path delay is limited, while high temperature leads to excessive path delay. Then, according to the above given temperature profiles of each method, it is expected that the *Fill Adjacent* and the proposed method will present smaller path delays when compared to the *MFA* and the *Random Fill* method. Indeed, the timing analysis showed that the worst path delay of the former methods is 1.61ns, while the worst path delays of the later methods are 1.64ns and 1.72ns correspondingly. Therefore, the proposed method reduces the delay in critical paths limiting that way the possibility to have yield losses due to excessive path

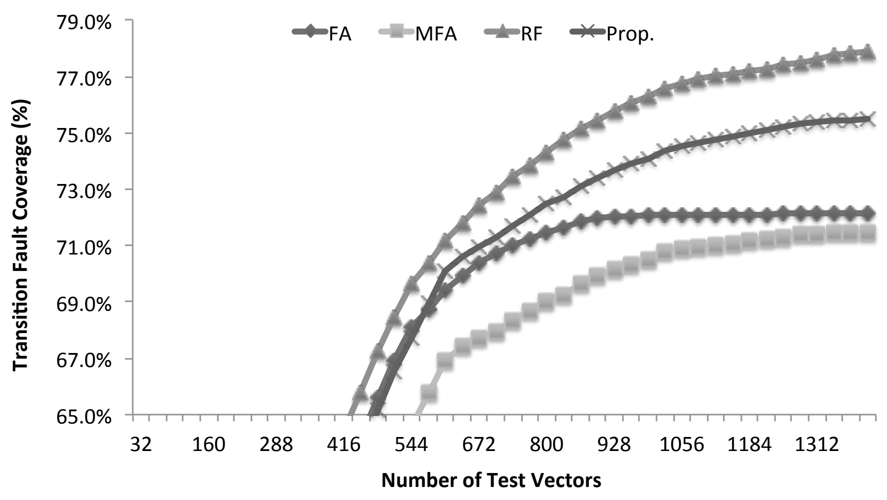


Figure 5.12: Transition-fault coverage ramp-up for Ethernet core.

delays.

In order to compare the four X-fill methods with respect to the un-modeled defect coverage, we present in Fig. 5.12 the transition-fault coverage provided by each one of them. The x-axis presents the number of vectors applied, and the y-axis the transition-fault coverage for each number of vectors applied. It is obvious that both FA and MFA provide lower transition-fault coverage than the RF and the Proposed methods. The highest transition-fault coverage is provided by the RF method, which however is not much higher than the transition-fault coverage offered by the proposed method. In addition, both of them offer very high ramp-up, which offers further test time savings at abort-at-first-fail environments. We remind that, as shown in Fig. 5.10 and Fig. 5.11, the power consumption and the temperature of the RF method are both very high. Therefore, the proposed method clearly outperforms all the other methods when all three parameters of power, temperature and un-modeled defect coverage are evaluated.

Benchmark Cores	RF	Proposed Method			MFA	FA
		$P = 0$	$P = 0.85$	$P = 1$		
ethernet	27.65	26.71	9.54	25.07	12.73	9.20
des3	19.60	14.34	11.47	12.65	13.32	12.74
aes_cipher	10.73	10.23	8.98	9.51	10.25	8.69
wb_conmax	6.67	6.09	4.77	5.35	6.37	4.41
tv80s	1.62	1.13	0.71	0.66	1.03	0.62
usbf	5.79	4.22	2.56	2.33	2.91	2.18
s38417	2.33	1.76	1.25	1.16	1.54	1.15
s38584	2.56	2.36	1.91	2.08	2.09	1.82

Table 5.22: Average Power Consumption (mWatt).

Benchmark Cores	RF	Proposed Method $P = 0$	Proposed Method $P = 0.85$	Proposed Method $P = 1$	MFA	FA
ethernet	38.43	37.98	29.73	37.19	31.26	29.57
des3	35.38	32.63	31.13	31.75	32.10	31.8
aes_cipher	41.57	40.79	38.88	39.68	40.82	38.44
wb_conmax	32.09	31.49	30.11	30.72	31.78	29.75
tv80s	30.57	28.93	27.54	27.35	28.6	27.24
usbf	33.53	31.25	28.86	28.52	29.37	28.31
s38417	29.12	28.14	27.29	27.13	27.78	27.11
s38584	30.3	29.88	28.29	29.32	29.36	28.8

Table 5.23: Temperature (Celsius Degrees).

Benchmark Cores	RF	Proposed Method $P = 0$	Proposed Method $P = 0.85$	Proposed Method $P = 1$	MFA	FA
ethernet	77.89	77.28	75.48	76.46	71.48	72.14
des3	90.61	88.80	84.01	80.67	80.55	80.21
aes_cipher	84.76	84.54	83.47	83.02	84.41	82.47
wb_conmax	93.8	93.37	92.13	92.25	93.68	91.35
tv80s	42.01	41.91	39.98	38.24	39.82	38.22
usbf	23.95	23.88	22.60	21.86	22.60	21.83
s38417	93.23	91.96	86.77	79.29	84.59	79.25
s38584	86.87	84.89	79.74	80.02	79.63	77.51

Table 5.24: Transition Fault Coverage (%).

Tables 5.22 5.23, 5.24 present the average power consumption, the steady temperature and the transition fault coverage of the RF, FA, MFA and the proposed method for all benchmark circuits. In particular, for the proposed method we present results for $P = 0$, $P = 0.85$ and $P = 1$. In the first case ($P = 0$) only a small number of scan cells is power constrained and thus this case provides results which are closer to the RF method than the other methods. On the contrary, in the $P = 1$ case all scan cells are power constrained, and this case is similar to the FA method. It is obvious that in all cases the proposed method offers power consumption and temperature that is very close to the most power-efficient FA method, while at the same time it offers un-modelled defect coverage that is very close to the RF method. Therefore, we conclude that the proposed method combines the advantages of both FA and RF method, and it offers high un-modelled defect coverage without any adverse impact on power dissipation and temperature during testing.

Chapter 6

Conclusions

This research has focused in the development of an efficient, integrated and computational friendly methodology able to minimize the test cost of moderate, large and very large multi-core, DVFS-based SoCs with voltage islands while power constraints are met. We have introduced Time Division Multiplexing (TDM), a novel method for testing DVFS-based SoCs with multiple voltage islands. We have presented three power-aware test scheduling approaches that effectively exploit the advantages offered by the TDM method at maximum level. Specifically, we proposed an integer-linear programming approach able to produce optimal test schedules for SoCs of small and moderate size. For large and very large SoCs, where the complexity of the ILP model is cost prohibitive, we proposed a scalable rectangle-packing/simulated-annealing approach able to provide with near-optimal, cost-effective solutions. For early design-phase decisions as well as for cases with strict CPU-time limits, a greedy approach was proposed that delivers fairly good results. In addition, we showed that the proposed TDM methods manage to compensate effectively the given power constraints and derive test schedules that are close or equal to efficiency to test schedules that are derived without taking into account power limitations. Experimental results on two industrial SoCs show the superiority of the TDM-based approach against conventional approaches.

We have extended the TDM method with Space Division Multiplexing (SDM) creating a new Space and Time Division Multiplexing (STDM) methodology that offers a highly efficient solution for multi-site test applications. In particular, we have shown that space multiplexing permits the use of test access mechanisms (TAMs) that are narrower than the wrappers of the embedded cores in an SoC while time multiplexing can exploit the available frequency bandwidth to parallelize test application, thereby minimizing the additional time overhead. Experimental results prove that STDM outperforms both TDM and non-TDM methods in multi-site test applications. The gain margin in test time is even greater in practical scenarios where ATE channels constitute an expensive resource and only a small number of channels is available per chip. There are cases in which STDM not only increases the multi-site factor, but it also offers shorter test time per device than

TDM.

Furthermore, we proposed a branch-&-bound (B-&-B) technique to optimize the test access mechanism for minimizing test-time when Time Division Multiplexing is used to schedule tests. Initially, we proved that in order to maximize the benefits offered by TDM, the underlying TAM should be tailored to TDM. Then, we mathematically derived a strict, computationally simple and accurate bounding criterion that enables the proposed B-&-B algorithm to rapidly prune more than 99% of the ineffective TAM configurations. We evaluated the remaining TAM configurations using a fast greedy test-scheduling approach, the use of which is justified by its high correlation (in evaluating TAM designs) with the very effective (but much slower) simulated annealing approach. Experimental results show that we manage to identify the most effective TAM configurations. For very large SoCs, a global TAM distribution approach is proposed, which optimally distributes the TAM lines into multiple SoC areas, in limited computational time. To the best of our knowledge, this is the first TAM optimization approach that takes into account the unique characteristics of multi-V_{dd} SoCs and the benefits of the highly effective TDM approach, and offers a complete solution to the test-scheduling problem for multi-V_{dd} SoCs. Experiments on two industrial SoCs, as well as on two very large artificial SoCs have shown the benefits of the proposed method in both single-site and multi-site test applications.

Finally, we have introduced a critical path−oriented thermal aware 'X'−filling for high un−modelled defect coverage. Experimental results show that interconnection delays in the area of a critical path can be fairly reduced creating a thermal safe neighbourhood around it. Such a caution protects from failures which are caused by increased temperatures during testing, and thus decreases yield loss. The proposed method succeeds this by applying power-oriented filling to the unspecified bits of the test vectors that are more critical for delay failures. In addition, it has been shown that the proposed method succeeds better un−modelled defect coverage in benchmark circuits than the existing X−filling low power techniques in bibliography. The traditional power-oriented X-fill methods do not correlate the thermal activity with delay failures, and they consume all the unspecified bits to reduce the power dissipation at every region of the core. Therefore, they adversely affect the un-modelled defect coverage of the generated test vectors. In contrast, the proposed method fills the non-critical unspecified bits using a probabilistic model based on output deviations that increase the un-modelled defect coverage of the test vectors. Overall, the proposed method offers a fair trade−off between critical paths delay and un−modelled defect coverage. Due to its nature, it may complement the formulation of the problem to be solved in many existing thermal and power aware techniques.

Concluding this research work, we would like to mention that the proposed methodology can be effectively extended and form the basis for testing effectively newcomers in the SoC design industry such as the 3D SoCs. Early results have been presented in [237].

Bibliography

[1] L. Atzori, A. Iera, and G. Morabito, The Internet of Things: A survey, *ELSEVIER Computer Networks* vol. 54 (2010) 2787–2805.

[2] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, Future Internet: The Internet of Things architecture, possible applications and key challenges, *Proc. of the International Conference Frontiers of Information Technology* (2012) 257–260.

[3] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, Internet of Things (IoT): A vision, architectural elements, and future directions, *ELSEVIER Future Generation of Computer Systems* vol. 29 (2013) 1645–1660.

[4] Synopsys, DesignWare IP for IoT SoC Designs, *http://www.synopsys.com/dw/doc.php/ds/o/internet_of_things_brochure.pdf* (2016)

[5] Gartner, IoT Market Report, *http://www.gartner.com/newsroom/id/3165317* (2015)

[6] D. Bol et al, Green SoCs for a Sustainable Internet-of-Things, *Proc. of the IEEE Faible Tension Faible Consommation - FTFC* (2013)

[7] P. Benett, Cadence Engineering Services, The why, where and what of low-power SoC design, *http://www.eetimes.com/document.asp?doc_id=1276973&* EETimes (2004)

[8] G. Smith, Test and system level integration, IEEE Des. Test Comput., *IEEE Design and Test of Computers* vol. 14 (1997) No.4.

[9] N. Weste, D. Harris, *CMOS VLSI Design - A Circuits and Systems Perspective*, 4th Edition, Addison-Wesley (2011).

[10] M. Pedram, Power minimization in IC design: principles and applications, *Proceedings of ACM Transactions on Design Automation of Electronic Systems* (1996) 3–56.

[11] K. Roy, "Leakage power reduction in low-voltage CMOS designs, *Proceedings of International Conference on Electronics, Circuits and Systems* (1998) 167–173.

[12] Y. Bin, W. Haihong, C. Riccobene, X. Qi, and L. Ming-Ren, Limits of gate-oxide scaling in nano-transistors, *Proceedings of IEEE Symposium on VLSI Technology* (2000) 90–91.

[13] M. Keating, D. Flynn, R. Aitken, A. Gibbons, and K. Shi, Low Power Methodology Manual, *Springer-Verlag* (2007).

[14] L. Benini and G. De Micheli, Automatic synthesis of low-power gated-clock finite-state machines, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 15 (1996) 630–643.

[15] M. Alidina, J. Monteiro, S. Devadas, a. Ghosh, and M. Papaefthymiou, Precomputation-based sequential logic optimization for low power, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* vol. 2 (1994) 426–436.

[16] L. Benini, G.D. Micheli, E. Macii, M. Poncino, and R. Scarsi, Symbolic synthesis of clock-gating logic for power optimization of control-oriented synchronous networks, *Proceedings of IEEE European Design and Test Conference* (1997) 514–520.

[17] A. Correale, Overview of the power minimization techniques employed in the IBM PowerPC 4xx embedded controllers, *Proceedings of International Symposium on Low Power Electronics and Design* (1995) 75–80.

[18] M. Munch, B. Wurth, R. Mehra, J. Sproch, and N. Wehn, Automating RT-level operand isolation to minimize power consumption in datapaths, *Proceedings of Design, Automation and Test in Europe Conference and Exhibition* (2000) 624–631.

[19] T. Chi-Ying, M. Pedram, and A. Despain, Power efficient technology decomposition and mapping under an extended power consumption model, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 13 (1994) 1110–1122.

[20] V.D. Agrawal, M.L. Bushnell, G. Parthasarathy, and R. Ramadoss, Digital Circuit Design for Minimum Transient Energy and a Linear Programming Method, *Proceedings of International Conference on VLSI Design* (1999) 434–439.

[21] S. Kim, J. Kim, and S. Hwang, New path balancing algorithm for glitch power reduction, *IEE Proceedings - Circuits, Devices and Systems* vol. 148 (2001) 151–156.

[22] B. Nikolic, J.M. Rabaey, and A.P. Chandrakasan, Digital integrated circuits: A design perspective *Pearson Education* (2003).

[23] A. Ghosh, S. Devadas, K. Keutzer, and J. White, Estimation of average switching activity in combinational and sequential circuits, *Proceedings of ACM/IEEE Design Automation Conference* (1992) 253–259.

[24] A. Srivastava, D. Sylvester, and D. Blaauw, Power minimization using simultaneous gate sizing, dual-Vdd and dual-Vth assignment, *Proceedings of ACM/IEEE Design Automation Conference* (2004) 783–787.

[25] C. Chunhong and M. Sarrafzadeh, Simultaneous voltage scaling and gate sizing for low-power design, *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* vol. 49 (2002) 400–408.

[26] M. Borah, R. Owens, and M. Irwin, Transistor sizing for low power CMOS circuits, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 15 (1996) 665–671.

[27] W.H. Kao, N. Fathi, and C. Lee, Algorithms for automatic transistor sizing in CMOS digital circuits, *Proceedings of ACM/IEEE Design Automation Conference* (1985) 781–784.

[28] S. Mukhopadhyay and K. Roy, Accurate modeling of transistor stacks to effectively reduce total standby leakage in nano-scale CMOS circuits, *Proceedings of Symposium on VLSI Circuits* (2003) 53–56.

[29] N. Hanchate and N. Ranganathan, A new technique for leakage reduction in CMOS circuits using self-controlled stacked transistors, *Proceedings of International Conference on VLSI Design* (2004) 228–233.

[30] M. Johnson, D. Somasekhar, C. Lih-Yih, and K. Roy, Leakage control with efficient use of transistor stacks in single threshold CMOS, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* vol. 10 (2002) 1–5.

[31] L. Yong and G. Zhiqiang, Timing analysis of transistor stack for leakage power saving, *Proceedings of International Conference on Electronics, Circuits and Systems* (2002) 41–44.

[32] A. Abdollahi, F. Fallah, and M. Pedram, Leakage current reduction in CMOS VLSI circuits by input vector control, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* vol. 12 (2004) 140–154.

[33] C. Xiaotao, F. Dongrui, H. Yinhe, Z. Zhimin, and L. Xiaowei, Fast Algorithm for Leakage Power Reduction by Input Vector Control, *Proceedings of International Conference on ASIC* (2005) 98–101.

[34] L. Wei, Z. Chen, K. Roy, M. Johnson, Y. Ye, and V. De, Design and optimization of dual-threshold circuits for low-voltage low-power applications, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* vol. 7 (1999) 16–24.

[35] M. Ketkar and S.S. Sapatnekar, Standby power optimization via transistor sizing and dual threshold voltage assignment, *Proceedings of IEEE International Conference on Computer Aided Design* (2002) 375–378.

[36] F. Gao and J.P. Hayes, Total power reduction in CMOS circuits via gate sizing and multiple threshold voltages, *Proceedings of ACM/IEEE Design Automation Conference* (2005) 31–36.

[37] T. Kuroda, T. Fujita, S. Mita, T. Nagamatsu, S. Yoshioka, K. Suzuki, F. Sano, M. Norishima, M. Murota, M. Kako, M. Kinugawa, M. Kakumu, and T. Sakurai, A 0.9-V, 150-MHz, 10-mW, 4 mm2, 2-D discrete cosine transform core processor with variable threshold-voltage (VT) scheme, *IEEE Journal of Solid-State Circuits* vol. 31 (1996) 1770–1779.

[38] S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, S. Shigematsu, and J. Yamada, 1-V power supply high-speed digital circuit technology with multithreshold-voltage CMOS, *IEEE Journal of Solid-State Circuits* vol. 30 (1995) 847–854.

[39] B. Calhoun, F. Honore, and A. Chandrakasan, A leakage reduction methodology for distributed MTCMOS, *IEEE Journal of Solid-State Circuits* vol. 39 (2004) 818–826.

[40] H. Ananthan, C.H. Kim, and K. Roy, Larger-than-Vdd forward body bias in sub-0.5V nanoscale CMOS, *Proceedings of International Symposium on Low Power Electronics and Design* (2004) 8–13.

[41] K. von Arnim, E. Borinski, P. Seegebrecht, H. Fiedler, R. Brederlow, R. Thewes, J. Berthold, and C. Pacha, Efficiency of body biasing in 90-nm CMOS for low-power digital circuits, *IEEE Journal of Solid-State Circuits* vol. 40 (2005) 1549–1556.

[42] C. Kim and K. Roy, Dynamic VTH Scaling Scheme for Active Leakage Power Reduction, *Proceedings of Conference on Design, Automation, and Test in Europe* (2002) 163–167.

[43] H. Mizuno, K. Ishibashi, T. Shimura, T. Hattori, S. Narita, K. Shiozawa, S. Ikeda, and K. Uchiyama, An 18- uA Standby Current 1.8-V, 200-MHz Microprocessor with Self-Substrate-Biased Data-Retention Mode, *IEEE Journal of Solid-State Circuits* vol. 34 (1999) 1492–1500.

[44] T. D. Burd and R. W. Brodersen, Design Issues for Dynamic Voltage Scaling, *Proc. of the International Symposium on Low Power Electronics and Design* (2000) 9–14.

[45] G. Magklis, G. Semeraro, D. H. Albonesi, S. G. Dropsho, S. Dwarkadas, and M. L. Scott, Dynamic Frequency and Voltage Scaling for a Multiple-Clock-Domain Microprocessor *IEEE Micro* vol. 23 (2003) 62–68.

[46] G.K. Yeap, Practical Low Power Digital VLSI Design, *Springer* (1997).

[47] K. Flautner, D. Flynn, D. Roberts, and D. Patel, IEM926: an energy efficient SoC with dynamic voltage scaling, *Proceedings of IEEE Design, Automation and Test in Europe Conference* (2004) 324–327.

[48] C. Chen, A. Srivastava, and M. Sanafzadeh, On gate level power optimization using dual-supply voltages, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* vol. 9 (2001) 616–629.

[49] B. Amelifard, A. Afzali-Kusha, and A. Khadernzadeh, Enhancing the Efficiency of Cluster Voltage Scaling Technique for Low-power Application, *Proceedings of IEEE International Symposium on Circuits and Systems* (2005) 1666–1669.

[50] K. Usami and M. Horowitz, Clustered voltage scaling technique for low-power design, *Proceedings of International Symposium on Low Power Electronics and Design* (1995) 3–8.

[51] K. Usami, K. Nogami, M. Igarashi, F. Minami, Y. Kawasaki, T. Ishikawa, M. Kanazawa, T. Aoki, M. Takano, C. Mizuno, M. Ichida, S. Sonoda, M. Takahashi, and N. Hatanaka, Automated low-power technique exploiting multiple supply voltages applied to a media processor, *Proceedings of IEEE Custom Integrated Circuits Conference* (1997) 131–134.

[52] ARM 1176JZ(F)-S documentation, *http://www.arm.com/products/CPUs/ARM1176.html*

[53] Intel Corp, Intel XScale Core Developer's Manual, *http://developer.intel.com/design/intelxscale/* (2003).

[54] Intel Corp, Itanium2, *http://www.intel/research* (2004)

[55] Transmeta Corporation, Crusoe processor documentation, *http://www.transmeta.com* (2002).

[56] Intel: PXA270 Processor Datasheet, *http://www.phytec.com/pdf/datasheets/* (2007)

[57] J.M. Rabaey, A. Chandrakasan, and B. Nikolic, Digital Integrated Circuits (2nd Edition), *Prentice Hall* (2003).

[58] D. E. Lackey, P. S. Zuchowski, T. R. Bednar, D. W. Stout, S. W. Gould, and J. M. Cohn, Managing Power and Performance for System-on-Chip Designs Using Voltage Islands, *Proc. of the IEEE/ACM International Conference on Computer Aided Design* (2002), 195–202.

[59] Q. Ma and E.F. Young, Voltage island-driven floorplanning, *Proceedings of IEEE International Conference on Computer Aided Design* (2007) 644–649.

[60] L. Bin, C. Yici, Z. Qiang, and H. Xianlong, Power driven placement with layout aware supply voltage assignment for voltage island generation in dual-Vdd designs, *Proceedings of Asia and South Pacific Conference on Design Automation* (2006) 582–587.

[61] R.L. Ching, E.F. Young, K.C. Leung, and C. Chu, Post-placement voltage island generation, *Proceedings of ACM/IEEE International Conference on Computer Aided Design* (2006) 641–646.

[62] W. Mak and J. Chen, Voltage Island Generation under Performance Requirement for SoC Designs, *Proceedings of Asia and South Pacific Design Automation Conference* (2007) 798–803.

[63] R. Puri, D. Kung, and L. Stok, Minimizing Power with Flexible Voltage Islands, *IEEE International Symposium on Circuits and Systems* vol. 1 (2005) 21–24.

[64] R. Puri, L. Stok, J. Cohn, D. Kung, D. Pan, D. Sylvester, A. Srivastava, and S. Kulkarni, Pushing ASIC Performance in a Power Envelope, *Proc. Design Automation Conference* (2003) 788–793.

[65] B. Amelifard and M. Pedram, Optimal Design of the Power-Delivery Network for Multiple Voltage-Island System-on-Chips, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 28 (2009) 888–900.

[66] T. D. Burd, T. A. Pering, A. J. Stratakos, and R. W. Brodersen, A Dynamic Voltage Scaled Microprocessor System, *IEEE Journal of Solid-State Circuits* vol. 35 (2000) 1571–1580.

[67] K. J. Nowka, G. D. Carpenter, E. W. MacDonald, H. C. Ngo, B. C. Brock, K. I. Ishii, T. Y. Nguyen, and J. L. Burns, A 32-bit PowerPC System-on-a-Chip with Support for Dynamic Voltage Scaling and Dynamic Frequency Scaling, *IEEE Journal of Solid-State Circuits* vol. 37 (2002) 1441–1447.

[68] International Technology Roadmap of semiconductors, *http://www.itrs2.net/itrs-reports.html* (2007).

[69] Design, in International Technology Roadmap of semiconductors, *http://www.itrs2.net/itrs-reports.html* (2011).

[70] S. Borkar, Design challenges of technology scaling, *IEEE Micro* vol. 19 (1999) 23–29.

[71] S. Borkar, Digital Design for Low Power Systems, *IEEE International Electron Devices Meeting* (2005).

[72] J. Srinivasan, S. Adve, P. Bose, and J. Rivers, The impact of technology scaling on lifetime reliability, *Proceeding of IEEE International Conference on Dependable Systems and Networks* (2004) 161–170.

[73] R. Mahajan, C. Chia-pin, and G. Chrysler, Cooling a Microprocessor Chip, *Proceedings of the IEEE* vol. 94 (2006) 1476–1486.

[74] K. Lahiri, A. Raghunathan, S. Dey, and D. Panigrahi, Battery-driven system design: a new frontier in low power design, *Proceedings of Asia and South Pacific Design Automation Conference* (2002) 261–267.

[75] M. Abramovici, M. A. Breuer, and A. D. Friedman, Digital Systems Testing and Testable Design, *IEEE Press* (1990).

[76] A. Cron, C. Allsup, D. Armstrong, Reducing test costs through multisite and concurrent testing, *Tech Design Forum, http://www.techdesignforums.com/practice/technique/reduce-test-costs-multisite-concurrent-testing/* (2015)

[77] IEEE std 1500 - Standard for embedded core test *http://http://grouper.ieee.org/groups/1500/* (2005)

[78] IEEE Std. 1149.1, IEEE Standard Test Access Port and Boundary Scan Architecture. *IEEE Press* (2001).

[79] K. Chakrabarty, Optimal Test Access Architectures for System-on-a-Chip, *ACM Transactions on Design Automation of Electronic Systems* vol. 6 (2001) 26–49.

[80] S. Goel, E. J. Marinissen, A. Sehgal, and K. Chakrabarty, Testing of SoCs with Hierarchical Cores: Common Fallacies, Test Access Optimization, and Test Scheduling, *IEEE Transactions on Computers* vol. 58 (2009) 409–423.

[81] S. K. Goel and E. J. Marinissen, SoC Test Architecture Design for Efficient Utilization of Test Bandwidth, *ACM Transaction on Design Automation of Electronic Systems* vol. 8 (2003) 399–429.

[82] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, Test Wrapper and Test Access Mechanism Co-Optimization for System-on-Chip, *Proc. International Test Conference* (2001) 1023–1032.

[83] S. Koranne and V. Iyengar, On The Use of k-tuples for SoC Test Schedule Representation, *Proceedings International Test Conference* (2002) 539–548.

[84] A. Larsson, E. Larsson, K. Chakrabarty, P. Eles, and Z. Peng, Test- Architecture Optimization and Test Scheduling for SoCs with Core-Level Expansion of Compressed Test Patterns, *Proc in Design, Automation and Test in Europe* (2008) 188–193.

[85] E. Larsson and Z. Peng, An Integrated Framework for the Design and Optimization of SoC Test Solutions, *Journal on Electronic Testing: Theory and Applications* vol. 18 (2002) 385–400.

[86] V. lyengar, K. Chakrabarty, and E. J. Marinissen, Test Access Mechanism Optimization, Test Scheduling, and Tester Data Volume Reduction for System-on-Chip, *IEEE Transactions on Computers* vol. 52 (2003) 1619–1632.

[87] T. Yoneda, M. Imanishi, and H. Fujiwara, An SoC Test Scheduling Algorithm using Reconfigurable Union Wrappers, *Proc n Design, Automation Test in Europe Conference Exhibition* (2007) 1–6.

[88] X. Kavousianos, K. Chakrabarty, A. Jain, and R. Parekhji, Test Schedule Optimization for Multicore SoCs: Handling Dynamic Voltage Scaling and Multiple Voltage Islands, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 31 (2012) 1754–1766.

[89] S. Khursheed, U. Ingelsson, P. Rosinger, B. M. Al-Hashimi, P. Harrod, Bridging fault test method with adaptive power management awareness, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 27 (2008) 1117–1127.

[90] B. Kruseman, M. Heiligers, On test conditions for the detection of open defects, *Proc. of the Design, Automation and Test in Europe conference - DATE* (2006) 896–901.

[91] N. B. Z. Ali, M. Zwolinski, B. M. Al-Hashimi, and P. Harrod, Dynamic Voltage Scaling Aware Delay Fault Testing, *Proc. of IEEE European Test Symposium - ETS* (2006) 15–20.

[92] U. Ingelsson, P. Rosinger, S. S. Khursheed, B. M. Al-Hashimi, and P. Harrod, Resistive Bridging Faults DFT with Adaptive Power Management Awareness, *Proc. of Asian Test Symposium* (2007) 101–106.

[93] S. Khursheed, B. M. Al-Hashimi, S. M. Reddy, and P. Harrod, Diagnosis of Multiple-Voltage Design With Bridge Defect, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 28 (2009) 406–416.

[94] S. Khursheed, B. M. Al-Hashimi, K. Chakrabarty, and P. Harrod, Gate-sizing-based single test for bridge defects in multi-voltage designs, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 29 (2010) 1409–1421.

[95] X. Kavousianos, K. Chakrabarty, A. Jain, and R. Parekhji , Time-Division Multiplexing for Testing SoCs with DVS and Multiple Voltage Islands, *Proc. of IEEE European Test Symposium - ETS* (2012) 1–6.

[96] J. Rivoir, Lowering Cost of Test: Parallel Test or Low-Cost ATE?, *Proc. of Asian Test Symposium* (2003) 360–363.

[97] J. Rivoir, Parallel Test Reduces Cost of Test More Effectively than just a Cheap Tester, *Proc. of IEEE/CPMT/SEMI Int. Electronics Manufacturing Technology Symp.* (2004) 263–272.

[98] W. Radermacher, J. Rivoir, An Evolution to a DFT-Centric Test Paradigm that Scales with Technology Progress, *Proc. of European Test Workshop* (2001)

[99] J. Rivoir, H. Volkerink, A. Khoche, Reduced Pin-count Multisite Test with I/O Bandwidth matching, Why and How?, *2nd IEEE Workshop on Test Resource Partitioning* (2001).

[100] H. Volkerink, A. Khoche, J. Rivoir, K. Hilliges, Test Economics for Multi-Site Test with Modern Test Cost Reduction Techniques, *Proc. of VLSI Test Symposium* (2002).

[101] H. Volkerink, A. Khoche, J. Rivoir, Modern Test Techniques: Trade-offs, Synergies, and Scalable Benefits, *Special Issue of Journal of Electronic Testing: Theory and Applications, Kluwer Academic Publishers* (2002).

[102] A.C Evans, Applications of Semiconductor Test Economics, and Multi-site Testing to Lower Cost of Test, *Proceedings of IEEE International Test Conference* (1999).

[103] B. Davis, The Economics of Automatic Testing, *McGraw-Hill* (1994).

[104] H. Volkerink, A. Khoche, L. Kamas, J. Rivoir, H. Kerkhoff, Tackling Test Tradeoffs from Design, Manufacturing to Market using Economic Modeling, *Proceedings of IEEE International Test Conference* (2001).

[105] Greg Smith, The challenge of multisite test, *EDN Network - http://www.edn.com/design/test-and-measurement/4379772/The-challenge-of-multisite-test* (2006).

[106] N. Touba, Survey of Test Vector Compression Techniques, *IEEE Design Test of Computers* vol. 23 (2006) 294–303.

[107] V. Iyengar, S. Goel, E. Marinissen, and K. Chakrabarty, Test Resource Optimization for Multi-Site Testing of SOCs under ATE Memory Depth Constraints, *Proc. of Interantional Test Conference* (2002) 1159–1168.

[108] S. Goel and E. Marinissen, Optimisation of On-Chip Design-For-Test Infrastructure for Maximal Multi-Site Test Throughput, *IEE Proc. - Computers and Digital Techn.* vol. 152 (2005) 442–456.

[109] C. Su, C. Wu, A graph-based approach to power-constrained SOC test scheduling *Journal of Electron Test Theory Appl* vol. 20 (2004) 45–60.

[110] V. lyengar, K. Chakrabarty, and E. J. Marinissen, Test Access Mechanism Optimization, Test Scheduling, and Tester Data Volume Reduction for System-on-Chip, *IEEE Trans. on Computers* vol. 52 (2003) 1619–1632.

[111] A. Larsson, et. al., Test-Architecture Optimization and Test Scheduling for SoCs with Core-Level Expansion of Compressed Test Patterns, *Proc. of the Design, Automation and Test in Europe conference - DATE* (2008) 188–193.

[112] X. Kavousianos, K. Chakrabarty, A. Jain, and R. Parekhji, Test Schedule Optimization for Multicore SoCs: Handling Dynamic Voltage Scaling and Multiple Voltage Islands, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 31 (2012) 1754–1766.

[113] E. Larsson, Architecture for Integrated Test Data Compression and Abort-on-Fail Testing in a Multi-Site Environment, *IET Computers Digital Techniques* vol. 2 (2008) 275–284.

[114] EUROPRACTICE Software Service, *http://www.europractice.stfc.ac.uk*

[115] Nangate 45nm, *www.nangate.com*

[116] D. Anastasakis, R. Damiano, H. Ma, and T. Stanion, A practical and efficient method for compare point matching, *Proc. of Design Automation Conference* (2002) 305–310.

[117] D. Perry and H. Foster,*Applied Formal Verification*, McGraw-Hill, (2005).

[118] E. B. Eichelberger and T. W. Williams, A Logic Design Structure for LSI Testability, *Journal of Design Automation and Fault-Tolerant Computing* vol. 2 (1978) 165–178.

[119] L. Wang, C. Wu, X. Wen, *VLSI Test Principles and Architectures: Design for Testability* Morgan Kaufmann (2006).

[120] T. May and M. Woods, Alpha-Particle-Induced Soft Errors in Dynamic Memories, *IEEE Trans. on Electron Devices* vol. ED-26, (1979) 2–9.

[121] R. Baumann, Soft errors in advanced computer systems, *IEEE Design Test of Computers* vol. 22 (2005) 258–266.

[122] M. Ohlsson, P. Dyreklev, K. Johansson, P. Alfke, Neutron Single Event Upsets in SRAM-Based FPGAs, *Proc. of the Nuclear and Space Radiation Effects Conf.* (1998) 177–180.

[123] L. Wang, C. Stroud, N. Touba, *System-on-Chip Test Architectures: Nanometer Design for Testability*, Morgan Kaufmann, (2007).

[124] E. Hnatek, *Integrated Circuit Quality and Reliability* Mercel Dekker, (1987)

[125] E. McCluskey and F. Buelow, IC Quality and Test Transparency, *Proc. of the International Test Conference* (1988) 295–301.

[126] T. Williams and N. Brown, Defect Level as a Function of Fault Coverage, *IEEE Trans. On Computers* vol. 30 (1981) 987–988.

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

[128] T. Williams, K. Parker, Design for Testability - A Survey, *Proceedings of the IEEE* vol. 71 (1983) 98–112.

[129] E. McCluskey, *Logic Design Principles: With Emphasis on Testable Semicustom Circuits* Prentice-Hall, Englewood Cliffs, (1986).

[130] N. Touba, Survey of Test Vector Compression Techniques, *IEEE Design and Test of Computers* vol. 23 (2006) 294–303.

[131] C. Stroud, *A Designer's Guide to Built-In Self-Test* Springer (2002).

[132] R. Wadsack, Fault Modeling and Logic Simulation for CMOS and NMOS Integrated Circuits, *The Bell System Technical Journal* vol. 57 (1978) 1449–1474.

[133] M. Bushnell, V. Agrawal, *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits* Kluwer Academic Publishers (2000).

[134] J. Emmert, C. Stroud, and J. Bailey, A New Bridging Fault Model for More Accurate Fault Behavior, Proc. of the Design Automation Conference (2000) 481–485.

[135] Y. Sato, S. Hamada, T. Maeda, A. Takatori, Y. Nozuyama, S. Kajihara, Invisible Delay Quality - SDQM Model Lights Up What Could Not Be Seen, *Proc. of the International Test Conference* (2005) Paper 47.1.

[136] M. Cho, D. Pan, PEAKASO: Peak-Temperature Aware Scan-Vector Optimization, *Proc. of VTS* (2006).

[137] K. Chakravadhanula, V. Chickermane, B. Keller, P. Gallagher, S. Gregor, Test generation for state retention logic, *Proc. of the Asian Test Symposium - ATS* (2008) 237–242.

[138] Y. Zorian, Test requirements for embedded core-based systems and IEEE P1500, *Proc. of IEEE International Test Conference* (1997) 191–199.

[139] C. Wu, J. Li, C. Huang, Core-based system-on-chip testing: Challenges and opportunities, *Chinese Institute of Electrica Engineering* vol. 8, (2001) 335–353.

[140] IEEE Std. 1450.6βε̇ő2001, *Core Test Language (CTL)*, IEEE Press, (2001).

[141] Automatic test equipment from Teradyne Inc., *http://www.coroflot.com/babkesm/portfolio1*

[142] P. Harrod, Testing reusable IP - a case study, *Proc. of IEEE International Test Conference - ITC* (1999) 493.

[143] J. Aerts, E. Marinissen, Scan chain design for test time reduction in core-based ICs, *Proc. of IEEE International Test Conference - ITC* (1998) 448–457.

[144] T. Waayers, R. Morren, R. Grandi, Definition of a robust modular SOC test architecture: Resurrection of the single TAM daisy-chain, *Proc. of IEEE International Test Conference - ITC* (2005) Paper 25.3, 10

[145] V. Immaneni, S. Raman, Direct access test scheme-design of block and core cells for embedded asics, *Proc. of IEEE International Test Conference - ITC* (1990) 488–492.

[146] P. Varma, S. Bhatia, A structured test re-use methodology for core-based system chips, *Proceedings of IEEE International Test Conference - ITC* (1998) 294–302.

[147] V. Iyengar, K. Chakrabarty, E. Marinissen, Integrated wrapper/TAM co-optimization, constraint-driven test scheduling, and tester data volume reduction for SOCs, *Proc. Design Automation Conference* (2002) 685–690.

[148] E. Larsson, Z. Peng, An estimation-based technique for test scheduling, *Proc. of Electronic Circuits and Systems Conference* (1999).

[149] E. Larsson, Z. Peng, A technique for test infrastructure design and test scheduling, *Proc. of IEEE Design and Diagnostics of Electronic Circuits and Systems Workshop - DDECS* (2000).

[150] K. Chakrabarty, Design of system-on-a-chip test access architectures under place-and-route and power constraints, *Proc. of ACM/IEEE Design Automation Conference - DAC* (2000) 432–437.

[151] S. Koranne, A novel reconfigurable wrapper for testing of embedded core-based SOCs and its associated scheduling algorithm. *Journal of Electron. Test Theory Appl.* vol. 18 415–434.

[152] T. Yoneda, H. Fujiwara, Design for consecutive testability of system-on-a-chip with built-in self testable cores, *Journal of Electron. Test Theory Appl.* vol. 18 (2002) 487–501.

[153] T. Yoneda, K. Masuda, H. Fujiwara, Power-constrained test scheduling for multi-clock domain SOCs. *Proc. of Design, Automation, and Test in Europe - DATE* (2006) 297–302.

[154] Z. He, Z. Peng, P. Eles, Multi-Temperature Testing for Core-based System-on-Chip, *Proc. of Design, Automation, and Test in Europe - DATE* (2010).

[155] D. Bild et al, Temperature-Aware Test Scheduling for Multiprocessor Systems-On-Chip, *IEEE/ACM International Conference on Computer Aided Design - ICCAD* (2008).

[156] Z. He, Z. Peng, P. Eles, Simulation-Driven Thermal-Safe Test Time Minimization for System-on-Chip, *Proc. of the Asian Test Symposium - ATS* (2008)

[157] C. Yao, K. Saluja, P. Ramanathan, Partition Based SoC Test Scheduling with Thermal and Power Constraints under Deep Submicron Technologies, *Proc. of the Asian Test Symposium - ATS* (2009).

[158] N. Aghaee, Z. He, Z. Peng, P. Eles, βἐὢTemperature-Aware SoC Test Scheduling Considering Inter-Chip Process Variation, *Proc. of the Asian Test Symposium - ATS* (2010)

[159] C. Yao, K. Saluja, P. Ramanathan, Power and Thermal Constrained Test Scheduling, *Proceedings of IEEE International Test Conference - ITC* (2009).

[160] P. Rosinger, B. Al-Hashimi, K. Chakrabarty, Rapid generation of thermal-safe test schedules, *Proc. of Design, Automation, and Test in Europe - DATE* (2005).

[161] C. Liu et al, Thermal-Aware Test Scheduling and Hot Spot Temperature Minimization for Core-Based Systems, DFT (2005)

[162] S. Ravi, Power-aware Test: Challenges and Solutions, *Proceedings of IEEE International Test Conference - ITC* (2007).

[163] H. Hao, E. McCluskey, Very-low-voltage testing for weak CMOS logic ICs, *Proceedings of IEEE International Test Conference - ITC* (1993) 275–284.

[164] S. Khursheed, B. Al-Hashimi, S. Reddy, P. Harrod, Diagnosis of multiple-voltage design with bridge defect, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 28 (2009) 406–416.

[165] W. Maly, Realistic fault modeling for VLSI testing, IEEE Design Automation Conference (1987) 173–180.

[166] P. Engelke, I. Polian, M. Renovell, B. Becker, Simulating resistive bridging and stuck-at faults, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol 25 (2006) 2181–2192.

[167] V. Sar-Dessai, D. Walker, Resistive bridge fault modeling, simulation and test generation, *Proceedings of IEEE International Test Conference - ITC* (1999) 596–605.

[168] T. Maeda, K. Kinoshita, Precise test generation for resistive bridging faults of CMOS combinational circuits *Proceedings of the International Test Conference* (2000) 510–519.

[169] P. Engelke P, I. Polian, M. Renovell, B. Seshadri, B. Becker, The pros and cons of very-low-voltage testing: an analysis based on resistive bridging faults, *Proceedings of the VLSI test symposium* (2004) 171–178.

[170] R. Montanes,J. de Gyvez, P. Volf, Resistance characterization for weak open defects, *IEEE Design and Test of Computers* vol. 19 (2002) 18–26.

[171] B. Kruseman, A. Majhi, G. Gronthoud, S. Eichenberger, On hazard-free patterns for fine delay fault testing, *Proceedings of IEEE International Test Conference - ITC* (2004) 213–222.

[172] J. Chang-Y, E. McCluskey, Detecting delay flaws by very-low-voltage testing, *Proceedings of IEEE International Test Conference - ITC* (1996) 367–376.

[173] J. T.-Y. Chang, E.J. McCluskey, Quantitative Analysis of Very-Low-Voltage Testing, *Proc. of VLSI Test Symposium* (1996) 332–337.

[174] Horowitz, M., T. Indermaur, and R. Gonzalez, Low Power Digital Design, *Symp. on Low Power Electronics* (1994) 8–11.

[175] Wagner, K., and E.J. McCluskey, Effect of Supply Voltage on Circuit Propagation Delay and Test Application, *Center for Reliable Computing Technical Report - Stanford University* (1986) 86–21.

[176] S. M. Martin, K. Flautner, T. Mudge, and D. Blaauw, Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads, *Proc. of IEEE/ACM International Conference on Computer Aided Design* (2002) 721–725.

[177] S. Khursheed, B. Al-Hashimi, P. Harrod, Test cost reduction for multiple-voltage designs with bridge defects through gate sizing *Proc. of Design, Automation, and Test in Europe - DATE* (2009).

[178] C. Shi, R. Kapur, How power-aware test improves reliability and yield, *http://www.eedesign.com/showArticlejhtml?articleID=47208594* (2004).

[179] A. Colle, S. Ramnath, M. Hirech, S. Chebiyam, Power and design for test: a design automation perspective, *Journal of Low Power Electronics* vol. 1 (2005) 73–84.

[180] V. Chickermane, P. Gallagher, J. Sage, P. Yuan, K. Chakravadhanula, A power-aware test methodology for multi-supply multi-voltage designs, *Proceedings of IEEE International Test Conference - ITC* (2008) 1–10.

[181] V. Devanathan, C. Ravikumar, R. Mehrotra, V. Kamakoti, PMScan: A power-managed scan for simultaneous reduction of dynamic and leakage power during scan test, *Proceedings of IEEE International Test Conference - ITC* (2007) 1–9.

[182] K. Skadron, M. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy and D. Tarjan, Temperature-aware microarchitecture: Modeling and implementation, *ACM Transactions on Architecture and Code Optimization* vol. 1 (2004) 94–125.

[183] H. Yan, Q. Zhou, and X. Hong, Efficient thermal aware placement approach integrated with 3D DCT placement algorithm*Proceedings of International Symposium on Quality Electronic Design* (2008) 289–292.

[184] W. Hung, C. Addo-Quaye, T. Theocharides, Y. Xie, N. Vijakrishnan, and M. Irwin, Thermal-aware IP virtualization and placement for networks-on-chip architecture, *Proceedings of International Conference on Computer Design - ICCD* (2004) 430–437.

[185] Abadir MS, Breuer MA, Test schedules for VLSI circuits having built-in test hardware, *IEEE Trans Comput - http://dx.doi.org/10.1109/TC.1986.1676771* vol. 35 (1986) 361–367.

[186] Craig GL, Kime CR, Saluja KK, Test scheduling and control for VLSI built-in self-test, *IEEE Trans Comput - http://dx.doi.org/10.1109/12.2260* vol. 37 (1988) 1099–1109.

[187] Marinissen EJ, Iyengar V, Chakrabarty K, A set of benchmarks for modular testing of SOCs, *Proceedings of IEEE International Test Conference* (2002)519–528.

[188] Samii S, Larsson E, Chakrabarty K, Peng Z, Cycle-accurate test power modelling and its application to SOC test scheduling, *Proceedings of IEEE International Test Conference* (2006) 1–10.

[189] Y. Zorian, A distributed BIST control scheme for complex VLSI devices, *Proceedings of VLSI Test Symposium* (1993) 4–9.

[190] R. Chou, K. Saluja, Y. Agrawal, Scheduling tests for VLSI systems under power constraints, *IEEE Transactions on VLSI Systems* vol 5 (1997) 175–185.

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

[192] Pouget J, Larsson E, Peng Z, SOC test time minimization under multiple constraints, *Proceedings of Asian Test symposium* (2003) 312–317.

[193] Pouget J, Larsson E, Peng Z, Flottes M, Rouzeyre B, An efficient approach to SOC wrapper design, TAM configuration and test scheduling, *Proceedings of IEEE European Test Symposium* (2003) 51–56.

[194] D. Zhao, S. Upadhyaya, Power constrained test scheduling with dynamically varied TAM, *Proceedings of IEEE VLSI Test Symposium* (2003), 273.

[195] Muresan V, Wang X, Muresan V, Vladutiu M, A comparison of classical scheduling approaches in power-constrained block-test scheduling, *Proceedings of IEEE International Test Conference* (2000) 882.

[196] Muresan V, Wang X, Muresan V, Vladutiu M, Greedy tree growing heuristics on block-test scheduling under power constraints, *J. Electron Test Theory Appl* vol. 20 (2004) 61–78.

[197] Ravikumar CP, Chandra G, Verma A, Simultaneous module selection and scheduling for power-constrained testing of core based systems, *Proceedings of International Conference on VLSI design* (2000) 462.

[198] V. Dabholkar, S. Chakravarty, I. Pomeranz et al, Techniques for Minimizing Power Dissipation in Scan and Combinational Circuits during Test Application, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* Vol. 17 (1998) 1325–1333.

[199] P. Girard, Survey of Low-Power Testing of VLSI Circuits, *IEEE Design and Test of Computers* Vol. 19 (2002) 82–92.

[200] N. Nicolici, and X. Wen, Embedded Tutorial on Low Power Test, *In Proc. IEEE European Test Symposium* (2007) 202–210.

[201] Muzaffer O. Simsir and Niraj K. Jha, βἐὢThermal Characterization of BIST, Scan Design and Sequential Test Methodologiesβἐὢ, ITC 2009, Paper 14.1.

[202] L. Chunsheng, K. Veeraraghavant, V. Iyengar, Thermal-Aware Test Scheduling and Hot Spot Temperature Minimization for Core-Based Systems, *Proceedings of IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems* (2005).

[203] Z. He, Z. Peng, P. Eles, Simulation-Driven Thermal-Safe Test Time Minimization for System-on-Chip *In Proc. of Asian Test Symposium* (2008).

[204] HotSpot Tool Suite, *http://lava.cs.virginia.edu/HotSpot/index.htm*

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

[206] A. Chandra and R. Kapur, Bounded adjacent fill for low capture power scan test-ing, *Proc. of VTS* (2008) 131–138.

[207] S. Remersaro, X. Lin, S. Reddy, I. Pomeranz, and J. Rajski, Scan-based tests with low switching activity, *J. IEEE Design Test of Computers* vol. 24 (2007) 268–275.

[208] S. Remersaro, X. Lin, Z. Zhang, S. Reddy, I. Pomeranz, and J. Rajski, Preferred Fill: A scalable method to reduce capture power for scan based designs, *Proc. of International Test Conference* (2006) 1–10.

[209] X. Kavousianos, V. Tenentes, K. Chakrabarty, and E. Kalligeros, Defect-oriented lfsr reseeding to target unmodeled defects using stuck-at test sets, *IEEE Trans. on VLSI Systems* vol. 19 (2011) 2330–βἐὂ2335.

[210] D. Czysz, G. Mrugalski, N. Mukherjee, J. Rajski, P. Szczerbicki, and J. Tyszer, Deterministic clustering of incompatible test cubes for higher power-aware edt compression, *IEEE Trans. on CAD* vol. 30 (2011) 1225–1238.

[211] D. Czysz, G. Mrugalski, J. Rajski, and J. Tyszer, Low-power test data application in edt environment through decompressor freeze, *IEEE Trans. on CAD* vol. 27 (2008) 1278–1290.

[212] P. Girard, L. Guiller, C. Landrault, and S. Pravossoudovitch, A test vector inhibiting technique for low energy bist design, *Proc. of IEEE VTS* (1999) 407–412.

[213] J. Lee and N. Touba, Low power test data compression based on lfsr reseeding, *Proc. in IEEE ICCD* (2004) 180–185.

[214] X. Wen, Y. Yamashita, S. Morishima, S. Kajihara, L.-T. Wang, K. Saluja, and K. Kinoshita, Low-capture-power test generation for scan-based at-speed testing, *Proc. of International Test Conference* (2005) 1028.

[215] X. Wen, K. Miyase, T. Suzuki, Y. Yamato, S. Kajihara, L.-T. Wang, and K. Saluja, A highly-guided x-filling method for effective low-capture-power scan test generation, *Proc. of ICCD* (2006) 251–258.

[216] S. Kajihara, K. Ishida, and K. Miyase, Test vector modification for power reduction during scan testing, Proc. of IEEE VTS (2002) 160–165.

[217] J. Li, Q. Xu, Y. Hu, and X. Li, ifill: An impact-oriented x-filling method for shift- and capture-power reduction in at-speed scan-based testing, *Proc. of DATE* (2008) 1184–1189.

[218] W. Li, S. Reddy, and I. Pomeranz, On reducing peak current and power during test, *Proc. of IEEE Computer Society Annual Symposium on VLSI* (2005) 156–161.

[219] S. Balatsouka, V. Tenentes, X. Kavousianos, and K. Chakrabarty, Defect aware x-filling for low-power scan testing, Proc. of DATE (2010) 873–878.

[220] L.-T. Wang, K. Abdel-Hafez, X. Wen, B. Sheu, S. Wu, S.-H. Lin, and M.-T. Chang, UltraScan: Using Time-Division Demultiplexing/Multiplexing (TDDM/TDM) with Virtual-Scan for Test Cost Reduction, *Proc. of IEEE International Test Conference* (2005) 953.

[221] B. Chazelle, The Bottomn-Left Bin-Packing Heuristic: An Efficient Implementation, *Proc. of IEEE Transactions on Computers* vol. C-32 (1983) 697–707.

[222] X. Kavousianos, K. Chakrabarty, A. Jain, and R. Parekhji, Time-Division Multiplexing for Testing SoCs with DVS and Multiple Voltage Islands, *Tech. Rep. Duke University, http://hdl.handle.net/10161/5120* (2012).

[223] E. Hopper and B. C. H. Turton, An Empirical Investigation of Metaheuristic and Heuristic Algorithms for a 2D Packing Problem, *European Journal of Operational Research* vol. 128 (2000) 34–57.

[224] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, Numerical Recipes: The Art of Scientific Computing , *3rd ed. New York: Cambridge University Press* (2007) ch. 10.12: Simulated Annealing Methods.

[225] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Optimization by Simulated Annealing, *Science 220* (1983) 671–680.

[226] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee, Embedded Deterministic Test, *IEEE Trans. on CAD of Integr. Circ. and Systems* vol. 23 (2004) 776–792.

[227] Evaluation Enginnering, *http://www.evaluationengineering.com*

[228] H. Sharp, Cardinality of Finite Topologies, *J. Combinatorial Theory* vol. 5 (1968) 82–86.

[229] Z. Wang and K. Chakrabarty, Test-quality/cost optimization using output-deviation-based reordering of test patterns, *IEEE Trans. on CAD* vol. 27 (2008) 352–365.

[230] Z. Wang, H. Fang, K. Chakrabarty, and M. Bienek, Deviation-based lfsr reseeding for test-data compression, *IEEE Trans. on CAD* vol. 28 (2009) 259–271.

[231] A. Jain, R. Parekhji, *Texas Instruments, https://www.ti.com/*

[232] IWLS, *http://iwls.org/iwls2005/benchmarks.html.*

[233] A. Agresti, Analysis of Ordinal Categorical Data, *New York: John Wiley & Sons* (2010).

[234] K. Chakrabarty, Test Scheduling for Core-Based Systems Using Mixed-Integer Linear Programming, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol. 19 (2000) 1163–1174.

[235] FICO Xpress Optimization Suite, *http://www.fico.com /en/Products/DMTools/Pages/FICO-Xpress-Optimization- Suite.aspx*

[236] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, On Using Rectangle Packing for SOC Wrapper / TAM Co- Optimization, *Proceedings of VLSI Test Symposium* (2002) 253–258.

[237] P Georgiou, F. Vartziotis, X. Kavousianos, K. Chakrabarty, Two-Dimensional Time-Division Multiplexing for 3D-SoCs, *Proceedings of IEEE European Test Symposium* (2016) 129–135.

Appendices

Appendix A

SoC configuration files in the simulation framework

A.1 Example of industrial SoC test configuration file

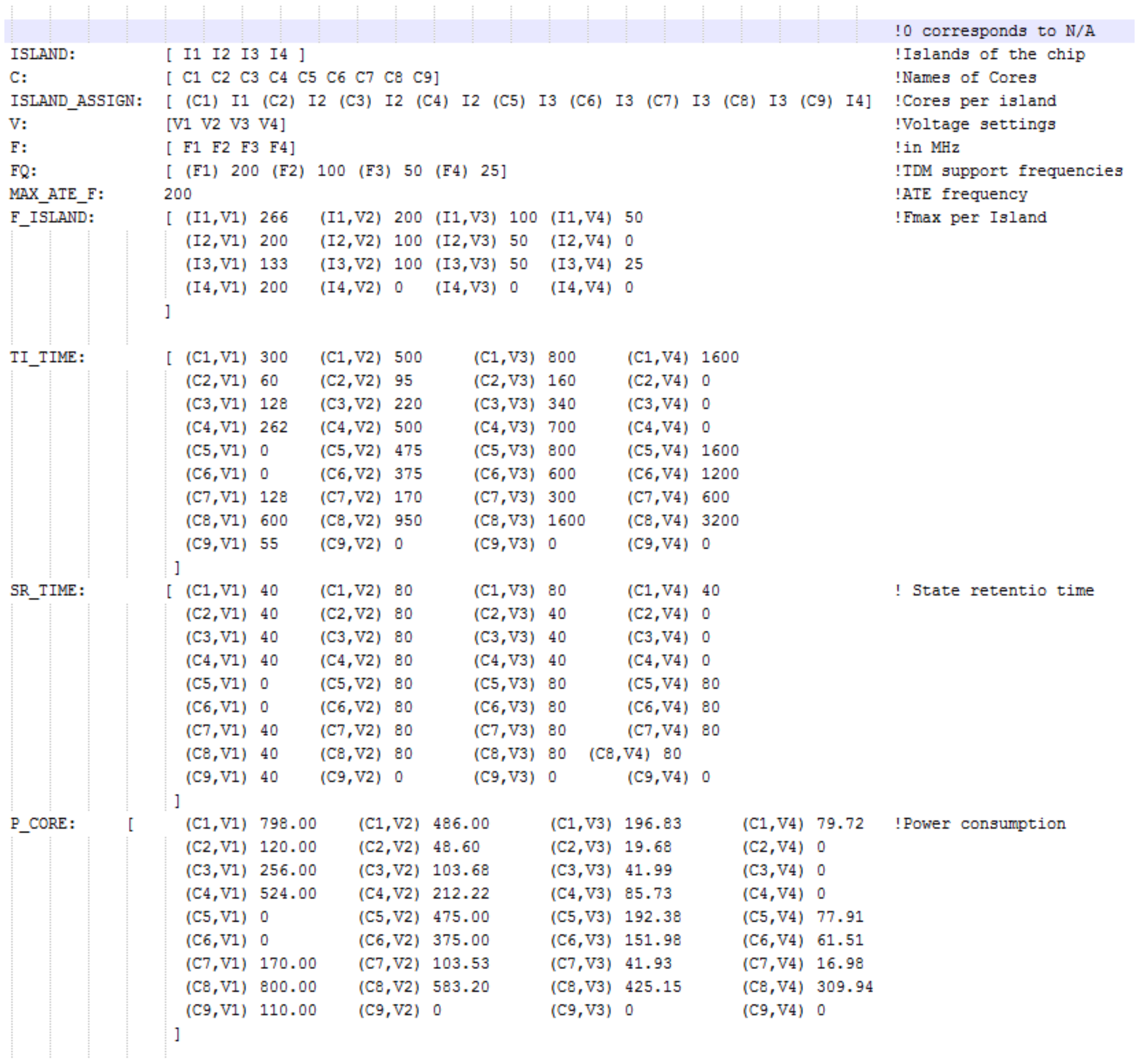
Fig. A.1 shows the configuration file of the industrial SoC, SoC-A. Such files are used as input to the *SoCTDMScheduler* tool to define the SoC under test. The extension of the file is *.soc*.

A.2 Example of TAM configuration file

Fig. A.2 shows an example TAM configuration file that corresponds to the SoC-A. Such files are used as input to the *SoCTDMScheduler* tool to define the TAM of the SoC under test. The extension of the file is *.tam*.

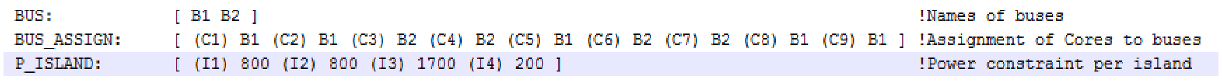
A.3 Example of benchmark core configuration file

Fig. A.3 shows the configuration file of the *Ethernet* core, that belongs to the IWLS suite. The configurations of cores from the IWLS and the ISCAS suite have been derived using the design flow for benchmark cores, described in section 4.2. These files are incorporated in the *SoCTDMScheduler* and can be used directly for the creation of artificial multi-core SoCs.



```
                                                                                    !0 corresponds to N/A
ISLAND:         [ I1 I2 I3 I4 ]                                                     !Islands of the chip
C:              [ C1 C2 C3 C4 C5 C6 C7 C8 C9]                                       !Names of Cores
ISLAND_ASSIGN:  [ (C1) I1 (C2) I2 (C3) I2 (C4) I2 (C5) I3 (C6) I3 (C7) I3 (C8) I3 (C9) I4]  !Cores per island
V:              [V1 V2 V3 V4]                                                       !Voltage settings
F:              [ F1 F2 F3 F4]                                                      !in MHz
FQ:             [ (F1) 200 (F2) 100 (F3) 50 (F4) 25]                                !TDM support frequencies
MAX_ATE_F:      200                                                                 !ATE frequency
F_ISLAND:       [ (I1,V1) 266   (I1,V2) 200 (I1,V3) 100 (I1,V4) 50                  !Fmax per Island
                  (I2,V1) 200   (I2,V2) 100 (I2,V3) 50  (I2,V4) 0
                  (I3,V1) 133   (I3,V2) 100 (I3,V3) 50  (I3,V4) 25
                  (I4,V1) 200   (I4,V2) 0   (I4,V3) 0   (I4,V4) 0
                ]

TI_TIME:        [ (C1,V1) 300   (C1,V2) 500     (C1,V3) 800     (C1,V4) 1600
                  (C2,V1) 60    (C2,V2) 95      (C2,V3) 160     (C2,V4) 0
                  (C3,V1) 128   (C3,V2) 220     (C3,V3) 340     (C3,V4) 0
                  (C4,V1) 262   (C4,V2) 500     (C4,V3) 700     (C4,V4) 0
                  (C5,V1) 0     (C5,V2) 475     (C5,V3) 800     (C5,V4) 1600
                  (C6,V1) 0     (C6,V2) 375     (C6,V3) 600     (C6,V4) 1200
                  (C7,V1) 128   (C7,V2) 170     (C7,V3) 300     (C7,V4) 600
                  (C8,V1) 600   (C8,V2) 950     (C8,V3) 1600    (C8,V4) 3200
                  (C9,V1) 55    (C9,V2) 0       (C9,V3) 0       (C9,V4) 0
                 ]
SR_TIME:        [ (C1,V1) 40    (C1,V2) 80      (C1,V3) 80      (C1,V4) 40          ! State retentio time
                  (C2,V1) 40    (C2,V2) 80      (C2,V3) 40      (C2,V4) 0
                  (C3,V1) 40    (C3,V2) 80      (C3,V3) 40      (C3,V4) 0
                  (C4,V1) 40    (C4,V2) 80      (C4,V3) 40      (C4,V4) 0
                  (C5,V1) 0     (C5,V2) 80      (C5,V3) 80      (C5,V4) 80
                  (C6,V1) 0     (C6,V2) 80      (C6,V3) 80      (C6,V4) 80
                  (C7,V1) 40    (C7,V2) 80      (C7,V3) 80      (C7,V4) 80
                  (C8,V1) 40    (C8,V2) 80      (C8,V3) 80  (C8,V4) 80
                  (C9,V1) 40    (C9,V2) 0       (C9,V3) 0       (C9,V4) 0
                 ]
P_CORE:     [     (C1,V1) 798.00    (C1,V2) 486.00      (C1,V3) 196.83      (C1,V4) 79.72   !Power consumption
                  (C2,V1) 120.00    (C2,V2) 48.60       (C2,V3) 19.68       (C2,V4) 0
                  (C3,V1) 256.00    (C3,V2) 103.68      (C3,V3) 41.99       (C3,V4) 0
                  (C4,V1) 524.00    (C4,V2) 212.22      (C4,V3) 85.73       (C4,V4) 0
                  (C5,V1) 0         (C5,V2) 475.00      (C5,V3) 192.38      (C5,V4) 77.91
                  (C6,V1) 0         (C6,V2) 375.00      (C6,V3) 151.98      (C6,V4) 61.51
                  (C7,V1) 170.00    (C7,V2) 103.53      (C7,V3) 41.93       (C7,V4) 16.98
                  (C8,V1) 800.00    (C8,V2) 583.20      (C8,V3) 425.15      (C8,V4) 309.94
                  (C9,V1) 110.00    (C9,V2) 0           (C9,V3) 0           (C9,V4) 0
                 ]
```

Figure A.1: SoC configuration file.

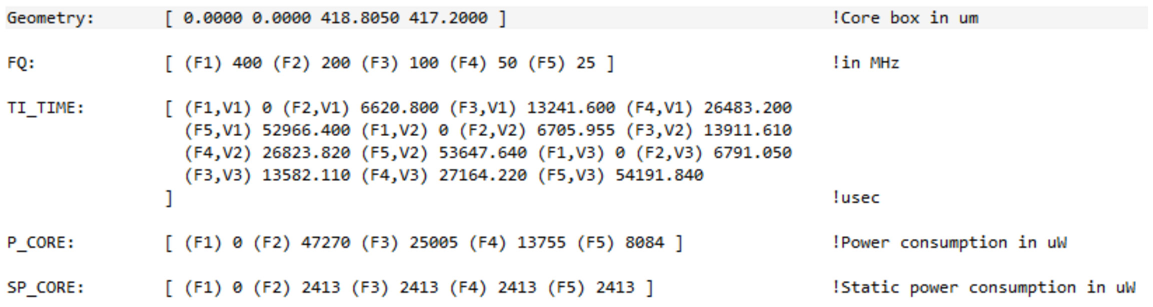


```
BUS:            [ B1 B2 ]                                                           !Names of buses
BUS_ASSIGN:     [ (C1) B1 (C2) B1 (C3) B2 (C4) B2 (C5) B1 (C6) B2 (C7) B2 (C8) B1 (C9) B1 ] !Assignment of Cores to buses
P_ISLAND:       [ (I1) 800 (I2) 800 (I3) 1700 (I4) 200 ]                            !Power constraint per island
```

Figure A.2: TAM configuration file.

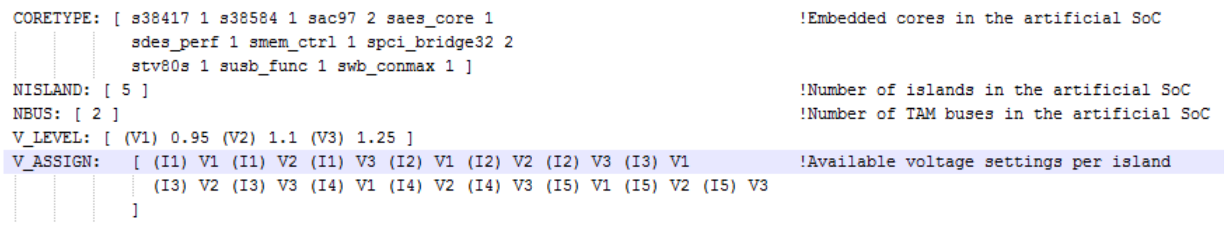
A.4 Example of configuration file for the creation of an artificial SoC

Fig. A.4 shows an example of a configuration file for the creation of an artificial SoC. It is used by the *SoCTDMScheduler* to produce the floor-plan, the SoC and the TAM configuration files of the artificial SoC. The extension of the file is *.init2D*.



```
Geometry:        [ 0.0000 0.0000 418.8050 417.2000 ]                              !Core box in um

FQ:              [ (F1) 400 (F2) 200 (F3) 100 (F4) 50 (F5) 25 ]                   !in MHz

TI_TIME:         [ (F1,V1) 0 (F2,V1) 6620.800 (F3,V1) 13241.600 (F4,V1) 26483.200
                   (F5,V1) 52966.400 (F1,V2) 0 (F2,V2) 6705.955 (F3,V2) 13911.610
                   (F4,V2) 26823.820 (F5,V2) 53647.640 (F1,V3) 0 (F2,V3) 6791.050
                   (F3,V3) 13582.110 (F4,V3) 27164.220 (F5,V3) 54191.840
                 ]                                                                !usec

P_CORE:          [ (F1) 0 (F2) 47270 (F3) 25005 (F4) 13755 (F5) 8084 ]            !Power consumption in uW

SP_CORE:         [ (F1) 0 (F2) 2413 (F3) 2413 (F4) 2413 (F5) 2413 ]               !Static power consumption in uW
```

Figure A.3: Benchmark core configuration file (*Ethernet*).



```
CORETYPE: [ s38417 1 s38584 1 sac97 2 saes_core 1                         !Embedded cores in the artificial SoC
            sdes_perf 1 smem_ctrl 1 spci_bridge32 2
            stv80s 1 susb_func 1 swb_conmax 1 ]
NISLAND: [ 5 ]                                                            !Number of islands in the artificial SoC
NBUS: [ 2 ]                                                               !Number of TAM buses in the artificial SoC
V_LEVEL: [ (V1) 0.95 (V2) 1.1 (V3) 1.25 ]
V_ASSIGN:   [ (I1) V1 (I1) V2 (I1) V3 (I2) V1 (I2) V2 (I2) V3 (I3) V1     !Available voltage settings per island
              (I3) V2 (I3) V3 (I4) V1 (I4) V2 (I4) V3 (I5) V1 (I5) V2 (I5) V3
            ]
```

Figure A.4: Artificial's SoC configuration file.

Appendix B

SoCTDMScheduler

B.1 Introduction

The *SoCTDMScheduler* aims to provide small, medium, large and very large multi-core SoCs with efficient test schedules that are derived using TDM methods. It is addressed to two user groups. First, to researchers that are activated in the SoC testing area and need an experimental and evaluation framework for SoC test scheduling methods. Second, to test designers that seek new, cost-effective ways to test SoCs. A description of its functionality is given below. The tool is available on-line (http://sonetto.teiep.gr/soctdmscheduler/).

B.2 SoCTDMScheduler's functionality

B.2.1 TDM-based test scheduling functionality

Fig. B.1 shows that users, via a graphical user interface (GUI), can:

- Select an SoC configuration file. As shown in section A.1, this file describes the SoC design and the data required to perform test scheduling.
- Select a TAM configuration file. As shown in section A.2, this file describes the TAM configuration of the selected SoC.
- Recalculate test times in an SoC configuration file, due to changes in a TAM bus width,
- Create an artificial SoC design and test data using random data,
- Create an artificial SoC design using data that are extracted from one or more SoC configuration files,
- Create an artificial SoC design using benchmark cores,

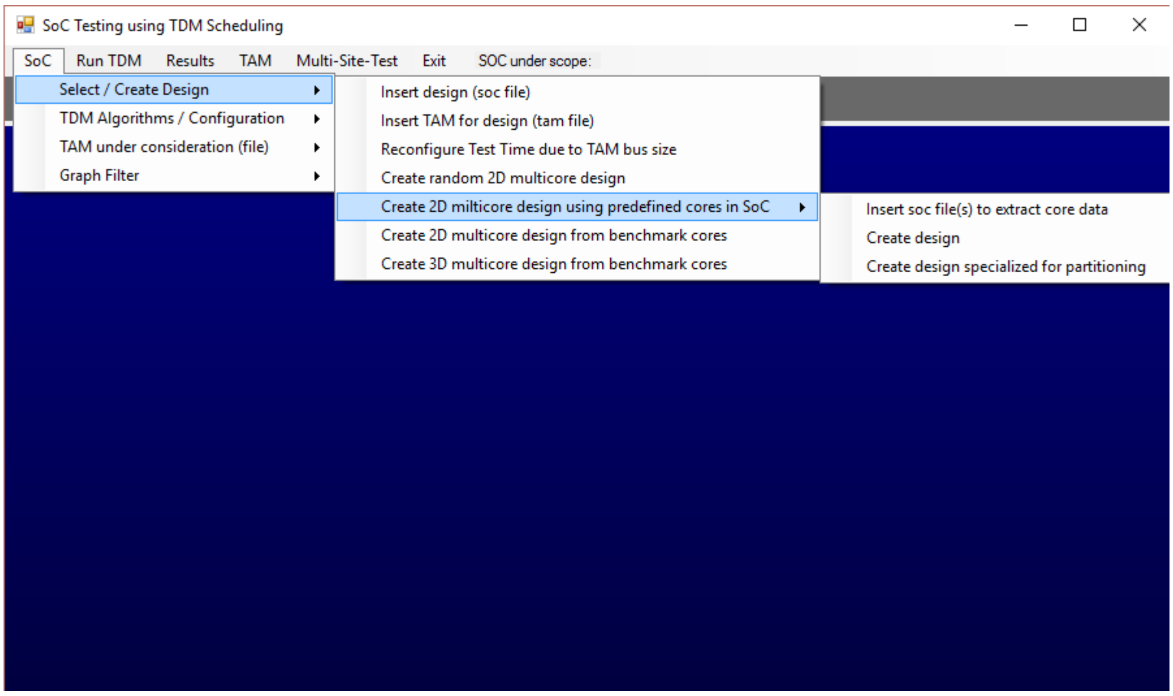

SoC Testing using TDM Scheduling
SoC
Run TDM
Results
TAM
Multi-Site-Test
Exit
SOC under scope:
Select / Create Design
TDM Algorithms / Configuration
TAM under consideration (file)
Graph Filter
Insert design (soc file)
Insert TAM for design (tam file)
Reconfigure Test Time due to TAM bus size
Create random 2D multicore design
Create 2D miltcore design using predefined cores in SoC
Create 2D multicore design from benchmark cores
Create 3D multicore design from benchmark cores
Insert soc file(s) to extract core data
Create design
Create design specialized for partitioning

Figure B.1: Selection / creation of SoC designs in the $SoCTDMScheduler$.

- Create a large partitioned artificial SoC design for evaluating the corresponding TAM optimization technique.

In addition, the users, via GUI, can select / configure the algorithm that will be used for the test scheduling process, as shown in Fig. B.2. The SA-RP and the Greedy algorithms are supported.

The test scheduling process can be configured according to the user's needs, as it is illustrated in Fig. B.3. Specifically, users are able to:

- Select among two implementations of the Greedy technique, the normal and the enhanced implementation. Shortly, we refer that in the enhanced implementation, the rectangle packing algorithm gives higher priority to the placement of the large rectangles in the bin.

- Define the parameters CR, T_{init} and r of the SA algorithm. In addition, user can select to run the SA-RP algorithm a number of times, using as first solution in every next run, the output of the previous one.

- Define the maximum height of the bin (for computational reasons) and select among a variety of rectangle packing implementations, such as $best-area-fit$, $bottom-left-rule$, $best-long-side-fit$, $best-short-side-fit$. Details about each method can be easily found in bibliography.

- Decide if the power consumption of an SoC during test should be estimated and how. Specifically, the $SoCTDMScheduler$ uses the derived test schedule of an SoC

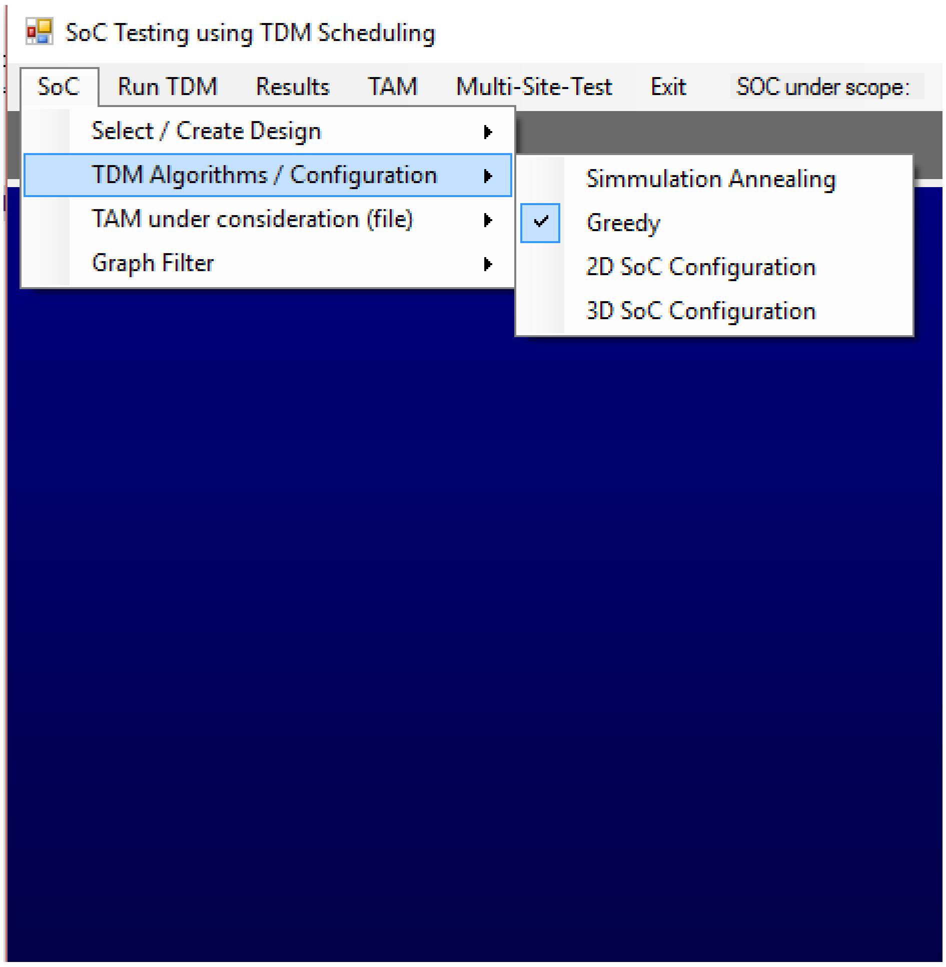


Figure B.2: Test scheduling algorithms in the $SoCTDMScheduler$.

in order to capture in user-defined time intervals the power consumption. The calculated values are written in a file called *ptrace*. The *ptrace* file along with the floor-plan file of the SoC are used as input in the *hotspot* tool [204], to derive the thermal profile of the SoC during test. Note that the $sampling_intvl$ parameter of the *hotspot* tool should be equal to the above mentioned user-defined time interval in order to acquire correct results. Moreover, the calculation of SoC's power consumption during test can be implemented using either the average power of the embedded cores in a specified frequency or using a specialized file that describes the test power consumption of the embedded cores in specific time intervals, that normally corresponds to some hundreds clock cycles. Naturally, in the second case the results are more reliable. Such specialized power consumption files have been created for the benchmark cores.

- Decide if at the beginning or the end of a specific test, an additional time is needed for SoC and test configuration purposes. The required time data are fully described in the SoC configuration file (section A.1) and the user should decide if they are going to be used in the calculations or not.

- Decide if intermediate results of the SA-RP process, namely sub-optimal test schedules, should be derived for evaluation purposes or not.

- Produce the values of the TDM registers through out the test schedule. These data are useful in real implementations of the TDM test method.

- Decide upon the core wrapper impact. Specifically, when the user activates this choice, the *SoCTDMScheduler* reads in the SoC configuration file the user-defined WPP widths of the wrapped embedded cores. Normally, the test time of each core in the SoC configuration file is given based on a predefined WPP width (the default WPP width is 8) that might be different from the one that user defines. In this case the test times of the embedded cores are recalculated automatically by the *SoCTDMScheduler*.

- Decide if the derived test schedule will be power-aware or not. If user selects a power-aware test schedule, then it can select between a power constraint for the whole SoC or a power constraint per voltage island. The power constraint can be defined via the GUI or in the TAM configuration file. Note that the static power of the inactive cores during a test period is taken into account.

Fig. B.4 shows how users, via GUI, can execute the test scheduling process. The SA-RP algorithm has been implemented in an asynchronous mode, so that it can be interrupted at an time. In addition, when the test schedule process ends, the user can proceed directly to the execution of the *hotspot* tool, as shown in Fig. B.5, and select the number of the *hotspot* repetitive executions.

A variety of graphs can be produced for a test scheduling process. Fig. B.5 shows that users, via GUI, are able to create the graph of:

- A test schedule,

- SoC power consumption vs test time,

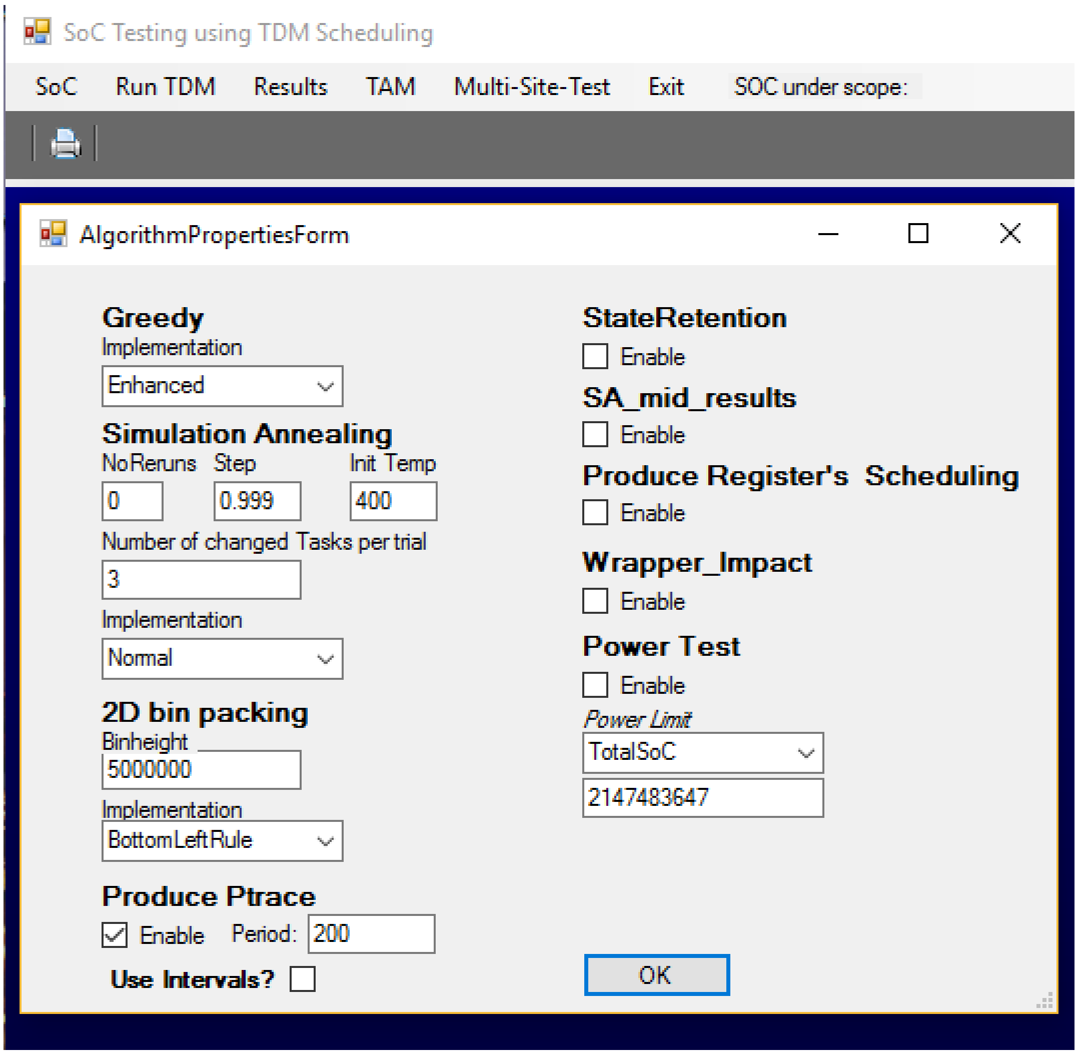
- SoC temperature vs test time,

- Selected embedded core's power consumption vs test time,

- Selected embedded core's temperature vs test time,

- SoC floor-plan. In this case, users can select to load the thermal and the power profile of the SoC too. Then, *SoCTDMScheduler* is able to create a video-like output that shows the development of the temperature in the embedded cores during test. The transitions in the temperature of each core are depicted as colour transitions.

Example SoC-related graphs are shown in Fig. 4.7. Fig. B.7 shows a graph-properties window that allows users to fully customize the graph presentation.


SoC Testing using TDM Scheduling

SoC | Run TDM | Results | TAM | Multi-Site-Test | Exit | SOC under scope:

AlgorithmPropertiesForm

Greedy
Implementation: Enhanced

Simulation Annealing
NoReruns: 0 | Step: 0.999 | Init Temp: 400
Number of changed Tasks per trial: 3
Implementation: Normal

2D bin packing
Binheight: 5000000
Implementation: BottomLeftRule

Produce Ptrace
☑ Enable | Period: 200

Use Intervals? ☐

StateRetention
☐ Enable

SA_mid_results
☐ Enable

Produce Register's Scheduling
☐ Enable

Wrapper_Impact
☐ Enable

Power Test
☐ Enable
Power Limit: TotalSoC
2147483647

OK

Figure B.3: Test scheduling configuration in the *SoCTDMScheduler*.

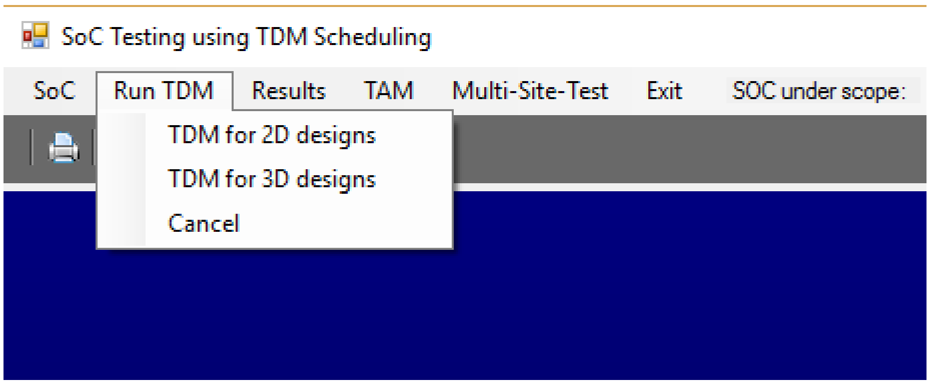


Figure B.4: Execution of the test scheduling process in the *SoCTDMScheduler*.

B.2.2 Functionality for TAM optimization

The *SoCTDMScheduler* provides with a complete set of tools for TAM optimization. These tools are accessible via the menu item *TAM*, as shown in Fig. B.8. As it is illus-

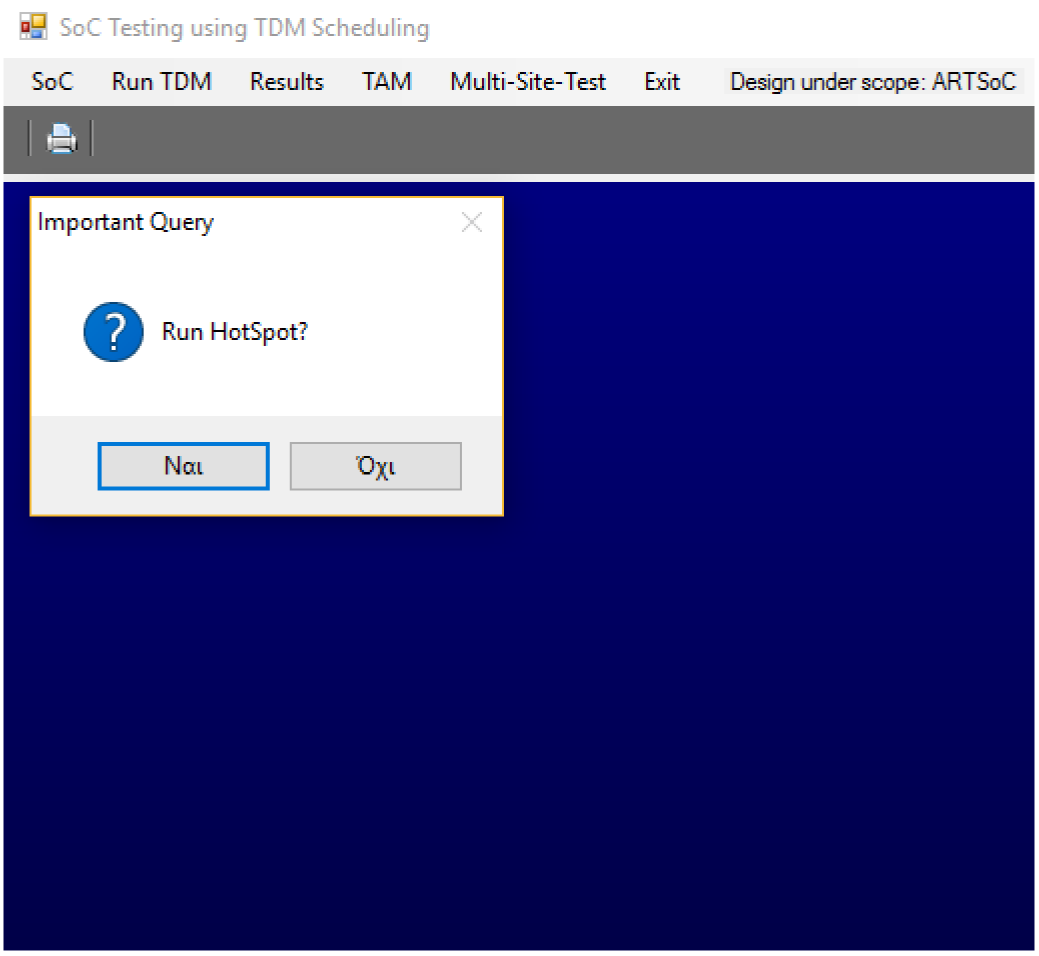


Figure B.5: Execution of the hotspot tool in the *SoCTDMScheduler*.

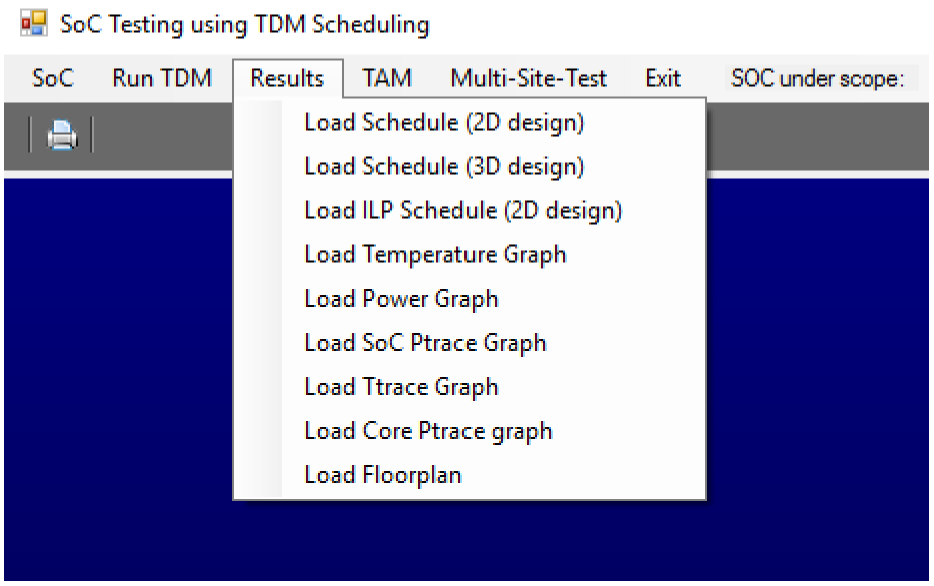
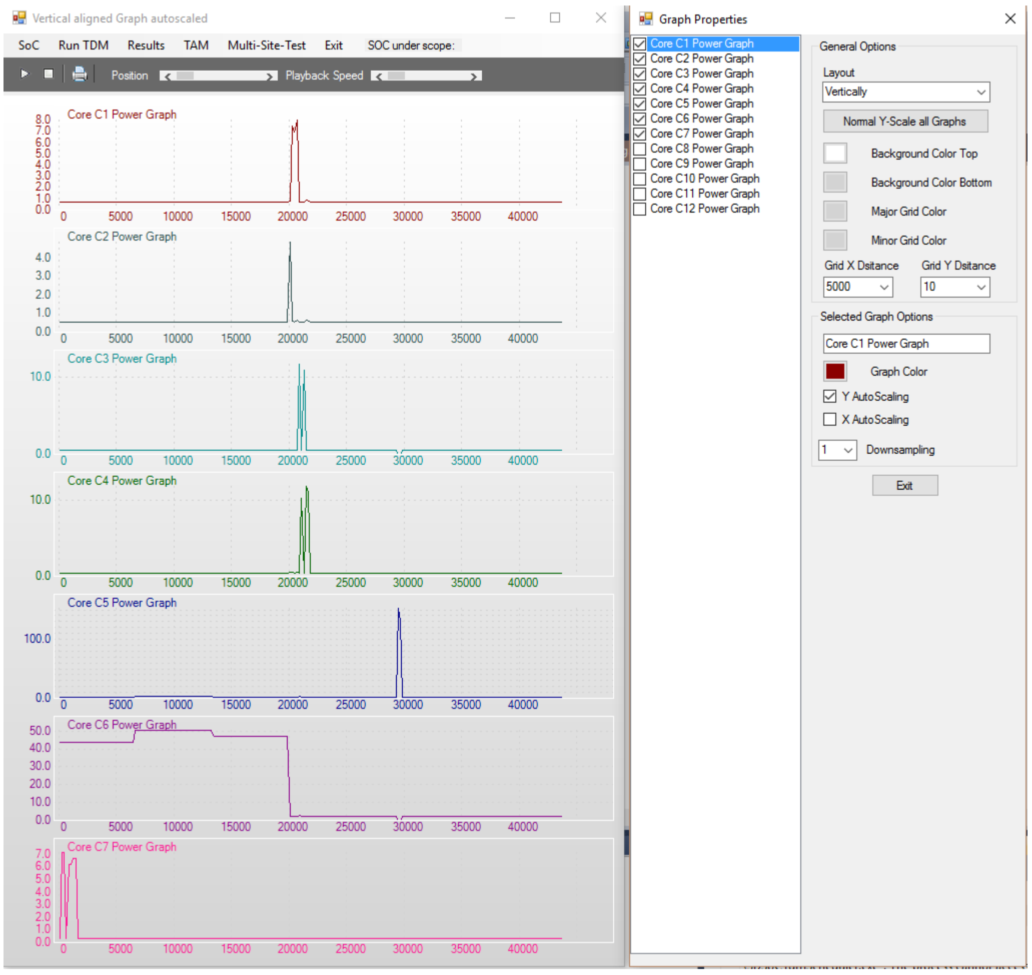


Figure B.6: Test scheduling process presentation in the *SoCTDMScheduler*.



- ✓ Core C1 Power Graph
- ✓ Core C2 Power Graph
- ✓ Core C3 Power Graph
- ✓ Core C4 Power Graph
- ✓ Core C5 Power Graph
- ✓ Core C6 Power Graph
- ✓ Core C7 Power Graph
- ☐ Core C8 Power Graph
- ☐ Core C9 Power Graph
- ☐ Core C10 Power Graph
- ☐ Core C11 Power Graph
- ☐ Core C12 Power Graph

Layout: Vertically

Grid X Distance: 5000 Grid Y Distance: 10

Core C1 Power Graph

✓ Y AutoScaling

☐ X AutoScaling

1 Downsampling

Figure B.7: The graph-properties window in the *SoCTDMScheduler*.

trated, the user can:

- Enter in the TAM optimization environment,
- Calculate the optimal distribution of a set of test channels in the partitions of a large SoC using a predefined set of TAM configurations per partition,
- Calculate the test load in each partition of a large SoC,
- Calculate the distribution of a set of test channels in the partitions of a large SoC in proportion to the test load of each partition.

In the TAM optimization environment, the user is able to select the SoC of interest and customize the TAM optimization process according to the needs. Thus, as shown in Fig. B.9, the user can configure the following parameters:

- The parameter *flexibility* concerns an heuristic method for TDM-based TAM optimization, not presented in this thesis. It is still in an experimental stage.

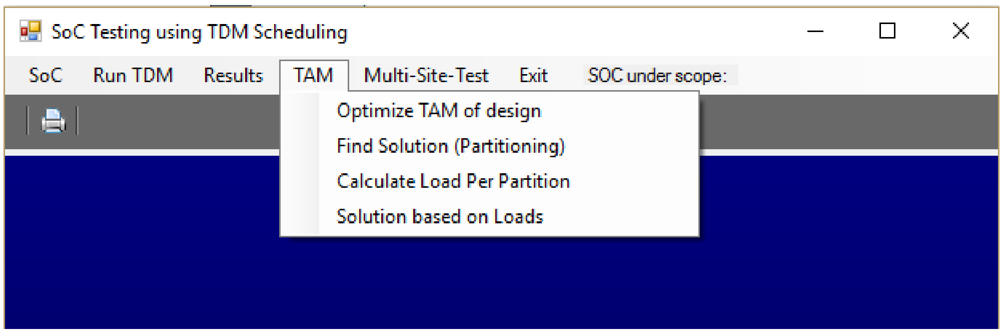


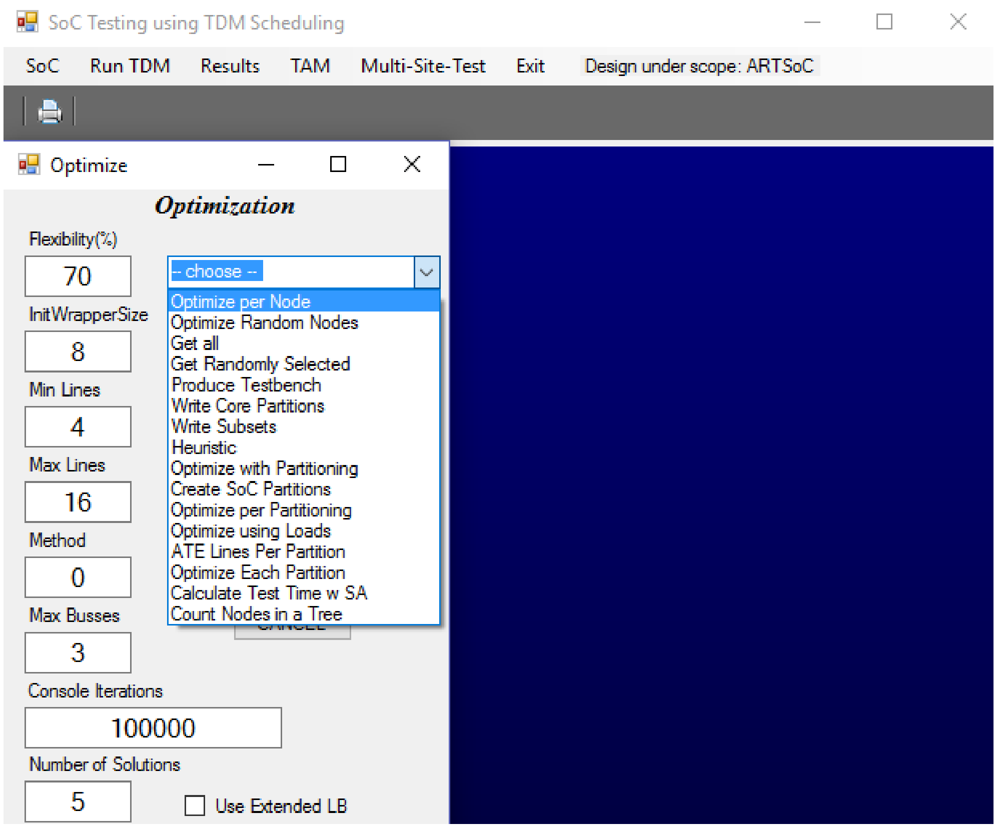
Figure B.8: The TAM optimization environment in the *SoCTDMScheduler*.

- The parameter *InitWrapperSize* refers to the default WPP width of each core's wrapper in the SoC, according to which the test time of this core has been calculated.
- The parameter *Min Lines* refers to the minimum number of the test channels that can be assigned in a TAM bus.
- The parameter *Max Lines* refers to the maximum number of the test channels that can be assigned in a TAM bus.
- The parameter *method* refers to the TDM algorithm that will be used to evaluate each TAM configuration. The values '0' and '1' corresponds to the GRD and the SA-RP methods.
- The parameter *Max Buses* refers to the maximum number of the test buses that can be used in a TAM.
- The parameter *Console iterations* is related to console presentation issues.
- The parameter *Number of solutions* refers to the number of the derived solutions that should be stored, when the TAM optimization process finishes. For example, if we use the value five, then the five best TAM configurations will be stored. An example of a solution is given in Fig. B.10. The extension of a TAM configuration file is *.c2d*.
- The parameter *Use extended LB* refers to the calculation of the *LB* used in the B-&-B algorithm. If it is not selected, then only the ideal *LB* is calculated (see section 3.5.3).

When the TAM optimization parameters have been set, the user can proceed in the execution of a variety of processes that are related to the TAM optimization. Specifically, the user is able to:

- Execute the TAM optimization process in small and medium SoCs for single-site and multi-site test,

- Execute the TAM optimization process in the partitions of large and very large SoCs for single-site and multi-site test,
- Derive random TAM configurations for evaluation purposes,
- Derive every possible TAM configuration, without rejecting any of them due to *LB* criterion,
- Execute the TAM optimization process using the heuristic method,
- Execute the global TAM distribution process, to assign the optimal number of test channels in each partition of a large / very large SoC,
- Produce an SoC and a TAM configuration file from the derived solutions, for evaluation purposes,
- Create and / or redefine partitions in a large and very large SoC using either a manual or an automated way,
- Create statistical information that is related to the TAM optimization process.



Flexibility(%) 70

InitWrapperSize 8

Min Lines 4

Max Lines 16

Method 0

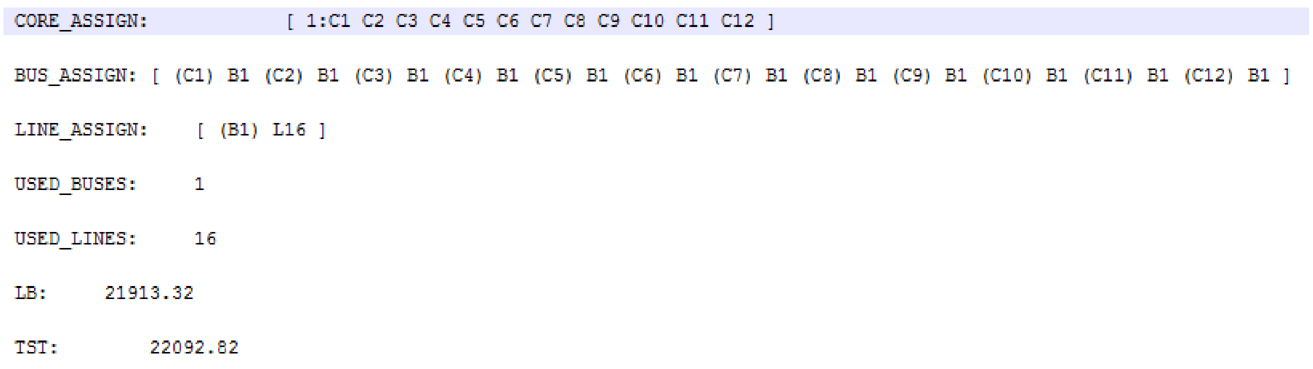
Max Busses 3

Console Iterations 100000

Number of Solutions 5

☐ Use Extended LB

Figure B.9: Configuration and execution of processes in the TAM optimization environment.



```
CORE_ASSIGN:            [ 1:C1 C2 C3 C4 C5 C6 C7 C8 C9 C10 C11 C12 ]

BUS_ASSIGN: [ (C1) B1 (C2) B1 (C3) B1 (C4) B1 (C5) B1 (C6) B1 (C7) B1 (C8) B1 (C9) B1 (C10) B1 (C11) B1 (C12) B1 ]

LINE_ASSIGN:    [ (B1) L16 ]

USED_BUSES:     1

USED_LINES:     16

LB:     21913.32

TST:        22092.82
```

Figure B.10: TAM configuration file.

B.2.3 Functionality for STDM-based multi-site testing

The $SoCTDMScheduler$ provides with the needed tool-set for STDM-based multi-site testing. These tools are accessible via the menu item $Multi-site \quad Test$, as it is illustrated in Fig. B.11. The following actions are supported:

- The user, based on a predefined SoC and TAM configuration, can alternate the WPP width of the core wrappers in the SoC and create a new valid SoC and TAM configuration file.
- The user can produce multi-site-oriented test schedules based on the TDM methods.
- The user can produce multi-site-oriented test schedules based on the STDM method.

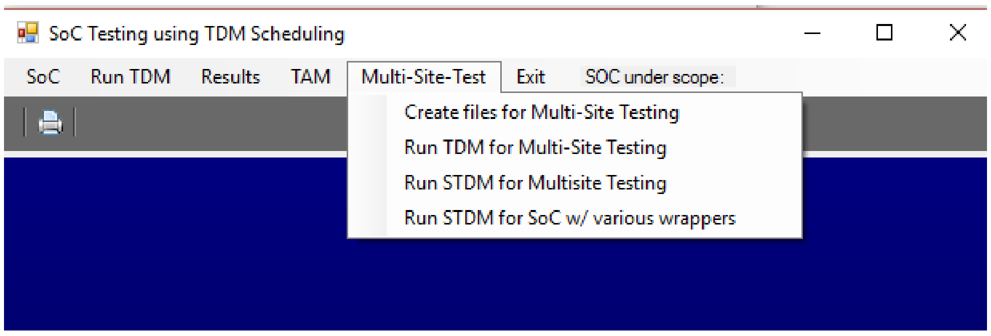

SoC Testing using TDM Scheduling
SoC
Run TDM
Results
TAM
Multi-Site-Test
Exit
SOC under scope:
Create files for Multi-Site Testing
Run TDM for Multi-Site Testing
Run STDM for Multisite Testing
Run STDM for SoC w/ various wrappers

Figure B.11: STDM-based multi-site testing in the $SoCTDMScheduler$.

Glossary

ASIC Application Specific Integrated Circuit. 12, 126, 128

ATE Automatic Testing Equipment. 21–23, 31, 37, 59, 60, 62, 64–68, 77, 78, 83–85, 89, 90, 92, 96, 98, 102, 108, 131, 132, 136, 137, 147, 148, 150–152

ATPG Automatic Test Pattern Generation. 20–22, 34, 117, 128

B-&-B Branch-&-Bound. 96, 108, 110, 112, 114–116, 148–151, 155, 156, 163

BAF Best-Area-Fit. 83

BIST Built-In Self-Test. 27, 31, 37, 51, 68

BL Bottom-Left. 77

BSC Boundary-Scan Cell. 35

CTL Core Test Language. 36

CVS Clustered Voltage Scaling. 10, 11

DEF Design Exchange Format. 128, 129

DFE Design-for-the-Environment. 5

DFM Design for Manufacturability. 14

DFR Design for Reliability. 14

DFT Design-for-Testability. 21, 25, 27, 29, 30, 32, 34, 59, 86, 128

DFY Design for Yield Enhancement. 14

DSP Digital Signal Processor. 3

DUT device under test. 5, 19–23, 25, 30–32, 45, 46, 58, 117

DVFS Dynamic Voltage and Frequency Scaling. 8, 9, 11, 58, 60–62, 85, 162

DVS Dynamic Voltage Scaling. 8

ECVS Extended Clustered Voltage Scaling. 10, 11

FA Fill-Adjacent. 117, 118, 158–161

FMA Failure Mode Analysis. 21

FSM Finite State Machine. 57

GRD Greedy. 83, 132, 136, 139, 140, 142, 143, 146

GRP Greedy Rectangle Packing. 110, 111, 115, 148–151, 155

GS Gate Sizing. 48

HDL Hardware Description Language. 126, 128

IC Integrated Circuit. 2, 14, 24, 27, 36, 37, 125, 126

ILP Integer Linear Programming. 62, 65, 76, 77, 82, 99, 136, 137, 139, 140, 142, 143, 146

IoT Internet of Things. 1–5

IP Intellectual Property. 2, 3, 8, 10, 36, 86, 146

LEF Library Exchange Format. 129

LOC Launch-On-Capture. 117

LSC Logic State Configuration. 42

LSSD Level-Sensitive Scan Design. 30

MFA Modified-Fill-Adjacent. 158–161

MILP Mixed Integer Linear Program. 55

MUX Multiplexer. 29

ORA Output Response Analyzer. 31

PCB Printed Circuit Board. 36

PF Preferred-Fill. 117

PMScan Power Managed Scan. 48

RBF Resistive Bridge Faults. 42

RF Random-Fill method. 158–161

RP Rectangle Packing. 77–80, 82–84, 95, 96, 100, 110, 111, 132, 136, 139, 140, 142, 143, 146, 149, 151

RTL Register Transfer Level. 126, 128, 131

SA Simulated Annealing. 79, 80, 82, 84, 95, 96, 100, 110, 111, 132, 136, 139, 140, 142, 143, 146, 149, 151

SBS Session-Based Scheduling. 136, 137, 139, 140, 142

SDM Space-Division-Multiplexing. 84, 86, 89, 90, 92, 94, 162

SJF Shortest-Job-First. 135, 136, 139, 140

SoC System on Chip. 2–5, 9–11, 22, 24, 25, 32, 34–38, 49, 50, 53–55, 58–63, 65, 66, 68–70, 76–79, 81–86, 89, 90, 92, 94, 96–104, 107, 111–116, 125, 126, 131–140, 142–144, 146, 148–154, 156, 162, 163

SPEF Standard Parasitic Exchange Format. 129

STDM Space and Time Division Multiplexing. 60, 84, 89, 90, 92, 96, 132, 144, 146–148, 162, 163

TAM Test Access Mechanism. 25, 32, 34–37, 48–51, 53, 59–64, 67–70, 74, 75, 77, 78, 84–86, 89, 90, 92, 96–103, 107, 108, 110–116, 125, 131, 132, 137, 138, 146, 148–156, 163

TAP Test Access Port. 35

TDM Time Division Multiplexing. 59–70, 78, 79, 84–86, 89, 90, 92, 95–101, 108, 110, 112, 116, 131, 132, 134, 136–140, 142–144, 146–148, 151, 154, 156, 162, 163

TPG Test Pattern Generator. 31

TPI Test Point Insertion. 47, 48

TTTC IEEE Test Technology Technical Council. 36

VLSI Very-Large-Scale Integration. 5, 13, 14, 18, 27, 29, 126

VSIA Virtual Socket Interface Alliance. 36

WBC Wrapper Boundary Cells. 35

WBR Wrapper Boundary Register. 35

WBY Wrapper Bypass Register. 35

WIR Wrapper Instruction Register. 35

WPC Wrapper Parallel Control. 35

WPI Wrapper Parallel Input. 35

WPO Wrapper Parallel Output. 35

WPP Wrapper Parallel Port. 51, 86, 89, 90, 92, 94, 96, 131, 144, 146, 147

WSC Wrapper Serial Control. 35

WSI Wrapper Serial Input. 34

WSO Wrapper Serial Output. 34

WSP Wrapper Serial Port. 34, 35

Author's Publications

1. F. Vartziotis, X. Kavousianos, K. Chakrabarty, R. Parekhji, A. Jain, Multi-Site Test Optimization for Multi-V_{dd} SoCs Using Space - and Time - Division Multiplexing, *Proc. of Design, Automation, and Test in Europe conference* (2014).

2. F. Vartziotis, X. Kavousianos, K. Chakrabarty, A. Jain, R. Parekhji, Time-Division Multiplexing for Testing DVFS-based SoCs, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* vol 34 (2015) 668–681.

3. F. Vartziotis, X. Kavousianos, K. Chakrabarty, A Branch-&-Bound Algorithm for TAM Optimization in Multi-Vdd SoCs, *Proc. of IEEE European Test Symposium* (2015).

4. F. Vartziotis, X. Kavousianos, P. Georgiou, K. Chakrabarty, Test-Access-Mechanism Optimization for Multi-Vdd SoCs, *Proc. of IEEE International Test Conference* (2015).

5. F. Vartziotis, X. Kavousianos, P. Georgiou, K. Chakrabarty, A Branch-&-Bound Test-Access-Mechanism Optimization Method for Multi-V_{dd} SoCs, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (under minor revision)

6. F. Vartziotis, X. Kavousianos, Critical Path - Oriented & Thermal Aware X-Filling for High Un-modeled Defect Coverage, (Accepted to Design, Automation, and Test in Europe conference)(2017)

Short Vita

[ ] Fotis Vartziotis received his Diploma in Electrical Engineering in 1998 (Aristotle University, GR), M.Sc. degree in Telematics in 2000 (University of Surrey, UK), MSc degree in Management of Technical Projects in 2010 (Hellenic Open University, GR) and PhD degree at University of Ioannina. Since 2010, he is professor of practice (Dept. of Computer Engineering, Technological Educational Institute of Epirus, GR).