ABSTRACT

Title of dissertation: PHYSICAL DESIGN METHODOLOGIES FOR LOW POWER AND RELIABLE 3D ICs

Tiantao Lu, Doctor of Philosophy, 2016

Dissertation directed by: Professor Ankur Srivastava
Department of Electrical Engineering

As the semiconductor industry struggles to maintain its momentum down the path following the Moore's Law, three dimensional integrated circuit (3D IC) technology has emerged as a promising solution to achieve higher integration density, better performance, and lower power consumption. However, despite its significant improvement in electrical performance, 3D IC presents several serious physical design challenges. In this dissertation, we investigate physical design methodologies for 3D ICs with primary focus on two areas: low power 3D clock tree design, and reliability degradation modeling and management.

Clock trees are essential parts for digital system which dissipate a large amount of power due to high capacitive loads. The majority of existing 3D clock tree designs focus on minimizing the total wire length, which produces sub-optimal results for power optimization. In this dissertation, we formulate a 3D clock tree design flow which directly optimizes for clock power. Besides, we also investigate the design methodology for clock gating a 3D clock tree, which uses shutdown gates to selec-

tively turn off unnecessary clock activities. Different from the common assumption in 2D ICs that shutdown gates are cheap thus can be applied at every clock node, shutdown gates in 3D ICs introduce additional control TSVs, which compete with clock TSVs for placement resources. We explore the design methodologies to produce the optimal allocation and placement for clock and control TSVs so that the clock power is minimized. We show that the proposed synthesis flow saves significant clock power while accounting for available TSV placement area.

Vertical integration also brings new reliability challenges including TSV's electromigration (EM) and several other reliability loss mechanisms caused by TSV-induced stress. These reliability loss models involve complex inter-dependencies between electrical and thermal conditions, which have not been investigated in the past. In this dissertation we set up an electrical/thermal/reliability co-simulation framework to capture the transient of reliability loss in 3D ICs. We further derive and validate an analytical reliability objective function that can be integrated into the 3D placement design flow. The reliability aware placement scheme enables co-design and co-optimization of both the electrical and reliability property, thus improves both the circuit's performance and its lifetime. Our electrical/reliability co-design scheme avoids unnecessary design cycles or application of ad-hoc fixes that lead to sub-optimal performance.

Vertical integration also enables stacking DRAM on top of CPU, providing high bandwidth and short latency. However, non-uniform voltage fluctuation and local thermal hotspot in CPU layers are coupled into DRAM layers, causing a non-uniform bit-cell leakage (thereby bit flip) distribution. We propose a performance-

power-resilience simulation framework to capture DRAM soft error in 3D multi-core CPU systems. In addition, a dynamic resilience management (DRM) scheme is investigated, which adaptively tunes CPU's operating points to adjust DRAM's voltage noise and thermal condition during runtime. The DRM uses dynamic frequency scaling to achieve a resilience borrow-in strategy, which effectively enhances DRAM's resilience without sacrificing performance.

The proposed physical design methodologies should act as important building blocks for 3D ICs and push 3D ICs toward mainstream acceptance in the near future.

PHYSICAL DESIGN METHODOLOGIES FOR LOW POWER AND RELIABLE 3D ICs

by

Tiantao Lu

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2016

Advisory Committee:
Professor Ankur Srivastava, Chair/Advisor
Professor Donald Yeung
Professor Gang Qu
Professor Manoj Franklin
Professor Peter Sandborn

Dedicated to my parents, for their love and support.

Acknowledgments

I would like to express my appreciation to my advisor, Professor Ankur Srivastava for his guidance, advice, and support throughout my Ph.D. study at University of Maryland. Professor Srivastava has introduced me to the exciting field of Electronic Design Automation, and has provided numerous helpful guidance and encouragement during my five years in Maryland.

I would also like to thank Professor Donald Yeung, Professor Shuvra Bhattacharyya, Professor Gang Qu, Professor Manoj Franklin and Professor Peter Sandborn for their time to serve on this committee and their valuable technical feedback on the content of this dissertation.

I would like to extend my thanks to all my colleagues. My appreciation first goes to two of my senior colleagues, Dr. Bing Shi and Professor Domenic Forte for helping me adapt to academia in early stage of my Ph.D. study. Second I thank my current colleagues, Caleb Serafy, Chongxi Bao, Zhiyuan Yang, Yang Xie and Yuntao Liu, for the great technical projects we have collaborated on, and the fun times we have together throughout the years.

Finally, I would like to express my deepest gratitude to my beloved parents, Professor Fang Lu and Professor Jianwei Wan for their endless love and devotion. I will try to return their love, and I wish my parents to live healthy and happily for long.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

3D IC	Three Dimensional Integrated Circuit
BCB	Benzocyclobuten
CMOS	Complementary Metal-Oxide Semiconductor
CTE	Coefficient of Thermal Expansion
CMP	Chemical Mechanical Polishing
DME	Deferred-Merge-Embedding
DRAM	Dynamic Random-Access Memory
DRIE	Deep Reactive Ion Etching
EM	Electromigration
FEM	Finite Element Method
ILD	Interlayer Dielectric
KOZ	Keep Out Zone
MEMS	Micro-electromechanical Systems
MTTF	Mean Time To Failure
PDN	Power Delivery Network
PVD	Physical Vapor Deposition
RF	Radio Frequency
RTL	Register-Transfer-Level
SA	Simulated Annealing
SEM	Scanning Electron Microscope
TSV	Through Silicon Via
WL	Wirelength

Chapter 1: Introduction

In the past few decades, the demand for high-performance and complex functionality in integrated circuits is primarily met through aggressive device scaling. Device scaling results in smaller and faster transistors and higher integration density on the same chip area. Device scaling has been fulfilling the prophecy of Moore's Law, and has been proved to be a very effective solution in many aspects such as performance and cost. At the time when this dissertation is written, 14 nm CMOS transistors have been successfully commercialized and 10nm transistors are under development. However, the trend of transistor scaling seems to be near saturation. Further transistor scaling will inevitably encounter physical limitations and might not be cost-effective. Another huge issue of device scaling is that interconnects are becoming slower. Interconnect's RC delay has increased to such an extent that wire delay has become the primary bottleneck for further performance boost. More complicated functionalities require longer global routing thereby higher power dissipation. The wire delay and power consumption will become larger as devices are made even smaller.

Three-dimensional integrated circuit (3D IC) is one of the most promising technologies that aim to continue Moore's Law. An illustration of a 3D IC is shown in Figure 1.1. 3D ICs stack multiple wafers/chips vertically, and through-silicon-vias (TSVs) establish vertical communications between adjacent tiers. Heterogeneous technologies such as CPUs, memories, radio frequency (RF) circuits, analog circuits and sensors can be freely integrated in the same package to avoid long and slow off-chip wires. Typically a 3D IC is connected to PCB through C4 bumps for power delivery. A heat spreader and a heat sink are on the other side of the chip.

The stacking structure of 3D ICs and the employment of TSVs replace long off-chip interconnects in 2D ICs, resulting in significant improvement in bandwidth, power and performance. However, there are many new design challenges for 3D ICs. This dissertation focuses on 3D clock tree design, and resolving 3D IC specific reliability issues. These challenges and design optimization opportunities are discussed in the following subsections.

1.1 3D Clock Network Design

Three-dimensional clock design has become an important and challenging research topic, striving to provide synchronization for all computations across the 3D chip. Once the locations of the clock source (*e.g.*, phase locked loop, delay locked loop *etc.*) and clock sinks (flip-flops and latches) are known, the clock signal is delivered to each clock domain of the chip through the clock network delivers.

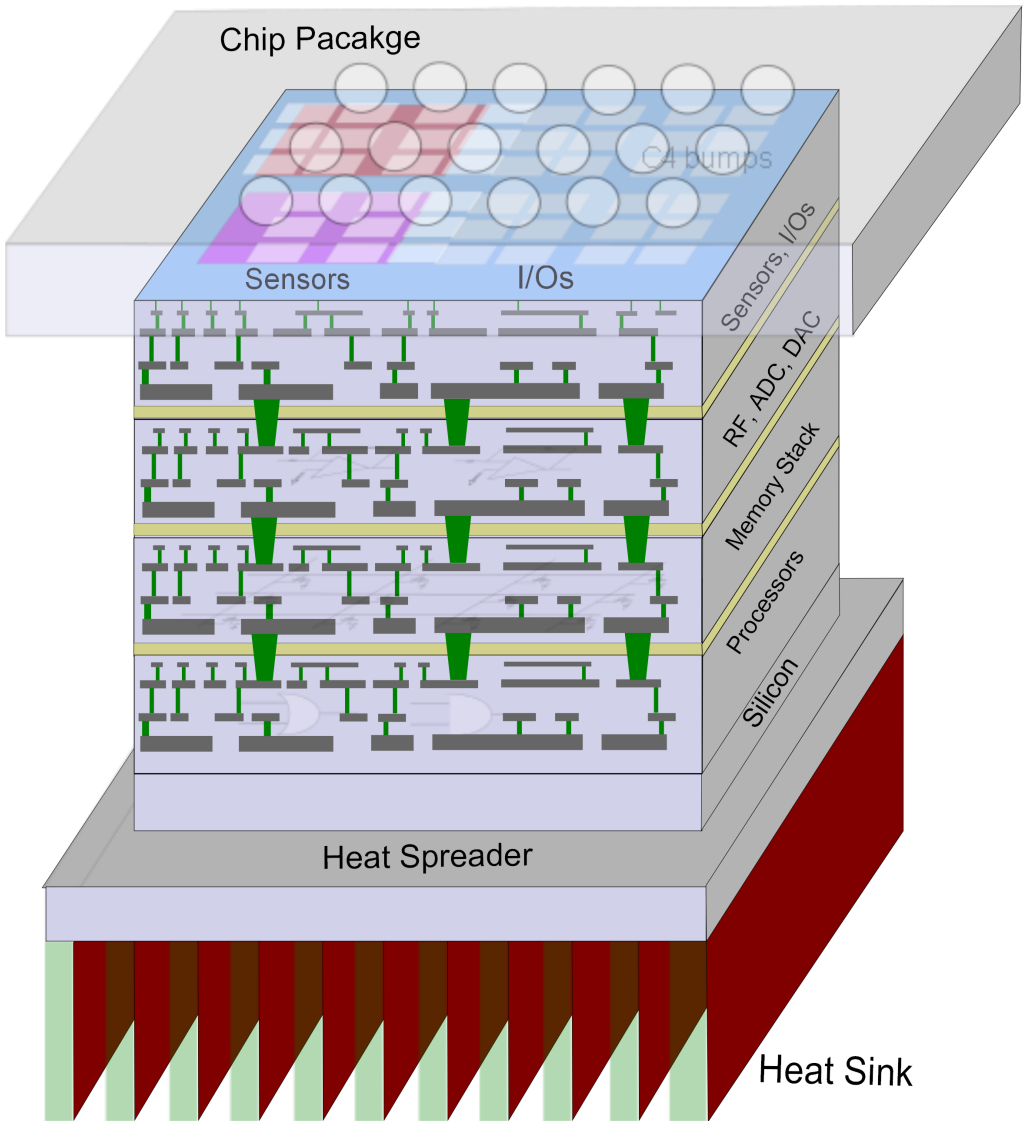


Figure 1.1: 3D IC provides a solution for heterogeneous integration. TSV establishes fast and massive vertical data transfer path, which is critical for modern computing and storage paradigms.

Like 2D clock networks, 3D clock networks can be implemented as either a mesh or a tree. A typical clock mesh structure is established by intersecting vertical and horizontal metal wires. Clock sinks connect to the clock mesh through short wires, often called "stubs". The clock signal is distributed from a clock source with a high-level clock tree structure to the intersection of horizontal and vertical metal

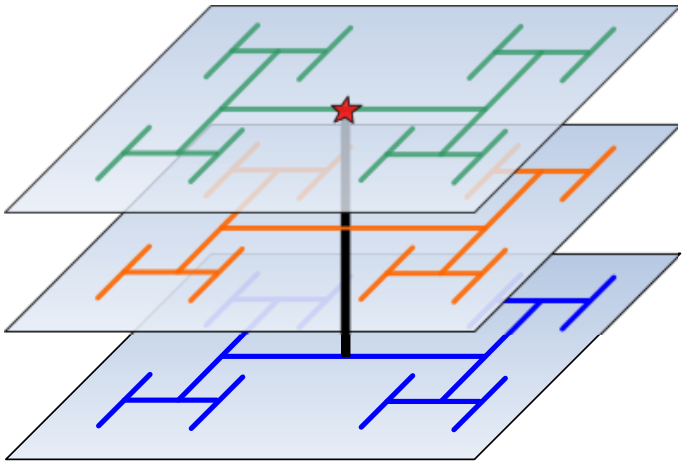


Figure 1.2: A naive H-tree implementation of 3D clock network. Clock TSVs are used to distribute clock signals across different layers, however, 3D H-tree results in high clock skew.

wires. Clock mesh provides low clock skew synchronization in the presence of process and environmental variations, however it imposes wiring utilization overheads and a high power consumption.

The clock tree topology is the prevailing topology used in practice. The power consumption of a typical clock tree is much lower than clock mesh, and the 2D synthesis flow has been heavily investigated in the past. One typical optimization goal for clock tree synthesis is to ensure zero, bounded, or useful clock skew (the maximum difference in clock arrival time between sinks). Other objectives include minimizing clock power, minimizing total wire length, *etc.* A naive 3D H-tree is shown in Figure 1.2. The clock signal is distributed across layers via clock TSVs. However, the H-tree structure (in blue) at the bottom layer has longer wiring, indicating late clock arrival time and non-zero clock skew. More sophisticated clock tree synthesis flow is obviously needed for 3D ICs.

The 3D clock tree synthesis flow needs to consider the usage and location of clock TSVs to optimize the clock tree's performance. Too few clock TSVs might not effectively exploit the potential of 3D integration, however too many clock TSVs might increase the clock tree's capacitive load (as TSVs have large capacitance) as well as increase the manufacturing cost and failure probability. In addition, since clock network design is generally performed after placement, 3D clock tree algorithms need to consider the feasibility of placing clock TSVs into the layout whitespace. TSVs placement is especially important as TSVs have larger dimensions than standard cells, and consume a large amount of placement resources. For example, a TSV could occupy $5\mu m \times 5\mu m$ *to* $30\mu m \times 30\mu m$ in area, depending on different TSV fabrication technologies, as compare to less than $1\mu m \times 1\mu m$ standard cells such as inverters and adders at sub-micro technology node. In cases where additional controlling signals are needed (*i.e.* the enable signal for clock gating), the placement of control TSV needs to be considered as well.

1.2 TSV's Electromigration

Electromigration (EM) degradation has become one of the most crucial failure causes in modern VLSI circuit [4]. EM refers to the migration of atoms in solid-state conductors. The migration of atoms will form hillocks and voids over time, and significantly changes the interconnect's resistance thus influences the circuit perfor-

mance (increases digital logic's delay, causes IR drops in power delivery systems, and etc.) and eventually leads to severe interconnect failures such as short-circuit and open-circuit.

Traditionally, the interconnect's wear-out due to EM in planar circuit has been expressed by the empirical Black equation [5]. The Black equation, as shown in Equation 1.1 is based on the observation that the life time of a single metal wire is inversely proportional to current density (j).

$$MTTF = Aj^{-n}exp\,(E_a/kT) \tag{1.1}$$

Equation 1.1 calculates interconnect's MTTF (Mean-time-to-failure) based on current density (j) and temperatures (T). Here, k is the Boltzmann's constant, E_a is the EM activation energy. The symbol A is a structural dependent constant. The value of n, according to Black, equals to 2.

However, many recent 3D IC experimental works have shown that the current density is not the only driving force for TSV's EM. TSV's wear-out is subject to several interacting forces including current density, thermal mechanical stress gradient, temperature gradient, and atomic concentration gradient.

TSVs in 3D ICs are faced with severe EM degradation. There are three primary reasons for the severe EM phenomenon: (1) TSV's EM is strong function of the thermal mechanical stress, and high thermal mechanical stress is observed around the TSV and its neighboring interconnects. The high mechanical stress is brought by coefficient of thermal expansion (CTE) mismatch between the TSV (i.e. copper

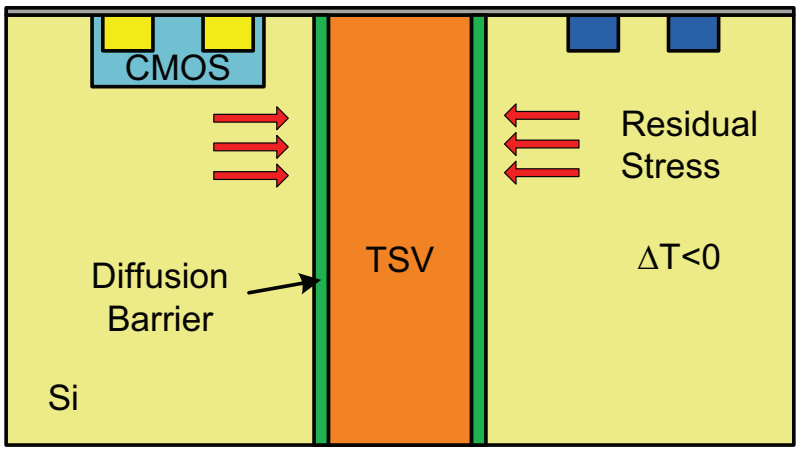


Figure 1.3: TSV's manufacturing process induces significant thermal mechanical stress around TSV. The stress field induces TSV stress migration and material fracture, and causes mobility deviation in nearby CMOS devices.

$1.77 \times 10^{-5} K^{-1}$) and the silicon substrate ($3.05 \times 10^{-6} K^{-1}$). When TSVs are cooled from a high stress-free annealing temperature to a low room temperature, tensile and compressive stress is formed inside the TSV and the substrate. This phenomenon is illustrated in Figure 1.3. The stress-induced EM severely deteriorates 3D IC's reliability.

(2) TSV's EM is also a strong function of temperature. Local hotspots are expected to appear in 3D ICs due to the poor heat conduction in the stacked structure (conductivity of inter-layer dielectric is around $1W \cdot m^{-1} \cdot K^{-1}$, and the conductivity of Si is $149W \cdot m^{-1} \cdot K^{-1}$). Local hotspot makes atoms in TSVs diffuse faster. Thirdly, in planar chip design, redundant wires and vias are widely adopted to enable on-line reconfiguration of interconnects [6]. However in 3D ICs, TSV's high manufacturing cost and low yield make redundant design less feasible than planar circuit.

1.3 TSV Stress-induced Material Fracture

Recently TSV-induced thermal mechanical stress has become one important reliability concern. Due to the CTE mismatch between TSV and silicon substrate, whenever 3D ICs experience temperature changes, TSV material deforms more than the silicon substrate, leaving significant residual thermal mechanical stress in TSV's neighboring area. Once exceeds the material's yield strength, residual stress results in TSV's interfacial delamination [7] and silicon substrate cracking [8] (we refer both as material fracture).

The von Mises yield criterion is one of the most widely used criterion to quantify material fracture. The von Mises yield criterion suggests that the yielding of materials begins when the von Mises stress exceeds material dependent yield strength [9–11]. The von Mises stress is a scalar stress value that can be computed from the stress tensor in the Cartesian coordinate system, as follows.

$$\sigma_v = \frac{1}{\sqrt{2}}\Big[(\sigma_{xx}-\sigma_{yy})^2+(\sigma_{yy}-\sigma_{zz})^2+(\sigma_{zz}-\sigma_{xx})^2 +6(\sigma_{xy}^2+\sigma_{yz}^2+\sigma_{zx}^2)\Big]^{\frac{1}{2}}. \tag{1.2}$$

1.4 Thesis Outline

In this thesis we first provide background and motivation for developing physical design methodologies for 3D CPUs in Chapter 2. This includes the ending of Moore's Law, the advantages of 3D integration, and its physical design challenges.

Chapter 3 discusses the low power design methodologies for 3D clock tree. The clock tree is one of the largest and most frequently switched networks in 3D IC, making it to be a major contributor to the chip's total power. Applying shutdown gates that selectively turn off certain clock tree branches to avoid unnecessary clock activities is a common practice in planar circuit. However, in 3D clock tree, shutdown gates are connected to the control unit via *control* TSVs, therefore excessive use of shutdown gates may cause TSV's placement conflicts to the existing floorplan. We have developed a 3D clock tree synthesis flow which simultaneously decides the physical routing of the clock tree as well as the usage of shutdown gates. Using the proposed synthesis flow, we are able to construct a 3D clock tree with significant power reduction while ensuring zero clock skew.

The modeling and EM-aware layout optimization for tapered TSV is discussed in Chapter 4. We have realistic TSV's shape deviates from an ideal cylinder, and recognize the source of the tapering effect from the TSV manufacturing process. To understand the EM implications on tapered TSVs, we set up finite-element-method (FEM) based simulation framework to quantify tapered TSV's current density distribution. We show that the DC current crowding effect is more pronounced at the narrow end of the tapered TSV than the ideal cylindrical TSV, making tapered

TSV more prone to EM failure. To further speed up current density simulation, we develop an adaptive resistive mesh for capturing the DC current density distribution inside the tapered TSV, which is faster than the FEM simulator by several orders of magnitude and achieves comparable accuracy. We formulate an optimization problem that minimizes the TSV-involved delay while constraining the peak current density values inside the tapered TSV. With the resistive model we size the width of the wires connecting to the TSV to spread out the current. TSV-involved timing analysis is performed using a second-order delay model, which has higher accuracy than the Elmore delay model. Finally we develop a dynamic programming based approach to solve this current constrained delay minimization problem.

In Chapter 5, a more advanced TSV's EM model is investigated. Different from the assumption used in Chapter 4 where EM is solely decided by current density, many recent experimental works have suggested thermal mechanical stress and temperature also impact TSV's EM. We develop a transient simulation framework using FEM that monitors how atoms migrate over time under the influence of electrical current, temperature gradient, and thermal mechanical stress. More importantly, we propose an analytical TSV's EM objective function that correlates well with the numerical simulation results. Besides, we propose another analytical objective function to predict material fracture based on the von Mises yield criterion. These two reliability objective functions are integrated into 3D IC's placement flow, which optimize 3D IC's reliability simultaneously with its performance. Re-

sults show that by locally changing the placement of TSVs and logic gates, we are able to achieve longer TSV lifetime and less thermal stress, with little degradation on performance.

In Chapter 6, we investigate the WL voltage noise induced stacked DRAM transient fault problem. We propose a performance - power - voltage - resilience simulation framework, which effectively captures the interaction between multi-core CPU layer's activity, and voltage noise induced DRAM transient fault. We observe significant correlation between CPU core's activity and DRAM voltage and thermal profiles, both in time and space. As the DRAM leakage strongly depends on voltage noise and temperature, we further observe significant temporal and spacial variation in bit-cell's leakage rate. These two observations imply that the DRAM failure probability strongly depends on 3D CPU's operating points. In our simulation experiments, we investigate an off-line task assignment problem, and propose a two-stage DRAM resilience management approach, which enables frequency allocation for CPU cores and refresh rate adjustment for DRAMs, in order to optimize 3D-CPU's performance while accommodating DRAM resilience requirement.

We provide the conclusion of this dissertation the future work in Chapter 7.

Chapter 2: Background and Motivation

CMOS technology has approached a critical junction where traditional device and interconnect scaling are unable to keep up with Moore's Law. The underlying reason is that engineers are facing several dilemmas in advanced technology nodes. The first dilemma comes as chip feature size approaches the lower limits of current photolithography technology. Investment in next-generation lithography solutions is possible but costly. The second dilemma is the ever-increasing leakage current. For example, thin gate oxide results in substantial gate tunneling leakage and also sub-threshold leakage. Employment of metal gates and high-k dielectrics is an effective approach to control the leakage current, however, its compatibility with CMOS process has raised some concerns. The third dilemma is increasing dominance of wire delay and power in future technology nodes. Limitations to DRAM bus bandwidth and speed are inherent to off-chip integration, and lead to a severe "memory wall" problem in the era of big data. Furthermore the power dissipated by such a coarse integration paradigm leads to huge dynamic power dissipation and inhibits scaling towards envisioned exascale computing systems of the future.

Vertical integration provides a revolutionary solution to reduce interconnect power and delay while increasing transistor density independent of costly device scaling. Chips fabricated by existing technology can be directly bonded together to increase the number of transistors per unit area. This configuration avoids investment of new generation of devices, and enables heterogeneous technology integration on chip. Vertical integration between layers is established by through-silicon-vias (TSVs). TSVs are essentially metal pillars that penetrate the silicon substrate to engage the metal pads of the layer below. This extra connectivity in the third dimension substantially reduces the wire delay and power, and also provides tremendous bandwidth for data transfer between layers. 3D integration across N layers can theoretically reduce chip wire length up to a factor of $\sqrt{N}$ [12].

2.1 Fundamental limitations of transistor scaling

Over the last several decades, the growth of computing power and IC's complexity is based on the fact that device scaling makes transistors switch faster and denser. However, in advanced technology nodes, when transistors and interconnects are made smaller and denser, several major challenges rises.

The first major challenge is that traditional transistor scaling is reaching its limit. There are two bottlenecks that hinder further transistor scaling. The first is transistor leakage, including gate leakage (due to thinner gate oxide), subthreshold leakage (lower threshold voltage) etc. These leakage currents make the transistor harder to be turned off, impacting its switching speed, and consuming more power.

Novel device solutions such as high-k and metal gate are being developed, however, it is uncertain that further transistor scaling yields any cost-effective solution for leakage control.

The second bottleneck is the significance of process variation in advanced technologies. One example is the variation in dopant concentration. Smaller devices possess fewer impurities, therefore as device dimension shrinks, dopant concentration variation will have a more pronounced influence on transistor's threshold voltage. Another example is line edge roughness, referring to the observation that a pattern's edge deviates from a smooth, ideal shape. This effect emerges as the feather size of a device is less than the wavelength of lithography.

2.2 The rising of interconnect delay and power

The second major trend is that the interconnect delay and power have become a limiting factor to the performance of the planar ICs even if the transistor scaling continues. As IC's functionality becomes more complex, global wire length increases, resulting in significant increase in interconnect delay. One famous example is that the memory latency produced by long and slow buses between processor cores and dynamic random access memories (DRAMs). Reading/writing to memory takes tens to hundreds of clock cycles, therefore memory access has become one of the major overheads in modern high-performance computing systems. Inserting more buffers is common practice to reduce the interconnect delay, but it inevitably leads to higher power consumption.

Another important interconnect related drawback is the relatively narrow bandwidth. The number of bus channels is limited by the I/O pin counts from the external package. In the era of multi-core processors and big data, huge amounts of data packets are processed and waiting to be transferred between processor cores and memories, therefore limited bandwidth becomes a huge performance bottleneck.

2.3 3D IC: higher density devices, and faster interconnect

A 3D IC stacks multiple planar wafers/dies vertically, which results in an increased transistor density and reduced wire length. The components in different layers can communicate through vertical interconnects called TSVs. A large amount of TSVs can be used, which dramatically increases chip's bandwidth and also decreases interconnect delay/power. In addition, 3D ICs provide new design options by allowing heterogeneous materials, technologies and systems to be integrated into a single chip.

2.4 Clock Tree Design

The clock tree is a prevailing clock distribution network in digital circuit. A typical clock tree spans across the entire chip, drives a large amount of capacitive loads, and is operated at high frequency. It is a major contributor to the chip's total power in high-performance VLSI circuit, and some applications can take 50% [13] or even 70% [14] of the chip's total power.

Another important design objective for clock trees is the clock skew, which is the maximum difference in clock signal's arrival time. According to ITRS [15], the clock skew needs to be controlled within 3% to 4% of the clock period.

2.4.1 Traditional 2D Clock Tree Design

The earliest implementation of clock tree is H-tree [16]. Clock skew is well controlled (although non-zero) in H-tree due to its symmetric structure. Later, as clock tree power continues to grow, more sophisticated clock tree generation algorithms decide the structure of the clock tree based on the clock sinks' geometry information. For example, the method of means and medians (MMM) [17] bi-partitions the clock sinks recursively and connects the centers of the two partitions. A bottom-up method is proposed in [18], where a pair of clock sinks which generates the least clock skew under linear delay model is connected. The deferred-merge-embedding (DME) algorithm is presented in [19, 20], which achieve zero clock skew and shorter global wire length than MMM [17] and the bottom-up method [18].

Clock gating is one the most widely used technique for 2D clock tree's power optimization [21–29]. Clock gating exploits the fact that instructions are not executed with even frequency, causing spatial and temporal variations in the "on" and "off" states of the sequential logic. In 2D clock tree, as the cost of clock gating is cheap, shutdown gates can be inserted at most intermediate clock tree node to shut

down all its descendants' (wires and sequential logics) clock signal and power supply, when its downstream sequential logics are inactive, thereby reducing the dynamic power dissipated by wires, buffers, and sequential logics.

2.4.2 3D Clock Tree Design

Similar as the 2D clock tree design, the primary design objectives for 3D clock tree are also clock power and clock skew. However, the physical design methodologies for 3D clock trees are not fully investigated. One design challenge for 3D clock tree is that clock sinks are distributed among several vertical layers, and the usage of clock TSVs needs to be considered when 3D clock tree is generated. Too few clock TSVs might not fully exploit the benefit brought by the 3D integration, and too many clock TSVs introduce heavy capacitive load (as TSVs are large capacitors), and extra manufacturing cost.

A few 3D clock tree synthesis algorithms include 3D-MMM [2], MMM-3D [30], and nearest-neighbor (NN3D) [31]. In 3D-MMM and MMM-3D, the 3D-IC was recursively partitioned in horizontal (X/Y) or vertical (Z) direction. The partition decision is based on the availability of clock TSVs. The NN-3D algorithm greedily picks the closest sequential pairs and recursively forms the clock tree following a bottom-up fashion. Other works have incorporated controlling the clock skew. For example, the 3D-DME algorithm extends the 2D-DME algorithm [19,20] to the 3D space to include the RC of clock TSVs.

The impact of clock gating has seldom been investigated for 3D clock trees. Unlike the assumption in 2D clock trees that clock gating is cheap thus can be applied at every clock node, the 3D clock gating introduces extra overhead: shutdown gates need extra control TSVs to be connected to a centralized control unit. Clock tree design methodologies that account for the selection of shutdown gates in 3D clock trees have been lacking.

2.5 Reliability Issues in 3D ICs

TSVs are crucial interconnects in 3D ICs, however, they suffer from various reliability degradations, such as EM, interfacial delamination, cracking, etc.

EM refers to the migration of atoms in metal interconnect, which forms voids and hillocks, eventually leading to an open-circuit or short-circuit. TSV's EM is driven by multiple factors, including high current density, high mechanical stress, and high temperature gradient. Modeling and integrating this multi-physics system is crucial for monitoring the transient EM behavior and successfully predicting the lifetime of TSVs.

Many reliability loss mechanisms are related to the coefficients of thermal expansion (CTE) mismatch between TSV material (e.g. copper) and silicon substrate. When 3D ICs are cooled from high-temperature manufacturing process to low-temperature operating condition, high amounts of thermal mechanical stresses

are formed around TSVs. These stresses induce stress migration, interfacial delamination between TSV and its liner, and silicon cracking once the stress exceeds the material's yield strength.

Most TSV reliability losses are heavily associated with layout geometry and chip's temperature. For example, TSV's current distribution is decided by how the wires are sized, and TSV EM is significantly accelerated when a higher amount of current flows through. Besides, TSV's stress related reliability losses depend on the magnitude of thermal load. Therefore, TSV's reliability estimation and optimization are often coupled with thermal estimation and optimization.

More importantly, compared to planar circuit, reliability degradations in 3D ICs are relatively more difficult to fix after the electrical design has been done. In planar chip design, redundant wires and vias are widely adopted to enable on-line reconfiguration of interconnects [6]. However in 3D ICs, TSV's high manufacturing cost and low yield make redundant design less feasible than planar circuit. This motivates to develop reliability aware physical design methodologies that simultaneously optimize the performance with circuit reliability.

2.6 DRAM Soft Errors in 3D-CPUs

Three-dimensional integration of stacked memory and CPUs has received many attentions recently for its potential to overcome the "Memory Wall" problem - the memory access time (in CPU cycles) has increased to an extent that the memory access latency has become the bottleneck. Vertical stacking enables heterogeneous

integration of multiple DRAM chips on top of a microprocessor, and the high-speed and wider memory bus interfaces (*i.e.* through-silicon-vias) between the two significantly reduce memory latency.

However, several recent publications have shown that transient fault is one of the common forms of DRAM failures in modern computing and storage systems. One important cause of DRAM transient faults is the power-delivery-network (PDN) noise on DRAM wordline (WL). A noisy WL makes DRAM transistor's gate unable to shut off, causing significant sub-threshold leakage from bit cell's capacitor. As volatile memory, a DRAM bit cell cannot retain its data permanently as the bit capacitor gradually loses its charge. If significant leakage occurs such that DRAM sense amplifier can no longer read the correct data as written last time, a DRAM transient fault happens.

The solutions for mitigating stacked DRAM's transient fault seem non-trivial. An intuitive idea is to enable shorter DRAM refresh period. However, refreshing operation blocks DRAM access, therefore degrades the performance and energy efficiency [81, 82]. Another approach is to throttle CPU performance to cool the chip or/and reduce PDN noise, which again hurts the performance. In our opinion, a promising opportunity lies at the moment when both low and high power tasks enter the system, and designers can exploit the reliability margin of these tasks and allocate different operating points to maximize the performance while achieving certain long-term resilience. A "borrow-in" strategy can be applied, and high-power tasks might borrow resilience margins to boost performance, and the resilience "loan" will be paid off during a low-power task period.

Chapter 3: Low Power 3D Clock Tree Synthesis

3.1 Introduction

The clock tree is one of the largest and most frequently switched networks in 3D IC, making it to be a major contributor to the chip's total power in high-performance VLSI circuit (can take up to 70% of the chip's total power [14]). Naturally we are interested in designing a low-power clock tree for 3D IC. One of the well-known techniques for low-power clock tree design is clock gating. Clock gating exploits the fact that instructions are not executed with even frequency, causing spatial and temporal variations in the "on" and "off" states of the sequential logic. The clock gating technique applies control signal at certain intermediate clock tree node to shut down all its descendants' (wires and sequential logics) clock signal and power supply, when its downstream sequential logics are inactive, thereby reducing the dynamic power dissipated by wires, buffers, and sequential logics.

Clock gating for 2D clock tree has been extensively studied in literature [21–29]. Instruction stream was introduced to calculate the switching probability for each tree nodes, and tree nodes with similar activity patterns were clustered greedily with higher priority [21, 22]. A follow-up work integrated the register-transfer-level (RTL) clock gating approach into industry synthesis tools [23]. Benini *et al.* treated

the circuit as a Finite-State Machine (FSM) and calculated the Observability Don't-Care (ODC) conditions using a formal mathematical formulation [28]. Bolzani *et al.* provided a physical synthesis method to integrate the clock gating and power gating techniques [29]. A deterministic clock gating design was proposed by H. Li and it controls the gating circuitry based on circuit block's actual usage during runtime [24]. W. Chao *et al.* considered clock gating simultaneously with buffer insertion to ensure zero clock skew [25]. J. Lu *et al.* discussed a slew-aware clock gating method [26]. W. Shen *et al.* optimized the placement of clock sinks to boost the power saving brought by clock gating [27].

Recently with the emergence of 3D ICs, 3D clock design has become an important research topic for 3D digital/mixed signal circuit. Generally the 3D clock tree synthesis is a two-step procedure: first the 3D abstract clock tree generation and then the 3D clock tree embedding. The 3D abstract clock tree is a binary tree structure, representing the connectivity from a centralized clock source to all the sequential logics. Currently the 3D abstract clock tree is designed purely from a wire length minimization standpoint. For example, X. Zhao *et al.* implemented a cutting-based approach for abstract clock tree generation, such that the 3D IC was recursively partitioned in X, Y, or Z direction. The location of the cutting line was determined based on the median value of the logic cells' coordinates [2]. Another example is the NN-3D algorithm proposed by Kim *et al.* [31]. The authors greedily picked the closest sequential pairs and recursively formed the abstract clock tree. During the 3D clock tree embedding step, the physical locations of the intermediate clock tree nodes as well as the locations of *clock* TSVs are determined such that the

clock tree's total wire length and clock skew are minimized. For example, Kim *et al.* extended the 2D deferred-merge-embedding (DME) algorithm [20] to the 3D space in order to decide the length of clock tree edges and the locations of *clock* TSVs to ensure minimum wire length and zero-skew in 3D clock tree [31]. In addition, Yang *et al.* investigated the impact of the TSV-induced stress on timing corners and optimized the location of clock buffers to minimize the stress-induced clock skew variation [32]. Lung *et al.* considered the reliability issues of *clock* TSVs and used TSV fault-tolerant unit (TFU) to enhance clock tree's robustness against TSV failures [33]. Further tuning techniques for the TFU and associated fault-tolerant 3D clock tree, such as better control of clock slew, TSV count, and clock slew were proposed by Park *et al.* [34].

The primary concern of this chapter is the design of the 3D low-power clock tree using the clock gating technique. We notice that the conventional 2D clock gating approach can not be applied directly to 3D IC, for the reason listed as follows. In a gated clock tree, the shutdown network provides enable signals for the shutdown gates. The shutdown network is a planar *Star* network (one centralized control center delivers the enable signals to all shutdown gates through separated wires). In 2D-IC, the wiring overhead of the shutdown network is usually ignored, and the authors of the 2D clock gating papers assume they can always deliver the enable signals to wherever needed [21–29]. Therefore, these works insert control gate anywhere as long as there is power saving. However, the assumption that designers can insert shutdown gates at every tree edge and the wiring overhead of the *Star* network is negligible is no longer valid for 3D clock tree.

In 3D clock tree, *clock* TSVs deliver clock signals from clock source to each of the clock sinks while *control* TSVs provide shutdown signals from a centralized control center to all shutdown gates [35]. Since clock synthesis is usually performed after cell placement, layout whitespace for TSVs is limited. However, both TSVs (*clock* and *control*) occupy large placement area and the reliability [36] and signal integrity [37] requirements enforce TSVs to maintain certain keep-out area, both of which constrain the usage of TSVs. In addition, although TSVs offer short and fast vertical connections, excessive usage of TSVs increases the manufacturing overhead and makes the system's reliability questionable, due to the degradation of TSVs over time. The restriction that limited numbers of *clock* TSVs and *control* TSVs are available to designers changes the way how the 3D clock tree should be synthesized and how the clock gating technique should be applied.

Aiming at a low-power 3D clock tree that accounts for the TSV usage constraints, we believe that both the 3D abstract clock tree design as well as the 3D clock tree embedding procedure need to be optimized. Firstly, the 3D abstract clock tree needs to account for not only the total wire length but also the similarities in sequential logic's "on" and "off" behavior, so that the maximum amount of power saving can be achieved with limited number of shutdown gates. Secondly, during the clock tree embedding step, we need to decide at which tree edge to put shutdown gates and the physical locations of *clock* TSVs and *control* TSVs such that the all the TSVs can fit into the layout whitespace, clock skew is zero, and the power saving is maximum.

To summarize, this chapter focuses on the problem of designing a low-power clock 3D clock tree using the clock gating technique, while satisfying the layout whitespace constraint and zero clock skew. The designs of the 3D abstract clock tree, the shutdown clock network, and how the shutdown network affects the design of the clock tree embedding step are investigated and optimized. In the following sections, we show our methodology to smartly generate the 3D abstract tree that is preferable for shutdown gate insertion, and select and allocate shutdown gates such that all the TSVs can be legally placed inside layout whitespace, clock skew is zero, and clock tree power is minimized. More specifically, this chapter makes the following contributions:

- We identify the problem of layout constrained shutdown network design for 3D ICs, and discuss its significant difference from 2D clock gating techniques.
- We propose a two-step design flow for low-power 3D clock tree synthesis, which comprises of the abstract clock tree generation step and the clock tree embedding step.
- During the abstract clock tree generation step, we propose a K-means clustering [38] based algorithm to cluster nodes which are close to each other and have similar activity patterns.
- During the clock tree embedding step, we develop an SA based heuristic to find at which tree edge to insert shutdown gates and to decide both the *clock* and *control* TSVs' locations by a force-directed TSV placer such that the overall power consumption is minimized.

The organization of the this chapter is as follows. We provide an overview of related preliminary studies on 3D clock tree's power model in Section 3.2. The follow-up section, Section 3.3 focuses on optimizing the 3D abstract tree topology. Section 3.4 focuses on optimizing the clock tree embedding procedure where we formulate and solve two consecutive problems to decide at which tree edge to put shutdown gates and the placement of *clock* TSVs and *control* TSVs. Experimental results are presented in Section 3.6 and Section 6.6 concludes the chapter.

3.2 Preliminary

Some basic models and concepts are introduced in this section. First we briefly cover the 3D clock tree's power models. Then we illustrate the basic idea of obtaining activity pattern for each clock tree node. After that we introduce the TSV's legal placement problem in 3D IC, and show that constrained layout whitespace impacts the decision of gate insertion thus completely changes the clock tree design flow. Finally, an example of a 3D gated clock tree is shown to shed light on the influence of 3D IC's layout whitespace constraint on clock gating strategy and power consumption.

3.2.1 Power model

Clock tree power consists of two components: dynamic power and static power. Dynamic power is the summation of the following three, as shown in Equation 3.1: clock sink(module)'s power P_m^T, clock tree's wiring power P_w^T, and controller net-

work's wiring power P_w^C. Static power is consumed by the buffers along a long signal or power wire, and any synchronized CMOS circuit. Shutdown signals can not only be applied to shutdown gates but also power delivery network (PDN) to shut down the power supply thus static power of buffers along power wire can also be reduced. Since the estimation of static power along power wire needs thorough investigation of how the chip's PDN is synthesized, and CMOS's static power is largely decided by device technology, we focus on the dynamic power of the clock network in this chapter.

$$Power = P_m^T + P_w^T + P_w^C. \tag{3.1}$$

where P_m^T denotes the overall power of the clock sinks (modules), P_w^T is the clock tree's wiring power, and P_w^C is the controller network's wiring power.

The dynamic power of a clock sink(P_m^T) is expressed using Equation 3.2. It is the weighted sum of its *active circuit power* P_A, when clock is applied and the clock sink is active, *standby circuit power* P_S, when clock is applied and clock sink is idle, and *idle circuit power* P_I, when clock is not applied and clock sink is idle. The weight for each power component is the probability that the clock sink is active, standby, and idle, respectively.

$$P_m^T = \sum_{i=1}^{n} [P_{A_i} \cdot p_{A_i} + P_{S_i} \cdot p_{S_i} + P_{I_i} \cdot p_{I_i}]. \tag{3.2}$$

where n is total number of clock sinks, p_{A_i}, p_{S_i}, and p_{I_i} is the probability that clock sink i is active, idle but clocked, and idle and masked during one cycle, respectively.

Clock tree and controller network's wiring powers are caused by charging and discharging the wire capacitance. We ignore the controller network's wiring power(P_w^C) in this chapter because the signal in controller network switches far less frequently than the clock signal, which turns on/off twice within one cycle. However, our power model can easily handle ad hoc designs that require frequent switching of the control signal as well.

The dynamic power consumed by the clock tree's wiring can be described as Equation 3.3:

$$P_w^T = \sum_{\forall e_i} \frac{1}{2}(c_w|e_i| + C_i)\alpha f V_{dd}^2. \tag{3.3}$$

where e_i, c_w, C_i, α, f and V_{dd} denotes a clock tree edge, wire's unit capacitance, node i's capacitance, switching activity ratio, clock frequency, and supply voltage level.

3.2.2 Activity Pattern

In this section, we show the methodology for obtaining the activity pattern of each sink module and clock tree edge. We assume each sink module has registers with clock signal on its input. The activity pattern of a sink module is a boolean string where "1" indicates clock signal is applied and "0" means it's shut down. When it is on "0", the sink module is gated and it only consumes *idle circuit power* P_I. On the contrary, when it is on "1", the sink module consumes *active circuit power* P_A or *standby circuit power* P_S, depending on whether data are fed or not.

In other words, P_S is the amount of power dissipated on the capacitive components inside the module when clock is still "on" even though the module is not doing any computation.

If the clock tree is completely gated (clock gate is inserted at every clock tree node), the activity pattern of an internal clock tree node (A_i) can be obtained by ORing (bit-wise OR) the activity patterns of all its descendant clock sinks (S_1, S_2, ..., S_k). This procedure (Phase 1) is illustrated in Equation 3.4.

$$A_i = S_1 \bigcup S_2 \bigcup ... \bigcup S_k \tag{3.4}$$

If the clock tree is partially gated, then after obtaining the activity pattern for a completely gated clock tree (Phase 1), by pre-order traversal, from the clock source to all clock sinks, a clock tree node inherits its parent's pattern if gating is not applied between that node and its parent, otherwise stays unchanged. We call this procedure Phase 2, as shown in Equation 3.5.

$$\hat{A}_i = (\hat{A}_p * \bar{G}_i) \bigcup A_i \tag{3.5}$$

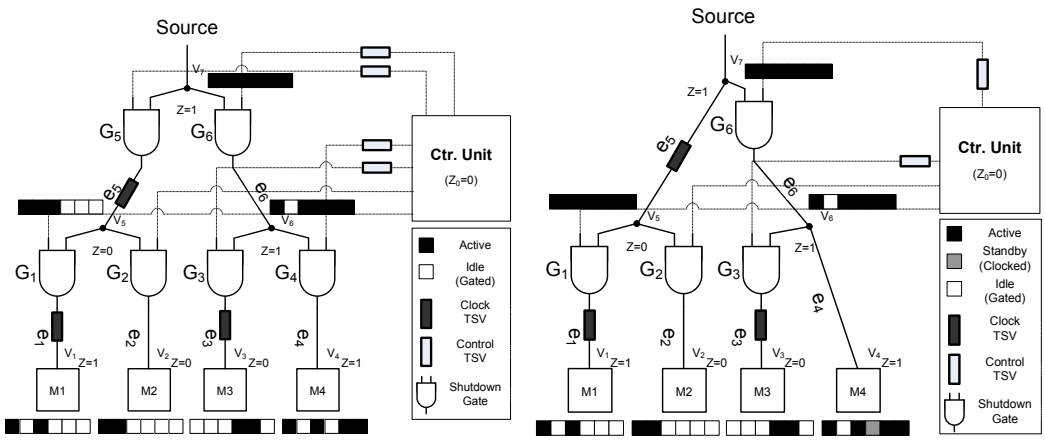
where $\hat{A}_p$ is the activity pattern of i's parent clock tree node, A_i is i's activity pattern obtained from Phase 1, and G_i is a boolean variable which is 1 when clock gate is applied between node i and its parent and 0 otherwise.

3.2.3 An example: minimizing clock power with TSV placement constraint

Figure 3.1(a) and Figure 3.1(b) are two 3D clock tree designs. Each design contains one clock source, seven nodes ($V_1 - V_7$), and four clock sink modules ($M_1 - M_4$). "AND" gate is used as shutdown gate. Each module is associated with one activity pattern, recording the module's activity over time. A module is considered as active when it is clocked and performing calculation (its activity pattern is filled/black). Otherwise it is in standby mode when it's clocked but performs no calculation (activity pattern is gray), or in idle mode if it is unclocked and performs no calculation (activity pattern is unfilled/white). Definition of internal node's activity pattern is slightly different from module's. Activity pattern for internal node is filled for "1" (active), or unfilled for "0" (idle). Activity patterns for internal nodes are calculated by Phase 1 and Phase 2 described in Section 3.2.2. Figure 3.1(a) shows Phase 1 where clock tree is completely gated, meaning shutdown gates are inserted at every clock tree node. Each internal node's activity pattern is obtained by OR-ing the activity patterns of its children. Since we ignore the power consumption of

the controller network, complete clock gating will achieve the optimal power saving. However, a complete clock gating might not be feasible for 3D ICs due to its high usage of *control* TSVs.

Figure 3.1(b) shows another design which uses less *control* TSVs. Clock tree edge e_4 and e_5 are not gated. According to Phase 2 in Section3.2.2, node V_5 inherits V_7's pattern but node V_6's pattern stays unchanged (compared to V_6's pattern in Figure 3.1(a)) because of the shutdown gate on edge e_6. Meanwhile, the forth state in M4's pattern is gray, which corresponds to the situation where a module is clocked but is not doing any computation.



(a) (b)

Figure 3.1: Suppose the control center that sends out enabling signals is at layer $Z_0 = 0$ and the maximum number of TSVs allowed is 5. Design (a) has 7 "AND" gate and 7 TSVs, which is infeasible to place. Design(b) uses only 5 TSVs while still achieves considerable power saving.

Table 3.1 compares the power consumptions for the designs in Figure 3.1(a) and Figure 3.1(b). We assume module's active power $P_A = 30$, standby power $P_S = 5$, and idle power $P_I = 3$. We also assume the wiring power is 100 for each wire. "AND" gate and TSV consumes 2 and 8 unit power, respectively. The power

Table 3.1: Illustrative power consumption

Power Contribution	Complete Gating Figure 3.1(a)	Partial Gating Figure 3.1(b)	No gating
Planar Wiring	400.0	466.7	700.0
TSV	9.3	13.3	24.0
Gate	12.0	8.0	0.0
Module	57.0	57.3	61.7
Total	478.3	545.3	785.7
Saving(%)	39.1	30.6	0.0

values for the TSVs and the wires are scaled based on TSV capacitance values ($50fF$) reported by previous works [2, 30, 39] and global wire capacitance reported by ITRS roadmap (assuming 1000 μm long global wire) [40], respectively.

Firstly, we calculate the overall power consumption for both designs and we find that Figure 3.1(b) uses less TSVs but still achieves considerable power saving compared to Figure 3.1(a), as shown in Table 3.1.

Secondly, we show that blindly inserting gates may end up with infeasible TSV placement. For illustration purpose, we assume we can fit at most 5 TSVs into the layout whitespace. (In practice, we use a force-driven placer to legalize all TSVs.) Both designs in Figure 3.1(a) and Figure 3.1(b) use 3 *clock* TSVs. Figure 3.1(a) inserts shutdown gates at every possible locations, resulting in 4 *control* TSVs (on gate G_3, G_4, G_5, G_6), which becomes an infeasible design under the layout whitespace constraint. On the other hand, Figure 3.1(b) only uses 2 *control* TSVs, which makes it a feasible design.

The main observation we have from this example is that clock gating for 3D clock tree is quite different from 2D-IC's case: because of the area overhead of TSVs, the design of the shutdown network in 3D clock tree must be re-considered along with the layout information. There is a trade-off between TSV layout whitespace (or the usage of TSVs) and maximum power saving, and deciding where to insert the shutdown gate is the key to address this problem.

3.3 Abstract 3D clock tree generation

Abstract clock tree generation is the first step of the clock tree synthesis. An abstract tree is a tree topology, whose leaves represent all sequential modules (clock sinks), and the root represents the clock source. The clock signal flows from the clock source, through the edges, to all clock sinks. In this chapter, we assume the abstract clock tree is fully binary.

Most existing 3D abstract clock tree generation methods focus on minimizing the total wire length and total TSV usage, while the clock skew is minimized in the subsequent clock tree embedding step (see Section 3.4). There are three existing algorithms to generate 3D abstract clock tree: 3D-MMM (method of means and medians) [2], MMM-3D [30] and NN-3D (nearest neighbor selection for 3D-ICs) [31]. The 3D-MMM method is an extension of the 2D-MMM algorithm proposed by M. Jackson *et al.* [17]. For a given TSV bound, the 3D-MMM algorithm partitions all the clock sinks horizontally (X/Y-cut) or vertically (Z-cut). During X/Y-cut, the clock sinks are projected to a 2D plane and then separated evenly, while in Z-cut, the

clock sinks are partitioned such that the sinks from the same chip belong to the same subset, and a TSV is inserted between adjacent chips. The MMM-3D method [30] uses a designer specified parameter to control the partitioning direction (X/Y cut or Z-cut) between sinks. The NN-3D method extends Edahiro's 2D bottom-up approach [41]. NN-3D recursively selects a pair of subtrees or clock sinks and then merge that pair. The selection criterion is based on a cost function which is a weighted sum of 2D Euclidean distance, TSV number, and capacitive load.

While these three algorithms are promising in generating low wire length 3D abstract clock tree, they are not suitable for generating low-power 3D abstract clock tree. Although minimizing the total wire length does reduce the wire power consumption, its power saving is sub-optimal. In addition, neither of these three algorithms is capable of handling the unique challenge in 3D clock tree design that the shutdown gates can no longer be inserted at every tree edge. As we have explained in Section 6.1, this is because the TSV placement whitespace is limited during the clock synthesis step, and excessive usage of TSVs brings extra manufacturing cost and raises TSV's reliability concerns. Therefore, we aim to generate a 3D abstract clock tree such that the minimum number of shutdown gates is needed while simultaneously reducing the wire length and the power consumption of the clock tree. To achieve this goal, it is desirable that: (1) clock sinks with similar activity patterns are clustered together (belong to the same subtree), and (2) clock sinks that are next to each other are clustered together. It's preferable to cluster sinks with similar activity patterns because a shutdown gate can turn off the entire subtree when none of the clock sinks in that subtree is active. It stays on (thus the entire subtree

consumes dynamic power) even though there's only one clock sink in that subtree is active. Clustering similar activity patterns ensures maximum shutdown rate and also reduces the need for the shutdown gates and therefore potentially reduces the number of *control* TSVs. Meanwhile, clustering clock sinks that are next to each other minimizes the total wire length thus minimizes wire's dynamic power.

To summarize, the 3D-MMM and NN-3D algorithms only consider total wire length as an objective, but neither of them consider the dynamic power consumption during the charging and discharging behaviors happened on the capacitive loads. As far as the clock gating technique is concerned, we believe the quality of a gated low-power 3D clock tree heavily depends on its abstract clock tree. Therefore, in the following subsection we propose a K-means clustering based algorithm to generate a low-power 3D abstract clock tree, which produces a high-quality abstract clock tree that is suitable for the subsequent low-power clock tree embedding.

3.3.1 K-means clustering

As mentioned earlier, a low-power 3D clock tree geared towards the clock gating technique should capture both the similarities between activity patterns, and the physical proximity between clock sinks. These "similarity" in activity patterns and physical locations can be captured by clustering algorithms, and we select the well-studied K-means clustering technique to cluster the clock sinks. The K-means clustering algorithm divides up a set of points into K clusters, such that the sum of the within-cluster dissimilarity is minimized. In our binary abstract clock tree's

case, $K = 2$. The with-in cluster dissimilarity is defined as the average distance between two points within a cluster. We also associate each clock sink S_i with a tuple: $(x_i, y_i, z_i, C_i, A_i)$, where x_i, y_i, z_i are the location, C_i and A_i are sink S_i's capacitive load and activity pattern.

Specifically our K-means clustering algorithm alternates between two steps: the assignment step and update step.

3.3.1.1 Assignment Step

The assignment step assigns each clock sink to the closest clustering center. During the assignment step, the distance function that accounts for activity pattern's similarity and physical proximity between two clock sinks i and j (or two subtrees' roots) is defined as the dynamic power consumption of a subtree if node i and j are children of the same subtree's root. This formulation is better than clustering clock sinks purely based on their activity patterns, or physical locations, because these two might be contradictory objectives. For example, clock sinks that are far away might possess highly similar activity patterns, and clock sinks that are next to each other possibly have distinct activity patterns. Our proposed distance function combines the *best* of both worlds therefore is used in this chapter. Specifically the distance function d_{ij} is defined as in Equation 3.6.

$$d_{ij} = \frac{1}{2}\Big[c_w\big(|x_i - x_j| + |y_i - y_j|\big) + c_v|z_i - z_j| + C_i + C_j\Big]\alpha_{ij} f V_{dd}^2 \tag{3.6}$$

where $x_{i,j}$, $y_{i,j}$ and $z_{i,j}$ is $i, j's$ location, and c_w, c_v, $C_{i,j}$, $\alpha_{i,j}$, f and V_{dd} denote wire's unit capacitance, TSV's unit capacitance, node $i, j's$ loading capacitance, switching activity ratio, clock frequency, and supply voltage level, respectively. Particularly, α_{ij} is defined as the possibility that at least one of the two nodes (node i and j) is "on" during a sampling time period:

$$\alpha_{ij} = \frac{sum(\bar{A}_i \cup \bar{A}_j)}{m} \tag{3.7}$$

where m is the length of the activity pattern, and $\bar{A}_i$ is the estimated activity pattern of node i, which is the average between two extreme cases (as shown in Equation 3.8): a completely gated clock tree and a clock tree with no clock gating. Since at the 3D abstract clock tree's generation stage, we know neither the physical implementation of the clock tree, nor the exact location of shutdown gates, therefore we don't know the exact activity pattern of each clock sink, we believe that taking the average of an ungated clock tree and a fully gated case to represent node i's activity pattern, as shown in Equation 3.8, is a good estimation.

$$\bar{A}_i = \frac{1}{2} \cdot (A_i + \hat{A}_i) \tag{3.8}$$

where A_i is node i's activity pattern when the clock tree is completely gated (shutdown gates inserted at every tree edge), and Figure 3.1(a) is an example of a fully gated clock tree. A_i can be obtained by simply ORing (bit-wise OR) the activity patterns of all its descendant clock sinks (refer to Phase 1's Equation 3.4 for more details). $\hat{A}_i$ is node i's activity pattern when the clock tree is not gated (no clock

gating at all). In that case, according to Phase 2's Equation 3.5, every G_i is zero, which means every node inherits its parent's activity pattern therefore every node's activity pattern is identical to the activity pattern of the tree root. The root's activity pattern can be obtained by bit-wise ORing all the clock sinks' activity pattern.

3.3.1.2 Update Step

The update step replaces each clustering center by the average of clock sinks in its cluster. During the update step, the new cluster center is generated by averaging out the tuples belong to the clock sinks in the same cluster: $(x_c, y_c, z_c, cap_c, a_c) = (\bar{x}_i, \bar{y}_i, \bar{z}_i, \bar{cap}_i, round(\bar{a}_i))$. The bit-wise average of activity pattern represents the most frequently appeared activity pattern among the clock sinks in the same cluster.

3.3.2 Our abstract tree generation algorithm

The abstract clock tree generation algorithm is summarized in Algorithm 1. The inputs of the algorithm are the clock source's layer index (z_src), TSV bound (B) and a set (S) representing the clock sinks. There are two terminating conditions: (1) When there's only one clock sink in S (line 10), and the algorithm returns with this clock sink (line 11); (2) When there's only one TSV per layer available ($B = 1$) and the clock sinks in S span over more than two dies ($sizeof(Z) > 1$), and the algorithm bi-partitions S into two sub-trees S_1 and S_2. If z_src is contained in the range of $[min(z), max(z)]$, then S_1 contains all the clock sinks that have layer index larger than z_src (line 14) while S_2 contains the rest. Otherwise S_1 contains all the

clock sinks that are located at the layer which is next to the z_src (line 16), while S_2 contains all other clock sinks. When neither of these two terminating conditions are satisfied, the K-means clustering algorithm bi-partitions S and B to minimize the clock tree power and wire length (line 21). Then our function "AbstractTree3D" is called recursively on (B_1, S_1) and (B_2, S_2) and the roots of S_1 and S_2 become the current tree node, $root(S)$'s children.

Input:
1. TSV bound B.
2. Clock source's layer index z_src.
3. A set (S) representing clock sinks, which contains:
4. Physical location of each clock sink (x, y, z);
5. Activity pattern of each clock sink (A);
6. Capacitive load of each clock sink (C).
7. Clock source's z index (z_{src})

Output:
8. Binary abstract tree.
9. AbstractTree3D (B, S)
10. **if** $|X| = |Y| = |Z| = |A| = |C| = 1$
11. return root (S)
12. **else if** $B = 1$ and $sizeof(Z) > 1$
13. **if** $min(z) \leq z_{src} \leq max(z)$
14. $S_1 = \{s | z_i \geq z_{src}\}, S_2 = \{s | s \notin S_1\}$
15. **else**
16. $S_1 = \{s | \, |z_i - z_{src}| = min(|z_i - z_{src}|)\}$
17. $S_2 = \{s | s \notin S_1\}$
18. **end**
19. $B_1 = B_2 = 1$
20. **else**
21. $[S_1, B_1, S_2, B_2]$ = K-means clustering (S)
22. **end**
23. AbstractTree3D($B1$, $S1$)
24. AbstractTree3D($B2$, $S2$)
25. $LeftChild(root(S)) = root(S_1)$
26. $RightChild(root(S)) = root(S_2)$
27. return $root(S)$.

Algorithm 1: Abstract Tree Generation

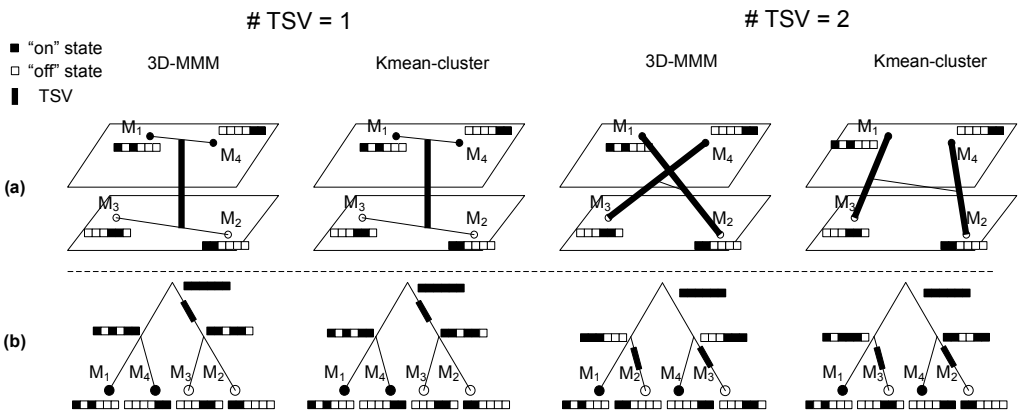


Figure 3.2: The 3D abstract tree generated by our K-means clustering based algorithm and the 3D-MMM algorithm [2], for a given TSV bound. (a) 3D view where thick lines represent TSVs, each clock sink has a layer index (black dot indicates upper layer, and while dot is for lower layer), and each clock sink is associated with an activity pattern (black square is "on", while white square is "off"). (b) The resulting 3D abstract trees where the black rectangles represents TSVs.

Figure 3.2 shows a simple example for 3D abstract clock tree generation. In Figure 3.2, 4 modules (M_1 to M_4) are distributed on a two-layer 3D IC. Each module has its activity pattern. When the TSV bound is 1, both the proposed K-means clustering based method and the 3D-MMM algorithm generate the same abstract clock tree: the modules on the same layer belong to the same subtree. However, when the TSV bound is more than 1, for instance, 2 in Figure 3.2, the 3D-MMM algorithm only clusters nearest modules, while the K-means method takes both the location and the activity pattern's information into account. The K-means method generates a 3D abstract clock tree that is suitable for shutdown gate insertion, which we explain in Section 3.4.

It's worth mentioning there exists a greedy algorithm called NN-3D [31] to generate 3D low-power abstract clock tree. The NN-3D algorithm tries to minimize the number of TSVs, the total wire length, and overall power consumption. During

each iteration, a pair of clock sinks (or subtree roots) with the lowest distance (distance defined as in Equation 3.6) are extracted from the clock sink set S, and then form the left and the right child of a subtree. Then these two sinks (or subtree roots) are deleted from S and the their parent node with an updated location, capacitive load and activity pattern is returned to S. Similar iterations carry on until there's only one subtree root in S.

Although the NN-3D is a promising technique, it overlooks one aspect that a designer may need to restrain the number of TSVs in the 3D clock tree, due to TSV's large size, manufacturing cost, reliability issues *etc.* In other words, we find NN-3D algorithm is incapable of handling the TSV bound constraint. It might be possible to adaptively tune the weighting parameters in Equation 3.6 to restrain the usage of *clock* TSVs, however, we haven't seen such tuning techniques in the literature. In addition, the greedy approach somehow loses the global view as in each iteration only two clock sinks (or subtree roots) are processed. Therefore we believe our K-means clustering based algorithm is better at handling the TSV number-aware 3D abstract tree construction problem. In the result section, we'll compare our K-means clustering based algorithm with 3D-MMM.

3.4 Clock tree embedding

For a given 3D abstract clock tree, the second phase of 3D clock tree synthesis, the wire embedding, determines the physical locations of the intermediate clock tree nodes, in order to minimize objective functions such as total wire length, clock skew,

clock power, and *etc.* In this section, we introduce the conventional clock skew and wire length-centered embedding method, and then we propose two new problems for 3D low-power clock tree generation, under certain placement whitespace constraint. The first problem asks the best locations for shutdown gate insertion, after applying the conventional clock tree embedding approach, while the second tackles the question whether the clock tree can be re-designed to accommodate the choice of shutdown gates.

3.4.1 Conventional embedding method

A prevailing objective for 3D clock tree embedding is to minimize the total wire length while ensuring zero clock skew. Kim *et al.* proposed a DME-3D (Deferred-merge-embedding) algorithm [31] to produce a 3D clock tree with near-optimal wire length. The DME-3D algorithm is an extension of the 2D-DME work proposed by Chao *et al.* [20]. In both DME algorithms, following a bottom-up fashion, the location of each intermediate clock node is investigated. With zero clock skew as constraint and minimum wire length as objective, the DME algorithm finds the solution space, namely, the "merging segments" (MS) for each of the intermediate tree node from clock sinks all the way to the clock tree root. After that, starting from the clock tree root, a pre-order traversal decides the exact location of each intermediate tree node such that the Manhattan distance from the intermediate node to its parent node is minimized. Then the algorithm inserts *clock* TSVs when a tree node is not at the same layer as its parent node. It is provable that the *clock*

TSV should be placed at the same location as its parent node to achieve minimum wire length. In a word, given a 3D abstract clock tree, the DME-3D algorithm [31] generates a 3D clock tree with low wire length and ensures zero clock skew. One other notable clock tree embedding technique includes the sDMBE algorithm [2] that performs simultaneous clock buffering to control the clock slew during the DME-3D procedure.

3.4.2 Clock gating for 3D low-power embedding

In this section, we formulation two problems for 3D low-power clock tree embedding. Let $M = \{M_1, M_2, ..., M_n\}$ represents clock sinks (modules). The location of each clock sink is (x_i, y_i, z_i). Another input is the 3D abstract clock tree, which is generated using the K-means clustering based algorithm proposed in Section 3.3.1. Let $V = \{v_1, v_2, ..., v_{2n-1}\}$ denote the nodes of the abstract clock tree, where $v_1, v_2, ..., v_n$ refer to leaf nodes (clock sinks) and the rest are internal tree nodes. We call the tree edge between v_i and its parent e_i. Each edge e_i is a potential location of a shutdown gate, G_i, and G_i is able to shut down the subtree rooted at v_i when necessary. If G_i is inserted at e_i, we always insert G_i at the end of the edge, next to v_i's parent so that we can save e_i's power consumption when necessary. We assume one control unit is located at the center of layer Z_0. The control signal paths form a *Star* network (each control gate connects to a centralized control unit through a separate wire). It's worth mentioning that the *Star* network is one simple but effective connecting structure. More sophisticated structures can be applied to

avoid excessive wiring capacitance. For example, a designer may exploit the opportunities that multiple shutdown gates with same enabling signals could share a common control signal path. Or one can add multi-layer control units in multiple layers and the "enable" signals are issued from the control units to the clock sinks in the same layer. The comparison between single control unit design and multiple control units design is elaborated in Section 3.5. The following section assumes the control signal network is a *Star* network. We investigate two problems.

3.4.2.1 Problem 1: Gate insertion under constrained placement of *control* TSVs

Given a 3D ungated clock tree generated by conventional wire embedding technique (DME-3D [31] for instance), Problem 1 investigates where to insert the shutdown gates among all $2n-1$ gating candidates, (n = total number of clock sinks) such that the dynamic power of the gated clock tree is minimized while the *control* TSVs can be legally placed. As shutdown signal requires *control* TSV's connection to reach the control center and TSV occupies large placement area, inserting shutdown gate at every tree edge is impractical in 3D IC. In addition, some tree edges don't need shutdown gates. For example, in Figure 3.3, if the children of a clock tree node possess the same activity patterns, there's no need to insert shutdown gates at these edges because these children nodes switch "on" and

"off" simultaneously. Another example could be in a situation where certain nodes are always "on" (activity patterns are mostly "1"s). In that case, clock gating is ineffective either.

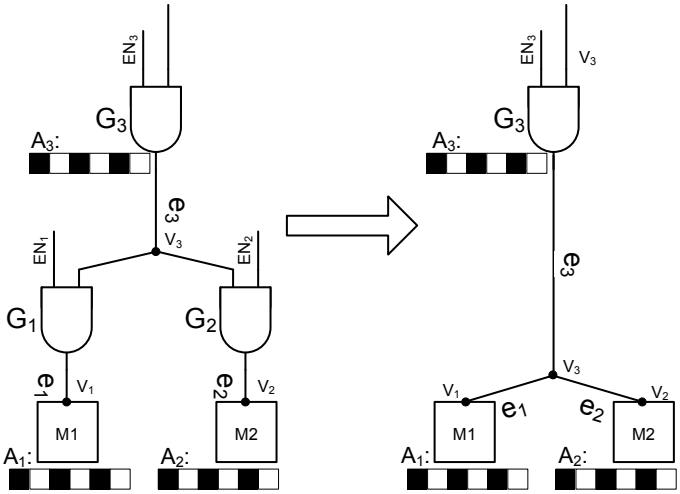


Figure 3.3: M_1 and M_2 have identical activity patterns, thus the shutdown gates G_1 and G_2 are unnecessary.

In Problem 1 we fixed the locations of *clock* TSVs, thus the *clock* TSVs are treated as placement blockages as we place the *control* TSVs.

3.4.2.2 Problem 2: 3D clock tree synthesis with TSV and *control* TSV co-placement

Given a 3D ungated abstract clock tree, Problem 2 aims to construct a low-power 3D clock tree with simultaneous gate insertion, under the constraint that the *clock* TSVs and *control* TSVs need to be legally placed inside the layout whitespace. Meanwhile, we also need to minimize the total wire length and clock skew.

Different from Problem 1, which performs shutdown gate insertion based on a fully embedded clock tree, Problem 2 makes one step forward, claiming that clock tree embedding procedure should accommodate the placement of the shutdown gates and *control* TSVs. This is because when the locations of all internal tree nodes, especially the location of the *clock* TSVs is completely fixed and the *clock* TSVs are purely treated as placement blockages, *control* TSVs may not be able to effectively utilize all the placement whitespace. By re-locating the *clock* TSVs, more space is created for *control* TSVs thus more shutdown gates are available for insertion. However, as the DME-3D algorithm [31] generates 3D clock tree with near-optimal wire length, relocating the *clock* TSVs may increase the clock tree's overall wire length, so we impose certain boundaries to restrict the movement of *clock* TSVs to reduce the wire length overhead.

3.4.3 Methodology

In this section we present our methodology to solve both Problem 1 and 2. We first develop a force-directed TSV placer to place the TSVs within the given placement whitespace, and then propose an SA-based approach as a heuristic to solve Problem 1 and 2. As the SA procedure could be time-consuming, we discuss approaches to speed up the heuristic.

3.4.3.1 TSV placement

The number of *control* TSVs, which carry the shutdown gates' control signals, is proportional to the number of the shutdown gates used. However, since clock tree synthesis is usually performed after the placement of blocks/standard cells/IO pins, the layout whitespace left for TSVs is limited. Moreover, TSVs have larger dimensions than standard cells, for example, a TSV could occupy $5\mu m \times 5\mu m$ to $30\mu m \times 30\mu m$ in area, depending on different fabrication technologies, as compared to less than $1\mu m \times 1\mu m$ standard cells at sub-micro technology node. In addition, TSVs have to maintain certain distance from each other, due to mechanical [36] and signal integrity [37] considerations. The TSV placement problem is to place the *clock* TSVs and *control* TSVs such that each TSV is located inside the layout area and outside other TSVs' keep-out-zone (KOZ).

We adopt the force-directed placement idea in [42] to solve the TSV placement problem. Overlap between TSVs, and overlap between TSVs and placement boundaries produce forces to move TSVs to the overlapping-free regions during each iteration. The magnitude of the force is proportional to the area that overlaps. The direction of force is the combination of two kinds of overlap: (1) the overlap between *control* TSVs and other TSV's KOZ; (2) the overlap between *control* TSVs and the boundary region. Figure 3.4 illustrates these two forces. Here we assume the clock tree is given so the *clock* TSVs are fixed and treated as placement obstacles (more details in Section 4.4.1). We later make all *clock* TSVs and *control* TSVs movable in Section 3.4.3.3.

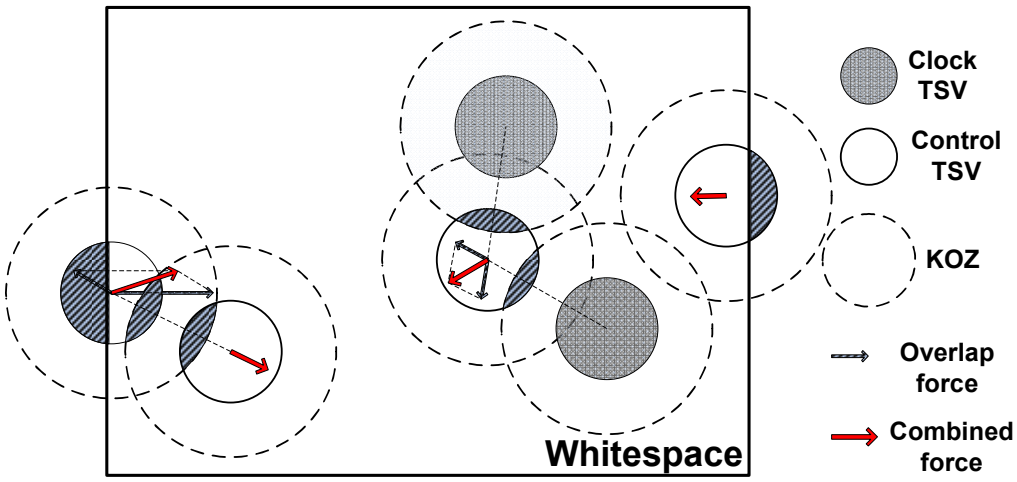


Figure 3.4: Force-directed placer to place *control* TSVs inside the TSV layout whitespace.

3.4.3.2 Problem 1

As mentioned in Section 3.4.2.1, Problem 1 treats *clock* TSVs as fixed placement obstacles. We use the force-directed TSV placer to iteratively move *control* TSVs to remove overlaps. The force-directed TSV placer is integrated into an SA framework, which aims to find a high-quality selection of *control* TSVs that could fit into current chip layout, while minimizing the total clock power.

An exhaustive search approach to decide at which tree edge to put shutdown gates takes at least $O(2^n)$, where n is the number of clock sinks. For a modern synchronizing processor, n varies from several hundreds to several millions, depending on the granularity of the design. In this chapter, we proposed an efficient heuristic based on SA to solve Problem 1, as shown in Figure 3.5. It starts with an estimation *control* TSVs number that can fit into the whitespace. Then a set of shutdown gates are randomly selected to form the initial state. Iteration begins as we try to fit the

selected *control* TSVs into the current layout whitespace, using the force-directed placer. If the placement fails, we reject the current selection immediately. Then we remove one *control* TSV in high overlapping area, hoping the remaining *control* TSVs can be legally placed in next iteration. Otherwise, if the placement is successful, SA process probabilistically accepts or rejects the current selection, based on the clock tree's total power consumption. If current selection is accepted, we add one more *control* TSV, hoping it'll bring us more power saving in the next iteration. If the selection is rejected due to an increase in clock power, we substitute one *control* TSV with one unused *control* TSV. We update the annealing temperature afterwards. When the annealing procedure converges (no significant improvement is seen in several consecutive iterations), a high-quality set of shutdown gates is chosen, which achieves maximum power saving and all the TSVs can be legally placed within the layout whitespace.

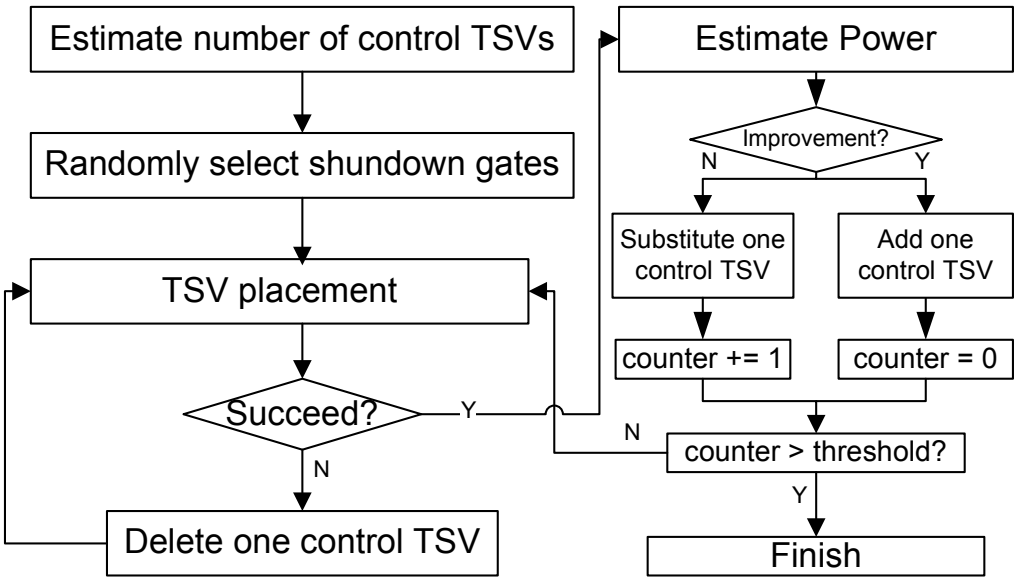


Figure 3.5: Flowchart of SA-based approach for solving Problem 1.

3.4.3.3 Problem 2

When solving Problem 1, we notice that when the location of *clock* TSVs is fixed, layout area cannot be fully utilized. Therefore, we modified the force-directed placer to place *clock* TSVs simultaneously with *control* TSVs.

We adopt similar SA based iterative framework to solve Problem 2. The primary modification is that after each successful TSV placement, the clock tree itself needs to be re-synthesized to meet the zero-skew constraint, and the clock tree's wire length might increase slightly, assuming the ungated clock tree already achieves minimum wire length. The time complexity of DME-based clock tree synthesis is bounded by O(n) where n is the number of clock sinks [31]. So this extra synthesis step won't increase the time complexity of our SA based approach asymptotically. In order to control the growth of total wire length, the objective function in SA is modified to be the weighted sum of the total clock power and total wire length.

3.4.3.4 Setting up initial state

SA's solution quality and runtime heavily rely on the initial state. In order to speed up our SA based algorithm, we develop a snippet to setup the initial state before SA begins. We start with an ungated routed zero-skew clock tree, and estimate the *control* TSV's count that can fit into the current layout whitespace, and then iteratively select the gate with maximum power saving. The selection continues until the *control* TSV count reaches our early estimation. The procedure is summarized in Algorithm 2.

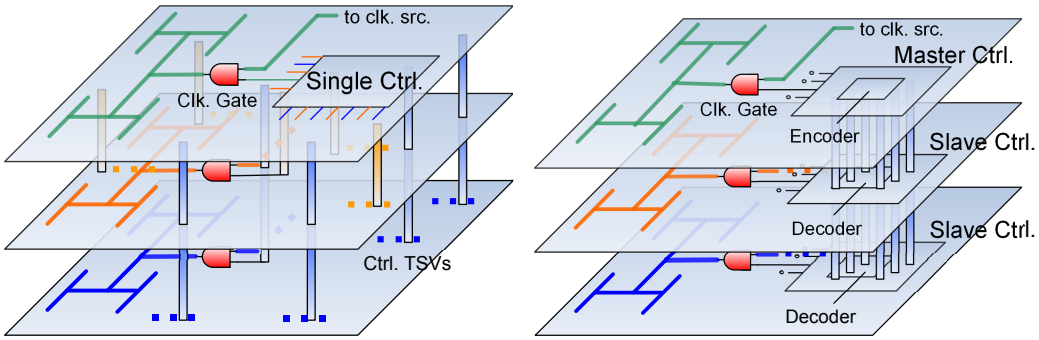
0. S = $\varnothing$; U = $\{G_1, G_2, ..., G_{2n-1}\}$;
1. m = estimation of available *control* TSV numbers;
2. Construct a zero-skew ungated clock tree using DME-3D algorithm [31];
3. **Repeat**
4. Calculate the power saving when inserting each one of the gate in set U;
5. Pick the gate insertion candidate G_i whose has the largest power saving;
6. Add G_i to S;
7. Remove G_i from U;
8. if *control* TSV is need when insert G_i, update m;
9. **Until** $m = 0$ or $U = \varnothing$.

Algorithm 2: Setting up initial state for SA

Initial conditions can speedup the SA convergence and slightly improve the power savings.

3.5 Exploration of multi-layer control scheme

In this section, we explore the possibility of implementing multi-layer control units.



(a) (b)

Figure 3.6: (a) Single control unit distribute "enable" signals through a "star" *control* TSV network. (b) Multi-layer control units employ internal buses to achieve inter-layer synchronization.

As shown in Figure 3.6(a), the single control unit analyzes the shutdown condition of each clock tree branch, and issues the "enable" signals to each of the shutdown gates through separate channels. These channels form a "star" network, in which shutdown gates at different layers are reached via *control* TSVs. For convenience, we assume an N-layer 3D-IC in which n_g shutdown gates are distributed among layers, and the i'th layer contains n_i shutdown gates ($i = 1, 2, \ldots, N$). The single control unit is located at the first layer. Therefore, a fully gated clock tree (*e.g.*, Figure 3.1(a)) requires n_2 *control* TSVs to reach the shutdown gates at the second layer, $2n_3$ *control* TSVs for the third layer, and $(N-1)n_N$ *control* TSVs for the N^{th} layer. The total usage of *control* TSVs equals $\sum_{i=2}^{N}(i-1)n_i$.

An implementation of multi-layer control units is shown in Figure.3.6(b). In a multi-layer control scheme, the master control unit encodes all the shutdown gates' indexes and sends the encoded bits to the slave control units at other layers. The slave control units receive and decode the signal, and then distribute the "enable" signal through plannar wiring. Although no *control* TSV is needed, signal TSVs are required between the master control unit to the slave control units, and the encoding and decoding circuitries yield extra overhead. A binary stream can be used to represent indexes of the selected shutdown gates. For example, "111000" represents that shutdown gates with index 1,2 and 3 are "on", and gates with index 4,5 and 6 are "off".

Let's consider the problem of how many TSVs are needed to reliably transmit the binary string from the master control unit to the slave control units without loss. First we calculate the TSV count from the master control unit to the slave

control unit at the second layer. The master control unit needs to encode the shutdown gates' indexes at the second, third, ... and the N'th layer, with a total count of $\sum_{i=2}^{N} n_i$. Therefore there are $2^{\sum_{i=2}^{N} n_i}$ possible combinations. Assuming the appearance of each binary stream is a binary random variable, with the probability of p_i, according to the Shannon entropy theory, the entropy of the interface between the master control unit and the first slave control unit, H_1 is defined as follows [43]:

$$H_1 = -\sum_{i=1}^{2^{\sum_{i=2}^{N} n_i}} p_i log_2 p_i. \tag{3.9}$$

The entropy H_1 represents the minimum channel width to reliably transmit the binary stream without loss [43]. The upper bound of H_1 is $log(2^{\sum_{i=2}^{N} n_i}) = \sum_{i=2}^{N} n_i$ when all the p_i are equal. Similarly, the entropy of the interface between the first slave control unit and the second slave control unit, $H_2 \leq \sum_{i=2}^{N} n_i$, and $H_{N-1} \leq \sum_{i=N-1}^{N} n_i$. Therefore, the upper bound of TSV usage in a multi-layer control scheme is:

$$H = \sum H_i \leq = \sum_{j=1}^{N-1} \sum_{i=j+1}^{N} n_i = \sum_{i=2}^{N} (i-1) n_i. \tag{3.10}$$

The entropy from the simulation results are reported as follows. In order to compare the single control unit with the multi-layer control units, we calculate the system's entropy value based on a completed gated design (*e.g.*, Figure 3.1(a)). We execute a queue of instructions, and for each instruction, we analyze the indexes of

the shutdown gates that need to be turned on. We count the appearance of each bit stream, and divide it by the total number of instructions to obtain its probability (p_i).

For a 2-layer 3D IC in which shutdown gates are evenly distributed among layers, we show the total number of TSVs needed for a single-layer control and multi-layer control scheme respectively with respect to the total number of shutdown gates, n_g, in Figure 3.7.

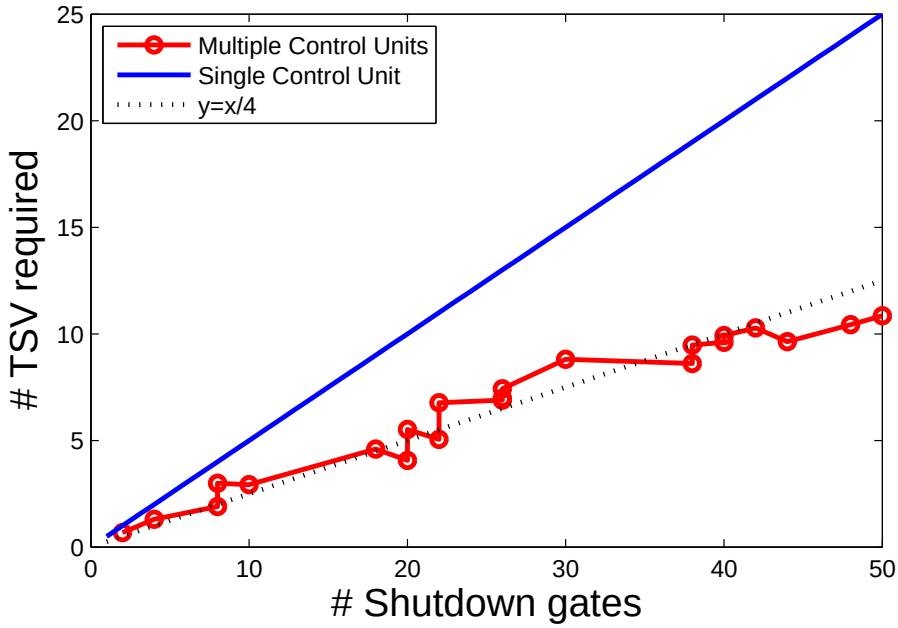


Figure 3.7: Binary stream's entropy increases linearly with the number of shutdown gates.

Figure 3.7 shows that the TSV required in a 2-layer control scheme in practice scales approximately with $n_g/4$. This is because the appearance of each binary stream is not of equal probability. For example, when a parent node in a clock tree is clock gated, it will be unnecessary to clock gate any of its descendant nodes.

In summary, a multi-layer control scheme uses less TSVs for delivering the control signal, at the cost of the overhead of encoding and decoding circuitries.

3.6 Experimental result

Our 3D clock tree synthesis flow is implemented using Matlab and run on a Windows machine by an Intel Core i5 3.1GHz CPU and 12GB RAM. We evaluate the performance of our clock gating algorithm based on the benchmark circuits from ISPD clock network synthesis contest [44]. These benchmarks contains up to 273 clock sinks. Since the benchmark circuits are originally designed for 2D chip (with an area of A), we randomly partition all the clock sinks into N layers, with an area of $\frac{A}{N}$ on each layer. In this chapter, we focus on 2-layer 3D ICs ($N = 2$), however, our algorithms can be applied to any layer number as well. We randomly generate several whitespaces for TSVs, where TSVs are located. For each of the six ISPD benchmarks, we fix the total chip area and impose three sets of layout whitespace constraint (for TSV placement), small $= 0.1\%$, medium $= 0.15\%$, and large $= 0.2\%$. As more placement whitespace is allocated for TSVs, more *clock* and *control* TSVs and be placed. When clock gating is applied, more shutdown gates are available to a designer, which help reduce the overall power dissipation.

The diameter of the TSV we use for simulation is 5 μm, and the minimum keep-out distance between centers of neighboring TSVs is 12 μm [40]. The technology parameters are based on the 45nm predictive technology [40], the 2009 ISPD clock network synthesis contest [44], and previous work on clock tree synthesis [2, 31]: $r_w = 0.1\Omega/\mu m$, $c_w = 0.2fF/\mu m$, $r_v = 0.035\Omega$, and $c_v = 50fF$.

In addition, we follow the work of J.Oh *et al.* to generate random activity patterns for clock sinks which have the following characteristics [22]. 10% of the instructions are made to appear 50% of the time in the instruction stream. The remaining 90% of the instructions make up the remaining 50% of the instruction stream. The average number of used modules per instruction is 20% of the total number of clock sinks.

3.6.1 Abstract tree generation

In this section we compare the quality of our K-means clustering based abstract tree generation algorithm (Algorithm 1) to the 3D-MMM method on a 2-layer 3D IC. We generate abstract clock trees using both methods and then embed the resulting clock trees using the DME algorithm assuming no clock gating. The DME algorithm ensures zero clock skew. We implement slew-aware buffer insertion (sDMBE) [2] simultaneously with DME. Buffers are inserted into clock wiring wherever its loading capacitance exceeds a predefined threshold. Similar buffer insertion scheme was

applied in [2]. For each benchmark, we sweep the *clock* TSV count by controlling the TSV bound parameter (B) in Algorithm 1. The relation between *clock* TSV count and clock power for a representative benchmark f22 is shown in Figure 3.8.

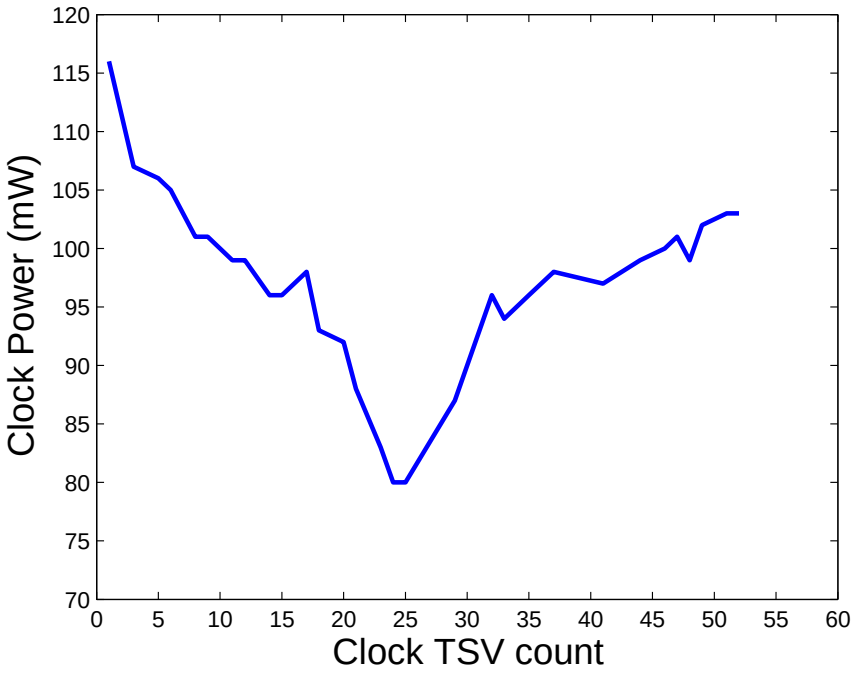


Figure 3.8: Clock power trends for benchmark f22 when sweeping *clock* TSV count. Too many *clock* TSVs increases clock power due to TSV's capacitance.

As shown in Figure 3.8, reducing the *clock* TSV count to a certain point saves clock power. However, increasing the *clock* TSV count beyond the minimum point increases the clock power, where the clock power is more and more influenced by *clock* TSV's capacitance. For each benchmark circuit, we sweep the *clock* TSV count and pick the design which results in the lowest clock power. The results for ungated and buffered clock tree are shown in Table 3.2.

Table 3.2: Comparison of 3D **ungated** and **buffered** zero-skew clock tree's TSV count (#TSV), wire length (WL, in μm), power consumption (Pw, in W), number of buffers used, delay at the clock root (ps), and simulation runtime (RT, in s) between using 3D-MMM and our K-means clustering algorithms

Ckt.	3D-MMM						K-means Clustering						Red. (%) in	
	TSV	WL	Pw	Bufs	Delay	RT	TSV	WL	Pw	#Bufs	Delay	RT	WL	Pw
f11	36	159222	0.165	90	0.179	0.17	20	140607	0.146	81	0.198	2.04	11.7	11.5
f12	34	154039	0.159	81	0.199	0.17	31	135449	0.143	78	0.154	1.05	12.1	10.1
f21	40	152703	0.158	93	0.189	0.16	20	149748	0.151	84	0.229	1.01	1.9	4.4
f22	28	97999	0.107	59	0.152	0.19	24	66835	0.080	46	0.145	0.86	31.8	25.2
f31	57	428124	0.426	241	0.426	0.31	40	373826	0.374	208	0.309	1.07	12.7	12.2
f32	54	298797	0.300	165	0.286	0.22	37	260383	0.263	142	0.263	6.11	12.9	12.3
Rat.	1.00	1.00	1.00	1.00	1.00	1.00	0.69	0.87	0.88	0.88	1.45	0.91	13.8	12.6

As shown in Table 3.2, our K-means clustering based algorithm produces 3D abstract clock trees with shorter total wire length and less power consumption, in every ISPD benchmark. For an ungated clock tree, our clustering algorithm captures the physical proximity of clock sinks, thus improves the total wire length and saves the clock tree's dynamic power. On average, for buffered clock trees, compared to the 3D-MMM algorithm, our wire length reduction reaches 13.8%, while the power reduction is 12.6%.

The wire length and power reduction comes from two reasons. Firstly, 3D-MMM groups the clock sinks based on the median value of their x (or y) coordinates, while the K-means clustering clusters the clock sinks that are close to each other. The following figure reveals the wire length advantage of the K-means clustering over 3D-MMM.

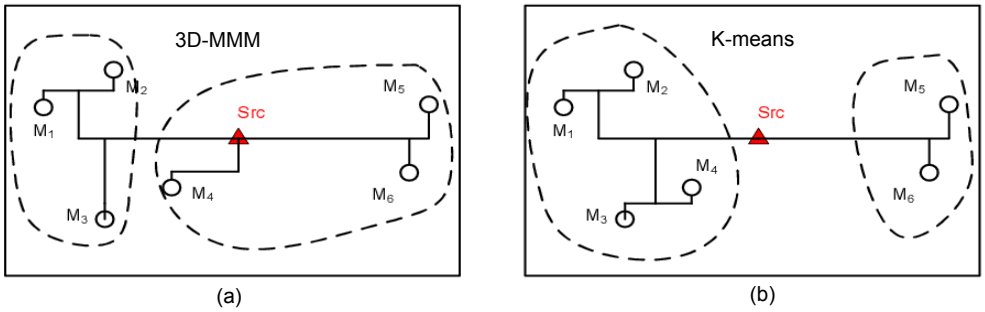


Figure 3.9: Wire length comparison between 3D-MMM and K-means.

In Figure 3.9, there are six clock sinks (M_1 to M_6). M_4 is physically close to M_1, M_2, and M_3, thus M_1, M_2, M_3 and M_4 should intuitively be grouped together. However, when we apply 3D-MMM, as Figure 3.9.(a) shows, M_1, M_2, and M_3 form one sub-tree and M_4, M_5 and M_6 form another. Grouping M_4, M_5 and M_6 inevitably increases the total wire length. Figure 3.9.(b) applies K-means clustering, and the physically closed sinks, M_1 to M_4 form one sub-tree while M_5, M_6 form another.

The second reason is that after abstract tree generation, it is necessary to ensure zero clock skew during embedding (*i.e.* DME algorithm). When two children nodes have different capacitive loads, the DME algorithm introduces extra wiring (wire snaking) at the node with smaller capacitance. Therefore, a high-quality abstract clock tree should be able to balance the capacitive loads between two children nodes. Unfortunately the 3D-MMM method only considers the clock sinks' physical location while ignoring the capacitive loads of the clock sinks. This leads to significant wiring overhead in order to achieve zero clock skew. On the contrary, our K-means clustering method inherently considers the clock sink's capacitive load, thereby significantly reduces extra wiring overhead.

The runtime of K-means clustering is $\mathcal{O}(nkdI)$, where n is the total number of clock sinks, k is number of clusters, ($k = 2$ in our case), d is the dimension ($d = 3$ in our case), and I is total number of iterations. K-means clustering converges when the centroid assignments no longer change, and it has been proved that if k and d are fixed values, the upper bound of time complexity is less or equal to $\mathcal{O}(n^{dk+1}logn)$ [45]. Therefore, the upper bound for K-means based abstract clock tree generation algorithm is less or equal to $\mathcal{O}(n^{dk+1}lognlogn)$, since we need $\mathcal{O}(logn)$ partitions. However, in our simulation, the number of iterations is often small until convergence (roughly $\mathcal{O}(n)$), and the total time complexity is roughly $\mathcal{O}(nlogn)$. The runtime is similar to 3D-MMM, which is also $\mathcal{O}(nlogn)$.

3.6.2 Clock tree embedding

The second phase of 3D clock tree synthesis, the clock tree embedding step, determines the locations of intermediate clock tree nodes and the *clock* TSVs. For a gated 3D clock tree, the location of shutdown gates and the locations of the *control* TSVs are also calculated in this step.

In this section, we construct the low-power gated 3D clock tree based on the abstract tree generated by the 3D-MMM and the K-means clustering algorithm. We sweep the *clock* TSV count to generate a set of abstract clock tree designs. For each of the abstract clock trees, we use the DME-3D algorithm [31] with slew-aware

buffer insertion (sDMBE) [2] to generate an ungated baseline design. The baseline design with the lowest clock power is selected and recorded, which is referred to as "Before Optimization" in the table.

Table 3.3: Summary of the *clock* TSV's count (Clk. TSV), total wire length (WL, in μm), total power consumption (Pw, in W), and number of buffers used (Bufs) before clock gating, and the *control* TSV's count (Ctrl. TSV), total power consumption (Pw, in W), and simulation run time (RT, in s) after solving P1

Ckt.	TSV White-space	3D-MMM							K-means						
		Before Opt.				After Opt.			Before Opt.				After Opt.		
		Clk. TSV	WL	Pw	Bufs	Ctrl. TSV	Pw	RT	Clk. TSV	WL	Pw	Bufs	Ctrl. TSV	Pw	RT
f11	small	15	173283	0.177	124	16	0.128	15	16	152829	0.157	100	14	0.095	76
	medium	35	168783	0.176	113	6	0.124	17	16	150149	0.155	99	24	0.092	84
	large	35	168783	0.176	113	15	0.119	21	17	148073	0.153	99	33	0.089	103
f12	small	25	150647	0.158	105	3	0.112	15	9	137211	0.142	94	19	0.084	73
	medium	23	145872	0.153	105	14	0.107	12	5	138055	0.142	95	32	0.082	59
	large	35	128872	0.141	99	12	0.104	15	6	138579	0.143	96	41	0.079	76
f21	small	8	169796	0.169	120	25	0.117	13	9	161844	0.161	108	24	0.093	64
	medium	8	169796	0.169	120	36	0.114	13	24	147056	0.151	99	21	0.089	66
	large	8	169796	0.169	120	47	0.110	14	23	146985	0.151	99	32	0.086	72
f22	small	11	121951	0.126	83	10	0.085	11	1	106437	0.111	72	20	0.068	56
	medium	20	109668	0.118	79	8	0.083	8	24	83756	0.095	62	5	0.061	38
	large	26	89061	0.101	71	8	0.074	11	24	83756	0.095	62	10	0.060	54
f31	small	32	445257	0.442	296	16	0.383	13	21	361524	0.367	254	27	0.239	65
	medium	36	435819	0.436	297	26	0.358	15	21	361524	0.367	254	41	0.232	73
	large	40	422831	0.425	287	39	0.347	10	25	363489	0.370	254	53	0.228	48
f32	small	21	311488	0.309	209	25	0.235	30	23	259059	0.264	184	24	0.166	116
	medium	48	298551	0.304	205	14	0.228	46	12	266694	0.268	185	51	0.158	142
	large	21	311488	0.309	209	56	0.221	34	9	273795	0.274	192	68	0.154	151
Avg		16	221763	0.225	134	30	0.169	17	16	193379	0.198	134	30	0.120	79
Ratio		-	-	1.00	-	-	0.75	1.00	-	-	0.88	-	-	0.53	4.55

Similarly, we apply the SA and the TSV placer introduced in Section 3.4.3 to solve Problem 1 and Problem 2, for each of the abstract clock tree designs. After clock gating, the optimal (in terms of clock power) design is referred to as "After Optimization" in the table. Three sets of TSV whitespace area are investigated. Larger TSV whitespace area can accommodate more *clock* and *control* TSVs. After solving Problem 1, the results for the baseline design and the optimized design are summarized in Table 3.3. Similarly, the results for solving Problem 2 are summarized in Table 3.4.

Table 3.3 indicates that on average, solving Problem 1 achieves 25% and 40% power reduction for clock trees generated by 3D-MMM and K-means respectively. When a larger TSV placement whitespace is given, more *control* TSVs can be legally placed, which leads to more shutdown gates and more significant power saving.

In Problem 1, the placement of *control* TSVs is hindered by fixed locations of *clock* TSVs. This is improved in Problem 2 where we place the *clock* TSVs simultaneously with *control* TSVs, which achieves a better whitespace utilization. Comparing Table 3.3 and Table 3.4, we see the average power for a clock gated tree reduces from 0.169W to 0.167W, and from 0.120W to 0.118W for 3D-MMM and K-means respectively.

Figure 3.10 illustrates the interaction between *clock* TSVs and the *control* TSVs. For a fixed TSV whitespace area constraint, increasing the number of *control* TSV count shrinks the whitespace for the *clock* TSVs. There exists an optimal allocation for *clock* and *control* TSVs such that the clock power is minimized. An

Table 3.4: Summary of the *clock* TSV's count (Clk. TSV), total wire length (WL, in μm), total power consumption (Pw, in W), and number of buffers used (Bufs) before clock gating, and the *control* TSV's count (Ctrl. TSV), total power consumption (Pw, in W), and simulation run time (RT, in s) after solving P2

Ckt.	TSV White-space	3D-MMM							K-means						
		Before Opt.				After Opt.			Before Opt.				After Opt.		
		Clk. TSV	WL	Pw	Bufs	Ctrl. TSV	Pw	RT	Clk. TSV	WL	Pw	Bufs	Ctrl. TSV	Pw	RT
f11	small	15	173153	0.176	122	19	0.127	16	12	159835	0.163	106	22	0.093	86
	medium	15	173153	0.176	122	29	0.122	23	17	148073	0.153	99	27	0.090	86
	large	26	166652	0.172	112	29	0.119	22	16	150149	0.155	99	40	0.086	95
f12	small	23	145872	0.153	105	5	0.108	14	5	138055	0.142	95	25	0.083	83
	medium	35	128872	0.141	99	7	0.105	19	5	138055	0.142	95	35	0.082	92
	large	35	128872	0.141	99	17	0.102	20	5	138055	0.142	95	46	0.077	96
f21	small	24	165690	0.169	116	12	0.119	20	11	162057	0.162	106	25	0.092	65
	medium	8	169796	0.169	120	42	0.111	48	21	148356	0.152	98	29	0.086	66
	large	8	169796	0.169	120	54	0.110	52	23	146985	0.151	99	38	0.085	74
f22	small	20	109668	0.118	79	3	0.083	12	5	104235	0.109	70	18	0.066	72
	medium	26	89061	0.101	71	4	0.076	9	25	83923	0.096	63	6	0.061	68
	large	27	88279	0.101	70	11	0.075	14	24	83756	0.095	62	15	0.058	61
f31	small	32	445257	0.442	296	20	0.370	11	23	361204	0.368	254	28	0.241	132
	medium	30	448160	0.444	294	39	0.356	19	21	361524	0.367	254	47	0.232	155
	large	32	445257	0.442	296	55	0.340	18	23	361204	0.368	254	64	0.222	193
f32	small	21	319016	0.316	211	30	0.233	32	23	259059	0.264	184	28	0.164	149
	medium	23	308803	0.308	211	45	0.226	52	12	266694	0.268	185	56	0.154	232
	large	46	300008	0.305	205	40	0.219	35	23	259059	0.264	184	63	0.151	168
Avg		25	220854	0.225	153	26	0.167	24	16	192793	0.198	133	34	0.118	110
Ratio		-	-	1.00	-	-	0.74	1.00	-	-	0.88	-	-	0.53	4.53

optimal ungated 3D clock tree design (*i.e.* minimum power) does not yield an optimal gated 3D clock tree. The impact of *control* TSVs needs to be considered during 3D clock tree synthesis stage in order to obtain the optimal power saving.

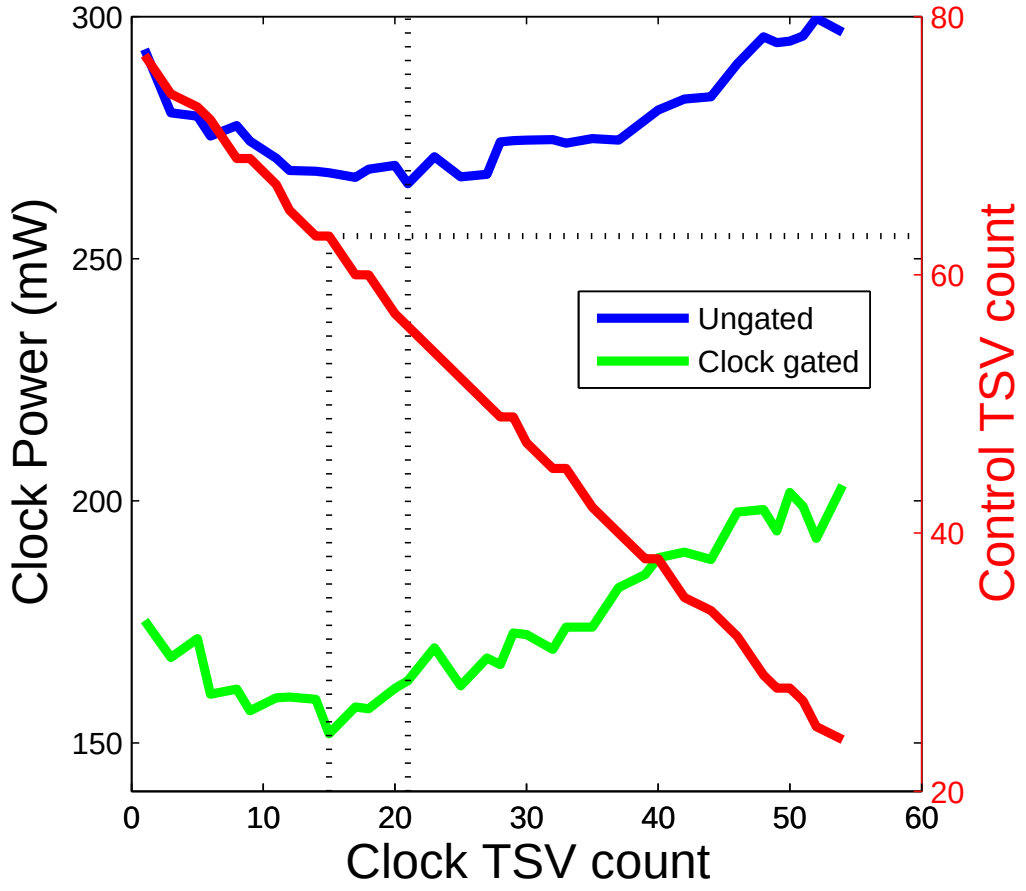


Figure 3.10: The clock power trend for benchmark f32 when a fixed TSV whitespace area is applied. Increasing the *clock* TSV count reduces the *control* TSV count. The 3D clock tree that is optimal for clock gating is different from the ungated clock tree of optimal power. The impact of *control* TSVs needs to be considered during 3D clock tree synthesis stage in order to obtain the optimal power saving.

At last, we present the comparison between placement unaware 3D clock tree design and our placement aware design. We use a completely gated clock tree as an example of TSV whitespace unaware design, and show the power consumption, TSV usage, and placement violation in Table 3.5. All clock trees in Table 3.5 are generated using our K-means based algorithm, with buffer insertion.

Table 3.5: Comparison of **ungated**, **gated**, and **completely gated** 3D clock trees.

Ckt.	TSV Whitespace	Ungated Tree		Partially Gated Tree		Completely Gated		
		Clk. TSV	Pw	Ctrl. TSV	Pw	Ctrl. TSV	Pw	Violation
f11	small	12	0.163	22	0.093	241	0.082	6.4x
	medium	17	0.153	27	0.090	241	0.079	4.9x
	large	16	0.155	40	0.086	241	0.079	3.6x
f12	small	5	0.142	25	0.083	233	0.072	6.9x
	medium	5	0.142	35	0.082	233	0.072	5.0x
	large	5	0.142	46	0.077	233	0.072	3.7x
f21	small	11	0.162	25	0.092	233	0.082	5.8x
	medium	21	0.152	29	0.086	233	0.078	4.1x
	large	23	0.151	38	0.085	233	0.078	3.2x
f22	small	5	0.109	18	0.066	181	0.057	7.1x
	medium	25	0.096	6	0.061	181	0.049	5.6x
	large	24	0.095	15	0.058	181	0.049	4.3x
f31	small	23	0.368	28	0.241	545	0.189	10.1x
	medium	21	0.367	47	0.232	545	0.188	7.3x
	large	23	0.368	64	0.222	545	0.189	5.5x
f32	small	23	0.264	28	0.164	379	0.132	6.9x
	medium	12	0.268	56	0.154	379	0.135	4.8x
	large	23	0.264	63	0.151	379	0.132	3.7x

Table 3.5 shows that a completely gated clock tree achieves better power comparing to a partially gated clock tree, however, its TSV placement violation varies from 3.6**x** to 10.1**x**, depending on the TSV whitespace availability. Therefore, in designs that TSV placement whitespace constraints are applied, completely gated clock trees are not practical.

3.7 Summary

In this chapter, we propose a design flow for 3D low-power clock tree synthesis, using the clock gating technique. Our design flow consists of two phases: (1) Firstly we propose a K-means clustering based algorithm to construct a 3D abstract tree topology that is suitable for shutdown gate insertion. (2) Secondly we use simulated annealing and a force-directed TSV placer to select the shutdown gates, while ensuring all the *clock* TSVs and *control* TSVs can be legally placed in the placement whitespace. We co-place *clock* TSVs and *control* TSVs to enhance the usage of TSV placement area, so that more *control* TSVs can be placed and better power savings are achieved. Our experimental results show that our algorithms can effectively generate a low-power 3D abstract tree and also decide at which tree edge to put shutdown gates and the TSV locations, which saves clock power while ensuring zero clock skew.

Chapter 4: Modeling and EM-aware Layout Optimization for Tapered TSVs

Through-Silicon-Via offers vertical connections for 3D ICs. Due to its large dimensions and non-ideal etching process, TSV's layout needs to be carefully optimized in order to balance peak current density and delay for digital circuit. This chapter investigates the TSV's tapering effect, (which is an inevitable byproduct of Deep Reactive Ion Etching based manufacturing) and its impact on the TSV's electrical properties. We show that the current crowding effect is more severe in realistic tapered TSVs than ideal cylindrical TSVs. We propose a non-uniform current density model for tapered TSVs which achieves considerable accuracy and speedup in estimating the current density distribution, when compared to existing models developed for cylindrical TSVs. We apply our model to perform a detailed study on (1)impact of TSV's tapering on peak current density, and (2)wire sizing problem in order to minimize TSV-involved path delay under second-order delay model while keeping the peak current density within tolerable levels. A new dynamic programming based heuristic is proposed to find the optimal wire configuration which reduces both peak current density and delay thereby improving the reliability and performance.

This chapter is organized as follows: In Section 4.1, we introduce the mainstream TSV's fabrication method and explain the source of the tapering effect. In Section 4.2 we set up a simulation to investigate the DC current distribution within both the cylindrical and the tapered TSV. In Section 4.3, an effective TSV non-uniform current density model is implemented and validated against the FEM based simulation result in Section 4.2. The non-uniform model along with a second-order delay model are applied to a layout optimization problem in Section 4.4. Section 4.5 presents the experimental result to show how the tapering effect and layout configuration affect peak current density and delay, and give some final thoughts regarding the TSV tapering effect.

4.1 Tapered TSVs: An Inevitable Byproduct of Manufacturing

4.1.1 TSV fabrication

TSV's fabrication includes DRIE, via filling by deposition of diffusion barrier and adhesion layers, metallization, wafer thinning and alignment, and bonding [46,47]. A typical via-first (via formation before CMOS process) can be summarized as follows. First of all conventional contact lithography is used to define the etching area. The wafer is subsequently etched (i.e. DRIE), and then thermal oxidation is performed to form the passivation layer. Once the passivation is grown, the seed layer is deposited using physical vapor deposition (PVD) and metal (Copper, Tungsten, etc.) is electroplated to fill the TSV. This is followed by chemical mechanical polishing (CMP), and then Cu is sputtered and patterned on both sides of

the wafer as contacts for bonding neighboring layers. Finally bonding is performed under high temperature and pressure. There are other fabrication methods, but the complete literature survey of all different approaches for TSV fabrication has been omitted for brevity.

4.1.2 Source of tapering effect

Much of the conventional researches on modeling and optimization of 3D ICs assume the TSVs to be perfectly cylindrical in shape. However, in reality, the fabricated TSVs end up with a conical tapered structure. This tapered structure comes from DRIE, the most common way to etch vias in silicon.

TSV's fabrication includes several key steps such as: DRIE, via filling, metallization, wafer thinning, alignment, and bonding [46, 47]. A typical via-first (via formation before CMOS process) can be summarized as follows. Firstly the etching area is defined by conventional contact lithography. The wafer is subsequently etched (i.e. DRIE), and then thermal oxidation is performed to form the passivation layer. Once the passivation is grown, the seed layer is deposited using Physical Vapor Deposition (PVD) and metal (Copper, Tungsten, etc.) is electroplated to fill the TSV. After Chemical Mechanical Polishing (CMP), Cu is then sputtered and patterned on both sides of the wafer as contacts and bonding is performed under high temperature and pressure.

The DRIE alternates repeatedly between etch mode and passivation mode to create vias with high aspect ratios, as shown in Figure 4.1 [48–50]. The DRIE manufacturing process starts with thermal oxidation, photo resist deposition, patterning, and the first isotropic etching (Figure 4.1(a)). This results in a trench in the silicon substrate where the TSV is to be located. Then a passivation layer is formed on the surface of the TSV trench. This is followed by removing the passivation layer at the bottom to reveal the TSV trench. The bottom of this trench is then etched further towards the silicon substrate. The passivation/etching process is iterated for several hundreds of times till the desired TSV height is reached, as illustrated in Figure 4.1(f). During each etching mode, the sidewall is protected by the passivation layer so that the regions which has already been etched won't be affected by the following etching process. However, each passivation mode reduces the TSV diameter, therefore, as the TSV trench goes deeper, the effective diameter of the trench becomes less. This is the primary reason that when the aspect ratio (height : radius) is high, a tapered structure is what DRIE process usually ends up with. Another obvious characteristic of the repeated etching-passivation process is the sidewall is usually rough and scalloped. The roughness of the sidewall will affect the copper filling process, but in this chapter, for simplicity, we assume the copper is perfectly sputtered and there's no void in the TSV cone.

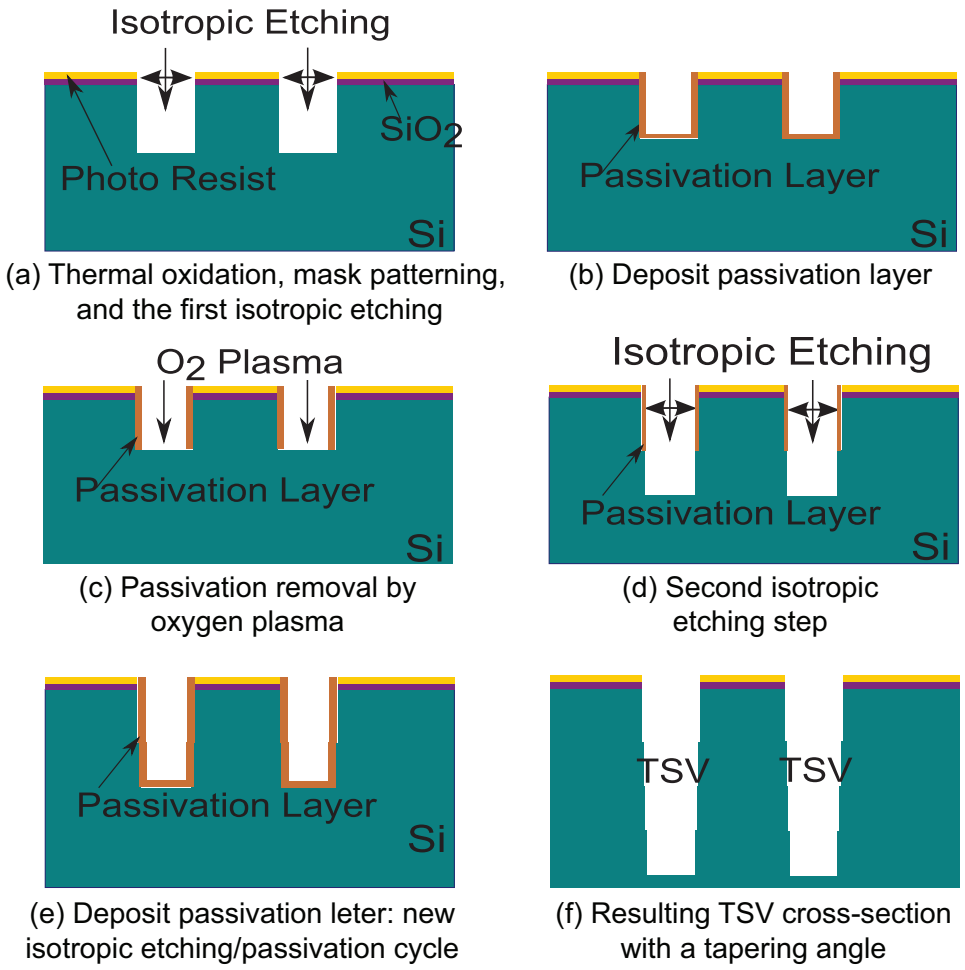


Figure 4.1: Illustration of the DRIE process flow, which is consist of cycles of isotropic etching(a,d), passivation(b,e), and passivation removal(c).

4.1.3 TSV tapering in real fabrication

Such an iterative manufacturing process inherently results in a tapered TSV structure, which has been reported in many experimental works [51–54]. The SEM pictures in [53] reveals a tapered TSV with a top diameter of $10.5\mu m$, a height of $51.4\mu m$, and a tapering angle of $89°$. [54] shows a tapered TSV with top diameter $\approx 5.8\mu m$, height$\approx 22.2\mu m$, bottom diameter $\approx 3.2\mu m$, tapering angle $\approx 87°$.

Some other works have shown that TSV's degree of taper is more pronounced in thinner TSVs with smaller diameters and higher aspect ratios [55]. The authors of [46, 56] indicate that the inherent tapering of TSVs during manufacturing helps improve the manufacturing yields of TSVs since it helps in conformal deposition of isolation dielectric, copper diffusion barrier metalization and copper seed metalization over the sidewalls of the deep silicon vias. A slight taper also enables a void-free copper via electroplating [56]. According to the simulation results from [57], tapered-shape TSV has the smallest reflection noise and signal loss, compared to some other TSV shapes, such as rectangular, circular, and annular shapes.

In conclusion, the tapering effect causes TSVs to resemble a cone with the bottom region being thinner than the top region, and TSV tapering is a natural, unavoidable consequence of TSV fabrication, and in some circumstances the TSV tapering helps improve the manufacturability and yields.

4.2 DC current density FEM simulation for tapered TSV

Excessively high DC current is one of the primary causes for many reliability problems such as overheating and EM. In order to analyze and quantify the DC current density distribution in tapered TSV and cylindrical TSV (as comparison), we use ANSYS WORKBENCH [58] as a simulation tool. The structure of the tapered TSV we used as a test case is shown in Figure 4.2(a). This test case consists of three components: (1) A tapered TSV cone(4 μm in top diameter, 0.856 μm in bottom, 87° of tapering angle and 30 μm in height), (2) two landing pads (

length×width× height = $5\mu m \times 5\mu m \times 0.2\mu m$), and (3) two signal wires (length × width × height =$0.2\mu m \times 0.2\mu m \times 10\mu m$). One current source is inserted at the top left corner, feeding 0.4mA current, and one current sink is inserted at the bottom right corner, absorbing the 0.4mA current.

Figure 4.2(a) shows the 3D view of current density distribution in a tapered TSV's structure. Significantly high current density value is observed approximately $3\mu m$ into the TSV from both the top and bottom interfaces in Z direction. At the bottom pad, as illustrated in Figure 4.2(b), the current not only crowds at the transition regions between the pad and the wire, but also at the interface between the TSV and the bottom pad. The current density distribution at the top pad only peaks at the transition region.

A similar cylindrical TSV case is simulated for comparison. As shown in Figure 4.4(a), the tapering angle is set to be 90°. The top and bottom diameters are both $4\mu m$, and height remains to be 30 μm. The size of the landing pads and signal wires is the same as in the tapered case, ($5\mu m \times 5\mu m \times 0.2\mu m$, and $0.2\mu m \times 0.2\mu m \times 10\mu m$, respectively). Current sources are again inserted at the top left corner and bottom right corner, with 0.4mA current.

In the cylindrical case, the magnitude of current density is again plotted in 3D view (Figure 4.4(a)), within the bottom pad (Figure 4.4(b)) and within the top pad (Figure 4.4(c)). Both the top pad and the bottom pad have only one current density peak, which is at the transition region between the pad and the wire. We

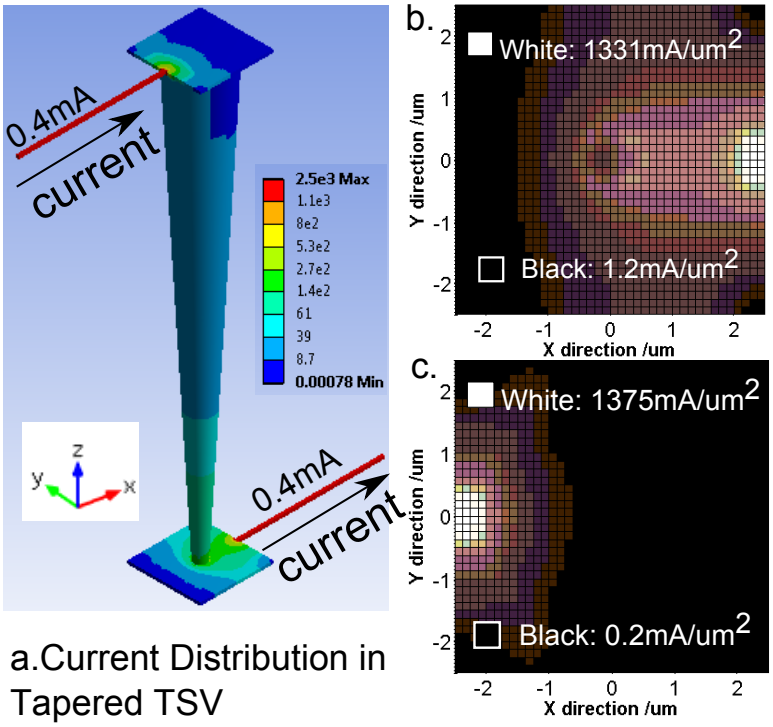


Figure 4.2: (a) Current density ($mA/\mu m^2$) in a *tapered* TSV and signal wires. (b)The current density distribution at the bottom pad. (c) The current density distribution at the top pad.

notice that current density distribution at the top pad is very similar to the top pad in a tapered TSV's structure, while the bottom pad doesn't. The bottom pad in the tapered case has two current density peaks, leading to higher non-uniformity.

A histogram(Figure 4.3) for current density distribution clearly uncovers the significant difference at bottom pad between both cases. Even though the peak current is similar, the current density variances for cylindrical and tapered TSV are 7.0×10^3 and $8.3 \times 10^3 mA/\mu m^2$, respectively. This 18.6% variation difference in current density is an important reason behind different EM trends in [49].

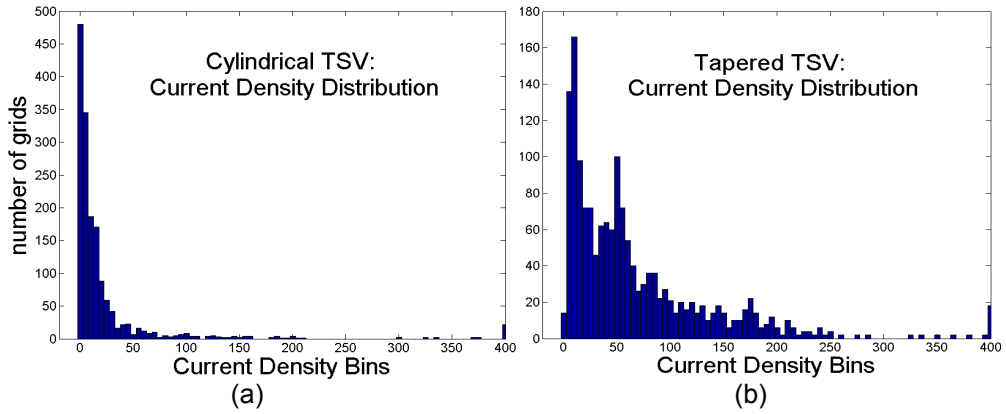


Figure 4.3: Current density distributions at bottom pad for (a) cylindrical TSV and (b) tapered TSV.

4.3 Adaptive Meshing Technique for Tapered TSV

Our previous discussion on tapered TSV's DC current density distribution is based on Finite-Element-Method (FEM) simulation [49]. However, FEM method requires high simulation time thus is not applicable for optimization problems in this chapter. Some recent work has developed distributed resistive models for estimating the current distribution within cylindrical TSVs [3]. The work in [3] essentially proposes a uniform meshing method. It starts with a basic estimate of the current distribution in the cylindrical TSV using ANSYS [58]. ANSYS is a FEM-based simulation tool which is capable of simulating the current distribution of arbitrary physical structure. Furthermore it divides the TSV into horizontal planes of varying thicknesses. Each plane is then represented using a uniform 2D mesh. All the planes and the associated meshes are combined together into a distributed resistive model for estimating the current density profile of the TSV.

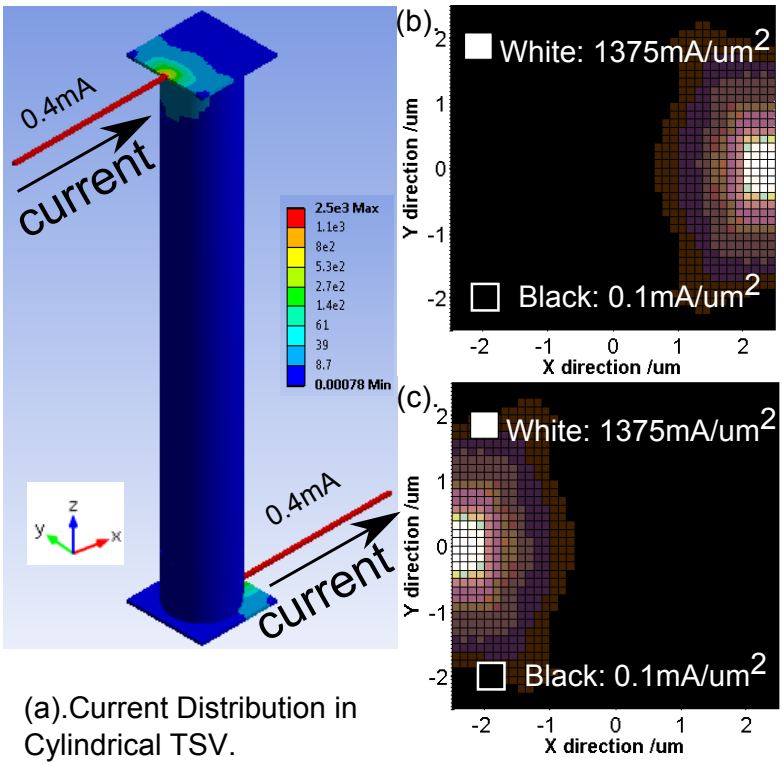


Figure 4.4: (a) Current density ($mA/\mu m^2$) in a *cylindrical* TSV and signal wires. (b)The current density distribution at the bottom pad. (c) The current density distribution at the top pad.

In order to capture the high degree of current variance in tapered TSV the horizontal planes would require a very fine grained uniform mesh. We find that doing so will result in a significant increase in model complexity thereby making it unscalable. In this work, we develop a non-uniform meshing approach for capturing the high degree of current variation in tapered TSVs. The model adaptively allocates finer meshes in higher current density areas, which helps in accurately capturing the tapered TSV current density without excessive modeling complexity.

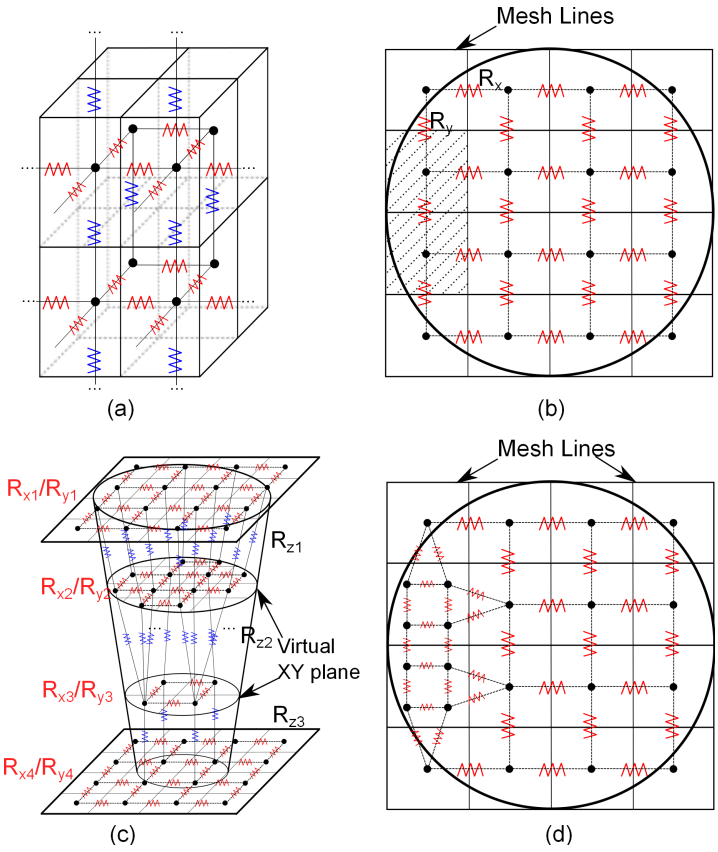


Figure 4.5: TSV modeling of a 3-D resistive network. (a) Basic rectangular box after 3-D meshing; (b) First iteration of horizontal mesh at top pad; (c) Horizontal and vertical interconnect networks; (d) Second iteration of horizontal mesh at top pad.

Our modeling approach follows an iterative-constructive approach. The tapered TSV is represented by a distributed resistive mesh(Figure 4.5(a)). For the sake of simplicity, we assume that the TSV is partitioned into P horizontal planes stacked together. We begin by representing all the horizontal planes using a simple uniform mesh. Figure 4.5(b) illustrates the first horizontal mesh at the top pad. This is followed by applying Kirchhoff's circuit laws to estimate the current at each grid on the mesh. Note that the resistance of each mesh is a function of its dimension and TSV material properties(Figure 4.5(c)). The allocation of the resistance

Starting from a simple uniform mesh:
1. Initialize the iteration limit for each TSV;
2. Initialize the variance threshold;
3. **Repeat**:
4. Use Kirchhoff's circuit laws to obtain the current density at each node;
5. Select the mesh tiles with large current density;
6. Decompose those mesh tiles into smaller 2×2 mesh tiles
7. **If** iteration times do not exceed the iteration limit and current density variance within each horizontal layer is higher than the variance threshold, go to step 3;
8. **Else** stop.

Algorithm 3: Algorithms for non-uniform TSV resistive model

values to the interconnection network has been omitted for brevity. Those regions of each horizontal plane which embody higher peak current are selectively refined while other regions with lower currents are left coarse(Figure 4.5(d)). This process is iterated till the current density variance in each horizontal layer is less than a user selected threshold. An additional stopping criterion is based on the maximum number of iterations that the user is willing to tolerate. The modeling approach is summarized in Algorithm 3.

There are two advantages of our modeling approach over a) ANSYS and b) approach in [3].

1. Accuracy-Runtime Trade-off: Our model is capable of allocating finer meshes in high current density areas and coarser meshes in low current density regions. Hence it combines the *best of both worlds* by providing significant accuracy while ensuring simplicity. Our approach is significantly faster compared to

ANSYS without compromising too much on accuracy. We are also better in accuracy over the model in [3] for the same runtime. The runtime vs accuracy trade-off comparison between our model and [3] is highlighted in Figure 4.9.

2. Another advantage over [3] is that we do not need to run ANSYS for generating our model while the previous approach relied on running ANSYS once.

4.4 TSV's Layout Optimization

In this section we describe a layout optimization that uses the developed current distribution model to optimize the maximum current density and delay in TSVs.

4.4.1 Problem 1: Reduce peak current density

As indicated in [49], high current densities in TSVs can cause reliability issues (*e.g.*, EM). Assuming the driving current stays the same, the primary reason behind high *local* current density in TSVs is the non-uniform current distribution profile. As shown in Figure 4.2, the top and bottom pad regions are the most critical regions of high TSV current density as well as high degree of current density variance. Hence we investigate a new approach where a small part of the wire segment connected to the top and bottom pads is made wider. This helps in spreading out the current on the pads and also on the TSVs to reduce the peak current values. In order to find the smallest wire segment width that reduces the maximum current in top and bottom regions of the TSV (middle regions have very little variation in current), we

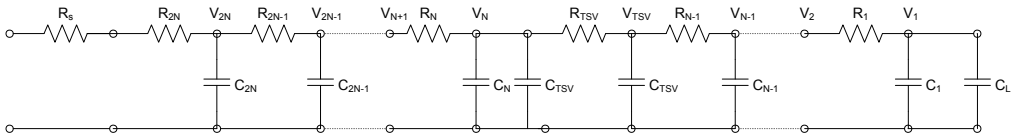


Figure 4.6: RC model of TSV and wires.

start with a small value of top and bottom wire segment values (denoted as w_n and w_{n+1} in Figure 4.7) and iteratively increase it (and generate new resistive model in the meantime) till the peak current density is within tolerable level. Note that AC current is converted into equivalent root-mean-square DC current.

4.4.2 Problem 2: TSV's second-order delay optimization

One aspect that "Problem 1" ignores is that the wire sizing may cause larger timing delays. Hence we want to allocate the wire segment sizes such that the peak current in the TSV is less than certain value and the delay across the TSV (from source to sink) is minimized.

Before formulating the problem, we make the following observations. The peak current density around the top pad (bottom pad) area is a monotonically decreasing function of the width of the wire segment connecting it (this fact will be illustrated in the results section). The formulation in Problem 1 will find the *smallest* values of w_n and w_{n+1} (see Figure 4.7) such that the peak current density is within tolerable levels. Let us suppose these optimal values are w_n^{opt} and w_{n+1}^{opt}. Allocating a width greater than these values would surely reduce the peak current even more. Hence any $w_n \geq w_n^{opt}$, $w_{n+1} \geq w_{n+1}^{opt}$ would be tolerable from a current density perspective.

Now we formulate the TSV delay driven wire sizing problem. Consider the illustration in Figure 4.7. The wire segments connected to the top and bottom pads are divided into n segments each. Wire segments w_n and w_{n+1} are connected to the TSV pads directly. Hence these wire segments need to be $w_n \geq w_n^{opt}$, $w_{n+1} \geq w_{n+1}^{opt}$ respectively. Our objective is to find the allocation of wire sizes to all the 2n wire segments such that the width constraints on w_n and w_{n+1} are satisfied and the overall delay is minimized.

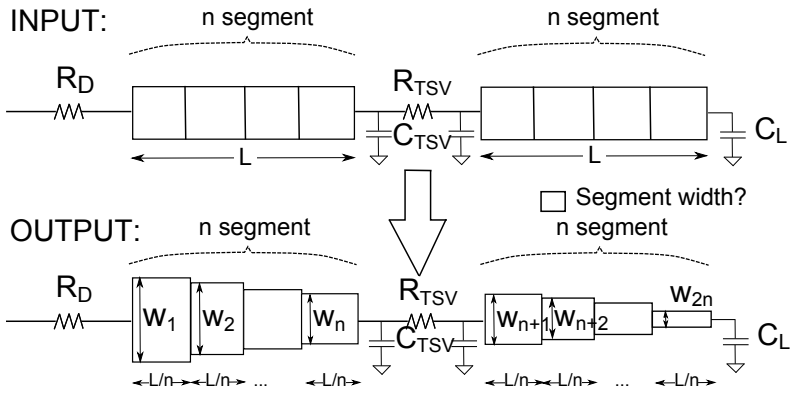


Figure 4.7: Illustration of the TSV-aware wire sizing problem. The input is R_D,C_L, and n.The output is w_1,w_2, ...,w_{2n}.

Let us assume that the TSV and wires can be represented as a lumped RC circuit as illustrated in Figure 4.6. The value of R_{TSV} and C_{TSV} are calculated by Equation 4.1.

$$R_{TSV} = \int_0^L \frac{\rho}{dS} dx = \int_0^L \frac{\rho}{\pi(r - xcot(\theta))^2} dx; \tag{4.1a}$$

$$C_{TSV} = c_o \cdot L. \tag{4.1b}$$

where L is the length of TSV, ρ is resistivity, r is the radius of TSV at the top, θ is TSV's tapering angle, and c_o is capacitance per unit length.

$$b_1^{i+1} = R_i \sum_{j=1}^{i} C_j + b_1^i \tag{4.2a}$$

$$b_2^{i+1} = R_i \sum_{j=1}^{i} C_j b_1^j + b_2^i \tag{4.2b}$$

$$D(i) = K_r \cdot \frac{b_1^i + \sqrt{{b_1^i}^2 - 4b_2^i}}{2} \tag{4.2c}$$

Assuming lumped RC model for each wire segment w_i, we express the delay across the TSV, D_i as in Equation 4.2 (using second-order delay model in [59]), where $K_r = 2.36$ according to [59]. Based on this delay model, our optimization problem can be formulated as follows.

$$min \qquad D_{a->b}(w) \tag{4.3a}$$

$$s.t. \qquad w_n^{opt} \leq w_n \leq M \tag{4.3b}$$

$$w_{n+1}^{opt} \leq w_{n+1} \leq M \tag{4.3c}$$

$$m \leq w_i \leq M, \qquad i = 1, ..., 2n \tag{4.3d}$$

where a and b are two ends of wires, n is number of wire segments, M and m are the largest and smallest sizes possible for the wire segments. The second-order delay model contains a non-convex term, making it difficult to solve using conventional convex optimization techniques such as [60]. We propose Algorithm 4 to solve Problem 2 effectively.

1. **Foreach** V_i (in Figure 4.6) from load to source, **do**:
2. **if** V_i is sink **then**
3. $P \leftarrow (0, 0, C_L, 0)$;
4. **else**
5. **Foreach** $p \in P$
6. **Foreach** wire width **do**
7. $P_{new} = (b_1^{i-1} + R_{i-1}C_{tot}^{i-1}, b_2^{i-1} + R_{i-1}Q^{i-1}, C_{tot}^{i-1} + C_i, Q^{i-1} + C_i b_1^i)$;
8. $P \leftarrow P \bigcup P_{new}$;
9. **end for**
10. **end for**
11. **end for**
12. Prune P;
13. **end for**
14. Calculate second-order delay and pick the optimal wire sizing.

Algorithm 4: Dynamic programming for delay optimization

Algorithm 4 is based on the dynamic programming technique. During the iteration from load to source (from V_1 to V_{2N}), a set of tetrad (b_1, b_2, C_{tot}, Q) is updated and maintained for every possible wire width options, where b_1^i and b_2^i are computed based on Equation (4.2a) and Equation (4.2b), C_{tot}^i is total downstream capacitance at V_i, and $Q^i = \sum_{j=1}^{i} C_j b_1^j$. After every possible wire width option is traversed, we prune the solution space based on the following criteria: Given two wire sizing options p^i and $p^{i\prime}$, we say $p^{i\prime}$ is redundant if $b_1^i \leq b_1^{i\prime}$, $b_2^i \geq b_2^{i\prime}$, $C_{tot}^i \leq C_{tot}^{i\prime}$, $Q^i \leq Q^{i\prime}$, and at least one of the four inequalities is strict. The pruning criteria are different than that in [61], where timing slack instead of b_1 and b_2 is computed and updated. However, some wire width configurations with large delay value (or small slack) at node V_i may have small C_{tot} and Q, thus pruning out these solutions in [61] might lead to suboptimal solutions. After (b_1, b_2, C_{tot}, Q) for node V_{2N} is computed, we calculate the second-order delay according to Equation 4.2c and pick the optimal wire sizing solution.

4.5 Results and Discussions

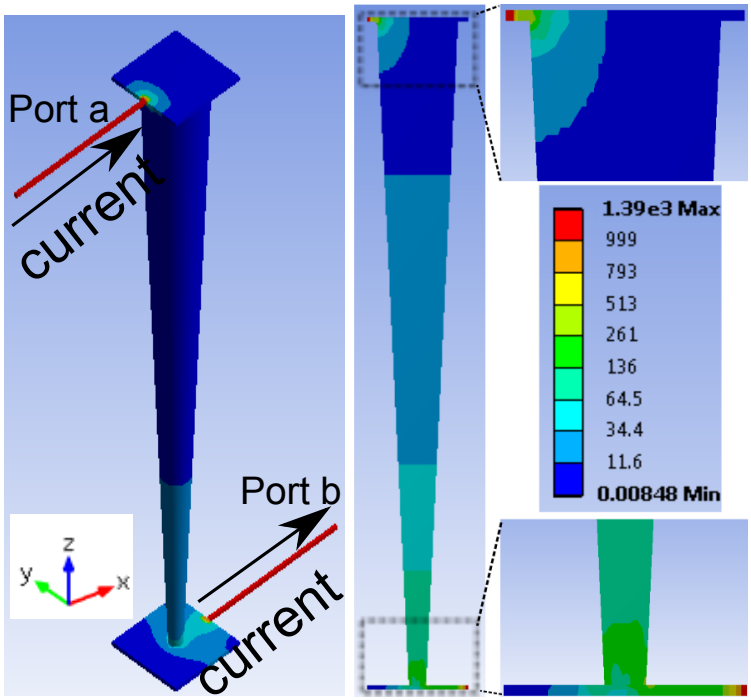
4.5.1 Simulation Setup

The tapered TSV structure we use for simulation and the baseline layout design are shown in Figure 4.8(a). The TSV's layout comprises three components: (1) The tapered TSV cone (top diameter is $4\mu m$, while the bottom diameter is $0.856\mu m$, and the TSV height is $30\mu m$), (2) two landing pads (length $\times$ width $\times$ height $= 5\mu m \times 5\mu m \times 0.2\mu m$), and (3) two signal wires (length $\times$ width $\times$ height $= 0.2\mu m \times 0.2\mu m \times 10\mu m$). One current source is inserted at the top left corner, feeding 0.4mA current, and one current sink is inserted at the bottom right corner, absorbing the 0.4mA current. The tapering angle is set to be 87°.

The non-uniform TSV resistive model (Algorithm 3) dynamic programming based delay optimization (Algorithm 4) are implemented in Matlab with a Intel Core i5 3.1GHz CPU and 8GB memory.

4.5.2 Non-uniform TSV resistive model

The runtime vs accuracy trade-off comparison between our non-uniform meshing approach in Section 4.3 and uniform meshing in [3] is highlighted in Figure 4.9. Both approaches' results are compared to ANSYS FEM-based simulation. Our non-uniform model has significantly higher accuracy under the same running time, compared to the approach used in [3].



(a) (b)

Figure 4.8: (a) Baseline layout structure for Tapered TSV, and the current density in the tapered TSV and signal wires. (b) Tapered TSV's current density distribution at side view.

Figure 4.10 shows both the uniform mesh [3] and our non-uniform mesh's grid and current density distribution on tapered TSV's bottom pad. Our non-uniform mesh strategy only provides finer mesh for areas which have high current density values. Since we are mostly interested in regions with large current densities, those grids with small current density values can be ignored.

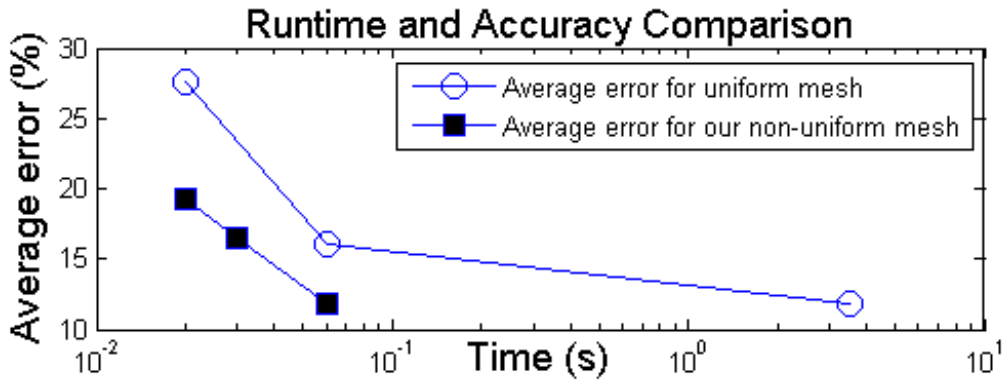


Figure 4.9: Average error (compared to ANSYS result) in percentage.

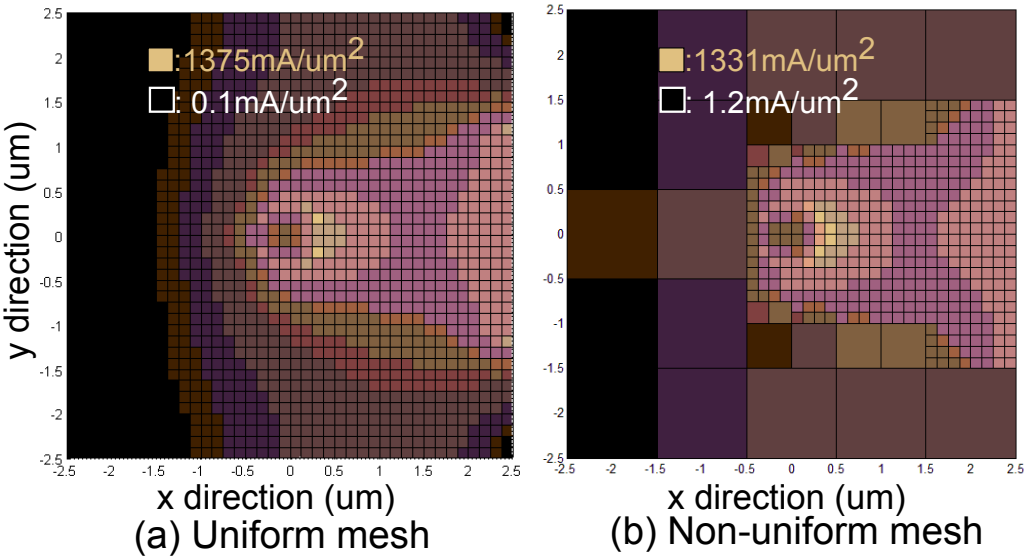


Figure 4.10: (a) Zhao's [3] and (b) our non-uniform mesh grid and current distribution at tapered TSV's bottom pad.

4.5.3 Wire Sizing for Peak Current Reduction (Problem 1)

Now we present our results for solving Problem 1. We model the signal wires as several segments with the same length, and fix the widths of all but w_n and w_{n+1} (see Figure 4.7). We apply our non-uniform model to figure out how w_n,w_{n+1} affects the peak current density within TSV (at different tapering angles, θ. $\theta = 90$ corresponds to cylindrical TSV). The results are shown in Table 4.1.

Table 4.1: Impact of TSV's angle (θ) and w_n ($= w_{n+1}$, in μm)on TSV's peak current density (in $mA/\mu m^2$)

θ /$w_n = w_{n+1}$	0.25	1.25	2.5	3.75	5
87	1414	464	275	272	229
88	1414	462.2	246.3	223.6	214.5
89	1414	456.4	239.7	180.8	166.5
90	1414	437.7	224.6	176.0	157.3

Table 4.1 indicates the following observations: (1) For a fixed tapering angle θ, increasing w_n(or w_{n+1}) monotonically reduces the peak current (more data points omitted for brevity); (2) As the TSV becomes more tapered, peak current increases.

4.5.4 Wire sizing for delay optimization (Problem 2)

Wire sizing optimization aims to minimize the second-order delay along a path from source to sink while keeping the current density under a certain level. TSV is modeled as a single resistor and capacitor. In this chapter, $L = 200\mu m$, $R_D = 50\Omega$,

$C_L = 0.1pF$, $r_0 = 0.084\Omega \cdot \mu m^2$, $c_0 = 4 \times 10^{-17} F/\mu m$, $C_{TSV} = 1 \times 10^{-13} F$, $R_{TSV} = 0.2\Omega$, and $n = 10$ (refer to Figure 4.7 for definition), the results are summarized in Table 4.2.

Table 4.2: TSV's delay (in ps) and peak current density (in $mA/\mu m^2$) co-optimization process

	Before solving P1		After solving P1		After solving P2	
θ	I_{peak}	$D_{a->b}$	I_{peak}	$D_{a->b}$	I_{peak}	$D_{a->b}$
87	1414	38.96	275	35.87	275	6.43
88	1414	32.90	246.3	30.36	246.3	5.64
89	1414	28.70	239.7	26.29	239.7	5.19
90	1414	25.61	224.6	23.65	224.6	4.62

In Table 4.2, before solving Problem 1, all wire segments have the same width, which is $0.25\mu m$. This configuration gives large current density values (2^{nd} column) and large delay (3^{rd} column). After solving Problem 1, the current density reduces (4^{th} column) but the delay remains high (5^{th} column). Then we perform Algorithm 4, and after solving Problem 2, the second-order delay effectively reduces (7^{th} column), while the peak current density is still under control (6^{th} column).

4.6 Summary

In this chapter, we have investigated the TSV's tapering effect, which is an inevitable byproduct of the TSV's fabrication process, and explore its impact on TSV's current density distribution. Through detailed FEM simulation, we have

shown that the TSV's tapering effect causes significant DC current crowding effect at the narrower end of the TSV, worsens the TSV's EM reliability. To overcome the fact that the FEM based simulation is not computationally efficient, we develop a fast and accurate non-uniform TSV mesh model for estimating the peak current density inside the TSV. We then formulate a problem of deciding the optimal wire layout such that the peak current density inside a TSV, and second-order delay, are minimized. By using an efficient dynamic programming based approach, we achieve significant reduction in both peak current delay density and TSV-involved path delay.

Chapter 5: Reliability Aware Placement for 3D ICs

In Chapter 4, current density simulation and thermal-mechanical simulation were performed for a single TSV. We used the peak current density inside a TSV as an EM constraint, and formulated a wire sizing problem to optimize the delay for a TSV-involved path. However, many recent 3D IC experimental works have shown that the current density is not the only driving force for TSV's EM. In this chapter, we start with the state-of-the-art EM analysis for TSVs, and then we develop detailed simulation framework to capture TSV's EM transient under various factors. We also introduce the methodology to investigate 3D IC's material fracture trend. The TSV's EM and material fracture models are then integrated into 3D IC's placement flow, which aims to optimize both 3D IC's performance and reliability.

This chapter is organized as follows. We present an overview of our co-design method in Section 6.1. We introduce several related models and placement approaches in Section 5.2, in which we briefly cover the TSV's EM model, 3D IC's material fracture model, TSV-induced thermal mechanical stress model and 3D-IC's compact thermal model. In the next section, Section 5.4, we derive and validate a simple TSV's EM objective function. We define our reliability-aware TSV placement problem in Section 5.6, and the following Section 5.7 provides an overview of

our co-design scheme. Three primary optimization steps in the co-design scheme are elaborated from Section 5.9 to 5.8. Experimental setup, results and discussions are presented in Section 6.5, and Section 6.6 summarizes this chapter.

5.1 Introduction

One of the most crucial reliability issues in 3D-IC is the EM in TSVs. The reason for this is twofold: (1) TSV's EM is strong function of the thermal mechanical stress, and the high mechanical stress is brought by CTE mismatch between TSV (i.e. copper $1.77 \times 10^{-5} K^{-1}$) and silicon substrate ($3.05 \times 10^{-6} K^{-1}$). (2) TSV's EM is also a strong function of temperature. Local hotspots are expected to appear in 3D-ICs due to the poor heat conduction in the stacked structure (conductivity of inter-layer dielectric is around $1W \cdot m^{-1} \cdot K^{-1}$, and the conductivity of Si is $149W \cdot m^{-1} \cdot K^{-1}$). Local hotspot makes atoms in TSVs diffuse faster.

As the TSV's EM trend is influenced by mechanical stress and chip's thermal profile, one natural thought is to deliberately create a "desirable" stress and thermal profile such that the TSV's EM is minimized. For 3D-IC's mechanical stress profile, several previous works have found that the mechanical stress has a superposition effect, which means the TSV-induced stress field at a certain location is a function of the locations of all other TSVs [11, 62, 63]. Therefore, one way to manipulate the stress profile is to change TSV's relative locations. As for altering the thermal profile, one solution that many researchers have been actively investigating is inserting

thermal TSVs (dummy TSVs). These thermal TSVs have no electrical functionality, but establish additional vertical heat conduction path from power sources to the heat spreader.

We consider optimizing the placement of TSVs (both signal and thermal TSVs) to deal with TSV's EM degradation. We believe that solving this TSV placement problem needs the electrical-thermal-reliability co-design methodology. The impact of TSV's placement on circuit's electrical performance, chip's thermal profile, and the TSV's EM property shall be considered simultaneously. Otherwise, randomly move the signal TSV around to obtain a good stress profile might degenerate the electrical performance, and blindly inserting thermal TSVs to seek a desirable thermal profile might even worsen the TSV's EM trend since thermal TSVs generate mechanical stress fields as well. Therefore, a holistic and systematic co-design scheme for TSV placement that optimizes both 3D-IC's electrical performance and thermal-mechanical reliability becomes highly necessary.

While lots of efforts have been made on the development of 3D electrical design automation (EDA) tools, the consideration of TSV's EM during the electrical design stage is lacking. One important drawback is the physical equation that describes TSV's EM is too complicated to be integrated in any EDA tool.

In this chapter, we model TSV's EM with considerations of thermal mechanical stress and TSV's temperature, and then propose an electrical-thermal-reliability co-design scheme to place the signal and thermal TSVs such that the signal TSV's average EM is minimized while 3D-IC's electrical performance is maximized. Our co-design method is better than incorporating the reliability awareness after the

electrical design, because the co-design method is able to balance conflicting objectives, and search for compromises in a much bigger solution space while the first-electrical-then-reliability method somehow loses this global view. Specifically our contributions are as follows:

1. We derive and validate a simple yet accurate physics-based TSV's EM objective function, which takes into account the mutual EM influence between one TSV and its neighboring TSV as well as temperature, and can easily be incorporated into prevailing 3D placement algorithms.

2. We propose an electrical-thermal-reliability co-design flow that is capable of considering the number and the locations of TSVs to optimize the TSV's EM during the electrical design stage.

3. We formulate a quadratic programming problem to insert thermal TSVs to the layout whitespace to generate a thermal profile which minimizes the signal TSV's average EM.

4. We propose an linear programing (LP) relaxation for solving an EM-desired TSV distribution to minimize the signal TSV's average EM.

The simulation result shows that the proposed co-design method prolongs the TSV's average mean-time-to-failure (MTTF) by 3.24**x**, with a mere 1% performance overhead, when compared to the conventional EM-unaware 3D design such as 3D-craft [64].

5.2 Background

In this section, we introduce the TSV EM's equation, TSV's stress model, the von Mises material yield criterion, and related computational methods for chip-scale stress and thermal estimation. We also briefly cover some notable existing 3D placement approaches, which usually aim at maximizing electrical performance and/or peaking temperature, while approaches that can handle TSV's EM are lacking.

5.2.1 TSV's EM Physics

The physics of TSV's EM can be described by the time-dependent multi-physics mass transportation equations (Equation 5.1).

$$\frac{\partial c}{\partial t} + \nabla \cdot \left[-D \bigtriangledown c + \frac{Dc\overrightarrow{j}e\rho Z}{kT} + \frac{Dc\Omega}{kT} \cdot (\nabla(\sigma)) + \frac{DcQ^{\star}}{kT} \cdot \frac{\nabla(T)}{T} \right] = 0 \quad (5.1a)$$

$$D = D_0 \cdot exp(\frac{\Omega\sigma - E_a}{kT}) \quad (5.1b)$$

$$\sigma = \frac{\sigma_x + \sigma_y + \sigma_z}{3} \quad (5.1c)$$

where c is atomic concentration, t is time, D is diffusivity, j is current density, e is electron charge, Z is effective charge, Ω is atomic volume, σ is hydrostatic stress, $Q^{\star}$ is heat of transport, ρ is resistivity, k is Boltzmann constant, T is temperature, and E_a is activation energy. The hydrostatic stress, σ, as shown in Equation 5.1c, is a scalar value, which is the average of principal stress values along x, y, and z axis.

5.2.2 The Von Mises Yield Criterion

The von Mises yield criterion suggests that the yielding of materials begins when the von Mises stress exceeds material dependent yield strength [9–11]. The von Mises stress is a scalar stress value that can be computed from the stress tensor in the Cartesian coordinate system, as follows.

$$\sigma_v = \frac{1}{\sqrt{2}}\Big[(\sigma_{xx}-\sigma_{yy})^2+(\sigma_{yy}-\sigma_{zz})^2+(\sigma_{zz}-\sigma_{xx})^2 +6(\sigma_{xy}^2+\sigma_{yz}^2+\sigma_{zx}^2)\Big]^{\frac{1}{2}}. \tag{5.2}$$

The stress tensor S_{xyz} in a Cartesian coordinate system can be obtained by applying the following matrix conversion:

$$S_{xyz} = QS_{r\theta z}Q^T. \tag{5.3}$$

where the conversion matrix Q is in the following form:

$$Q = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.4}$$

where θ is the angle between the x-axis and the line from the TSV's center to the point of interest.

5.2.3 Analytical Stress Model for TSVs

Due to the mismatch in the coefficients of thermal expansion (CTEs) between copper (common TSV material) and the silicon substrate, stresses are induced during TSV's manufacturing process. In this chapter, we use a semi-analytical TSV stress model developed in [65], as follows.

$$\sigma_r = \frac{-E_{Cu}(\alpha_{Cu}-\alpha_{Si})\Delta T}{2(1-\nu_{Cu})} = \frac{4c_0\Delta T}{D^2}, if\ r < D/2 \tag{5.5a}$$

$$\sigma_r = \frac{-E_{Cu}(\alpha_{Cu}-\alpha_{Si})\Delta T}{2(1-\nu_{Cu})}\left(\frac{D}{2r}\right)^2 = \frac{c_0\Delta T}{r^2}, if\ r \geq D/2 \tag{5.5b}$$

where material properties E, α, and ν are the Young's modulus, CTE, and Poisson's ratio, ΔT is the thermal load, D is the diameter of TSV and r is the distance from the measured point to the center of the TSV, and $c_0 = \frac{-E(\alpha_{Cu}-\alpha_{Si})}{2(1-\nu_{Cu})}(\frac{D}{2})^2$.

5.2.4 Material Fracture Modeling for 3D ICs

Material fractures in 3D ICs have been observed in forms of interfacial delamination [7] and substrate cracking [8]. Both mathematical [9–11] and FEM-based models [7, 8, 62] have been created and validated for accurate prediction. Among these works, the von Mises yield criterion [9–11,62] is one of the widely acknowledged material fracture failure models.

5.2.4.1 Single TSV

Applying the TSV stress model in Equation 5.5 and the matrix conversion (Equation 5.3), we obtain the stress tensor in a Cartesian coordinate system, S_{xyz}, as follows [65].

$$S_{xyz} = \begin{bmatrix} cos2\theta \cdot \sigma_{rr} & sin2\theta \cdot \sigma_{rr} & 0 \\ sin2\theta \cdot \sigma_{rr} & -cos2\theta \cdot \sigma_{rr} & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5.6}$$

where σ_{rr} is the stress value along radial direction (Equation 5.5a).

Therefore, the von Mises stress for a single TSV-induced stress field can be derived using Equation 5.2 and Equation 5.6.

$$\sigma_v = \sqrt{3} \cdot \sigma_{rr}. \tag{5.7}$$

5.2.4.2 Multiple TSVs

When multiple TSVs are present, the stress tensor at a point of interest (i.e. point j) is the sum of each individual TSV-induced stress field. Similarly, the von Mises stress can be calculated as follows.

$$\begin{aligned}\sigma_v^j =& \sqrt{3} \cdot \sqrt{(\sigma_{xx})^2 + (\sigma_{xy})^2} \\ =& \sqrt{3} \cdot \sqrt{(\sum_{i \in TSV} \sigma_{rr}^{ij} cos2\theta_{ij})^2 + (\sum_{i \in TSV} \sigma_{rr}^{ij} sin2\theta_{ij})^2}\end{aligned} \tag{5.8}$$

where j is a point of interest, σ_{rr}^{ij} is the radial stress value of the i^{th} TSV, and θ_{ij} is the angle between the x-axis and the line from the i^{th} TSV's center to point j.

5.2.5 Stress Superposition Principle

The CTE mismatch between TSV and silicon substrate induces significant thermal mechanical stress during TSV's fabrication process. When multiple TSVs are present, we assume the TSV and silicon substrate are linearly elastic structures, thus according to the stress superposition principle explained in [11,62,63], the stress coming simultaneously from several different bodies is the sum of the stress applied separately. In other words, the stress value at a certain point is the accumulative TSV-induced stress caused by each TSV as follows:

$$\sigma = \sum_{i=1}^{n} \sigma_i. \tag{5.9}$$

where σ is the total stress at the point of interest and σ_i is the stress value at this point induced by the i^{th} TSV.

5.2.6 Compact Thermal Model

In this section, we use the compact thermal model described in [66] to estimate the chip's thermal profile. This thermal model exploits the electrical/thermal duality by considering the temperature value at node i (T_i) as the node voltage at node i, the power consumption at node i (P_i) as the node i's current source, and G_{ij} as the (thermal) conductance between node i and node j. Equation 5.10 shows the resulting Kirchhoff's Current Law (KCL) equation.

$$\begin{bmatrix} G_{1,1} & G_{1,2} & \cdots & G_{1,M} \\ G_{2,1} & G_{2,2} & \cdots & G_{2,M} \\ \vdots & \vdots & \ddots & \vdots \\ G_{M,1} & G_{M,2} & \cdots & G_{M,M} \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ \vdots \\ T_M \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_M \end{bmatrix} \tag{5.10}$$

where T is an $M \times 1$ vector, represents M meshing grids, $G_{M\times M}$ is the transfer thermal conductance matrix, and $P_i(i = 1, \cdots, M)$ are M power sources. Each entry in G, $G_{i,j}$, is the thermal conductance between grid i and grid j.

An illustrative thermal resistive network for a two-layer 3D-IC is shown in Figure 5.1. This 3D-IC consists of one layer of heat sink, two layers of silicon substrate, and three layers of silicon dioxide. TSVs (thermal and signal TSVs) go through both the silicon and the silicon dioxide layer. We assume the temperature

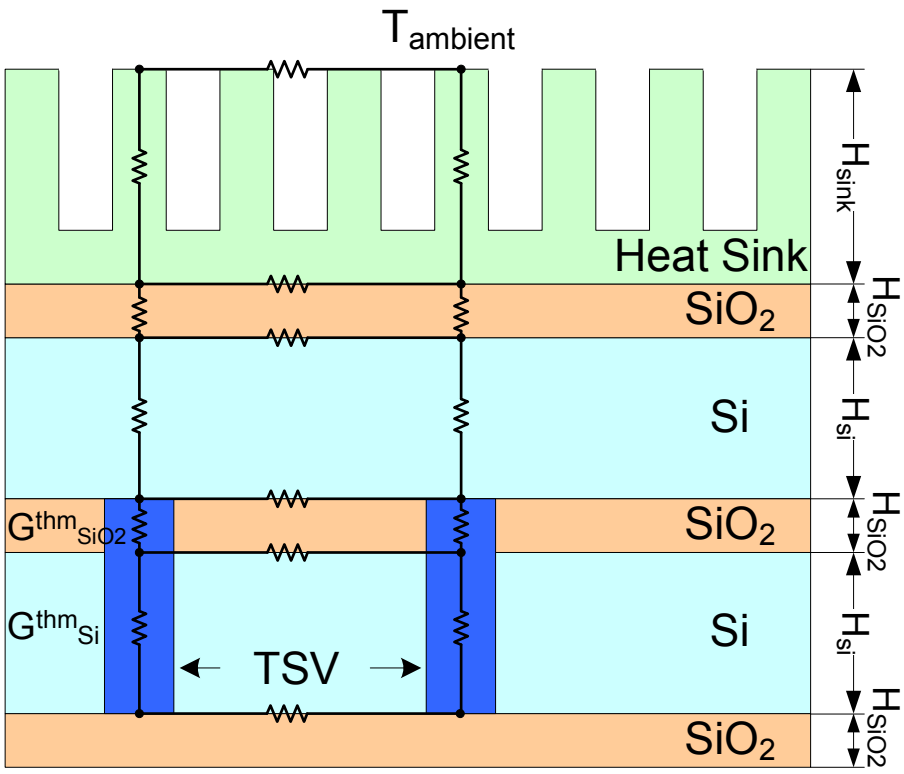


Figure 5.1: Side view for the thermal resistive network of 3D-ICs

at the boundary of the heat sink is a fixed ambient temperature ($T_{ambient}$) and other side of the 3D-IC (bottom SiO_2 layer) is adiabatic. The thermal conductance between two adjacent nodes is calculated using Equation 5.11. If a TSV exists between grid i and grid j, $G_{i,j}$ is the parallel thermal conductance of the silicon substrate and the thermal TSV. Horizontal thermal conductance can be calculated similarly.

$$G_{i,j} = \frac{K \cdot A}{H} \tag{5.11}$$

where K is material's thermal conductivity, A is the cross-sectional area, and H is the thickness of the material.

5.2.7 Existing 3D Placement Approaches

Performance enhancement has been the focus of 3D design tool's development in the past few decades. Some notable works include the analytical placer [64, 67], which uses smoothing technique to handle the placement constraints and is effective for achieving trade-offs between total wire length and peak temperature; force-directed placer [68], which iteratively uses repulsive forces to remove overlaps and to move gates away from thermal hotspots; partition-based placer [69], where thermal-aware net weights are modeled, and then sequential bi-partition is applied with inter-tier z *cuts* to minimize the number of TSVs, and intra-tier x/y *cuts* to minimize the wire length.

Although these works have successfully enhanced the circuit performance and handled thermal issues in 3D-ICs, none of them considers the TSV's EM degradation.

5.3 TSV's Stress/EM Simulation using FEM

In this section we describe our FEM based simulation framework for TSV's thermal mechanical stress and EM. Our simulation framework is presented in Figure 5.2. After deciding TSV's geometry and material, ANSYS Workbench [58] meshes the TSV structure into three-dimensional grids, and numerical simulations are performed to obtain the thermal/stress and current density distribution profiles. These three profiles are input into a differential equation solver [70] to generate the transient of atomic concentration.

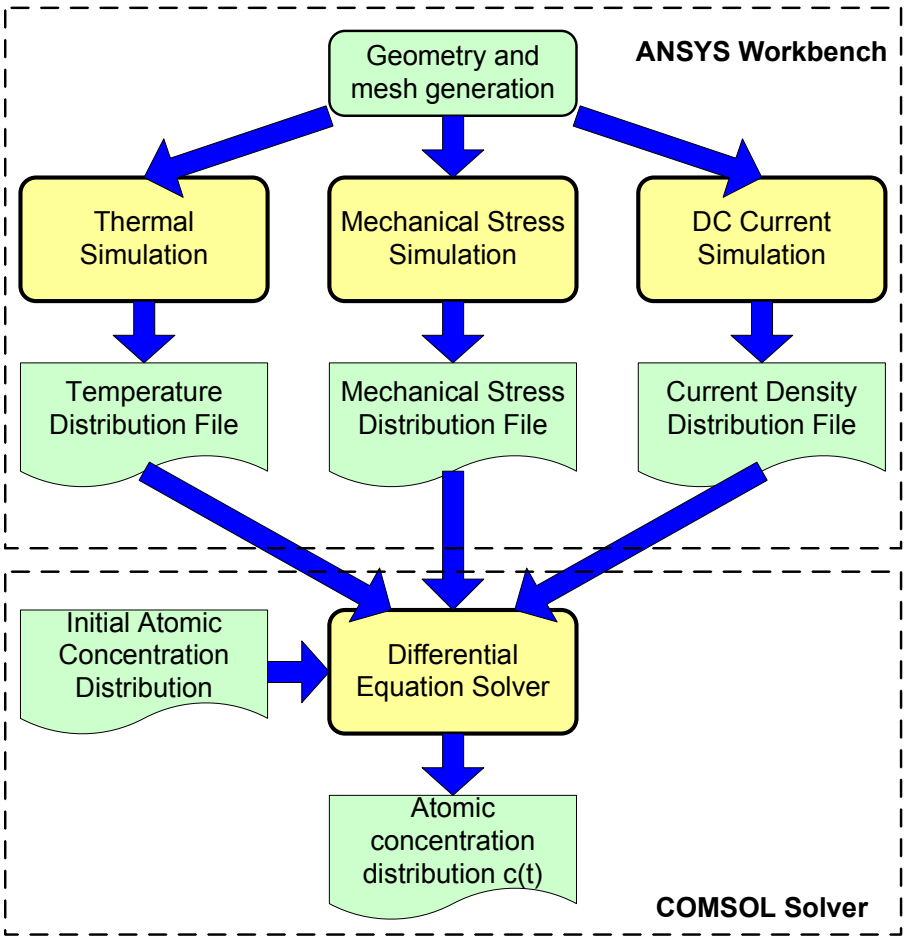


Figure 5.2: TSV's stress and EM simulation flow.

5.3.1 TSV's Stress simulation using FEM

During the IC fabrication process, TSVs endure many thermal cycles. For example, annealing is one of the common steps to remove the lattice damages caused by ion implantation and activate the implanted ions as well. Another example is the wafer bonding process, where high temperature is needed in order to obtain reliable 3D stack. When the whole chip cools down, because of the significant mismatch between different materials' coefficients of the thermal expansion (CTEs), significant thermal mechanical stress remains inside the material. These stress accelerates

material fatigue, roughs the silicon's surface, and shortens the circuit's lifetime. A very simple solution to enhance the device's reliability is to keep the device away from TSV's Keep Out Zone (KOZ). However, large KOZ in the routing area becomes routing obstacles while KOZ in the placement region may potentially enlarges the chip's total area. In this section, we analyze and quantify the magnitude of the thermal mechanical stress in the taper TSV, using a finite element analysis tools called COMSOL [70].

5.3.2 Physical dimensions and simulation setup

Figure 5.3(a) shows the geometry containing the tapered TSV we use for thermal mechanical simulation. It consists of three layers of silicon, a tapered TSV, and bonding materials such as interlayer dielectric (ILD), silicon dioxide and benzocyclobuten (BCB). Material properties used in stress simulation is summarized in Table 5.1, similar to [1].

Table 5.1: Material Property of TSV structure [1]

	Elastic Modulus (GPa)	Poisson's Ratio	CTE (1/K)
Silicon	162	0.28	3.05e-6
Copper	111.5	0.343	1.77e-5
Silicon Oxide	71.7	0.16	5.1e-7
ILD	9.5	0.3	2.0e-5
BCB	6.1	0.35	3.3e-5

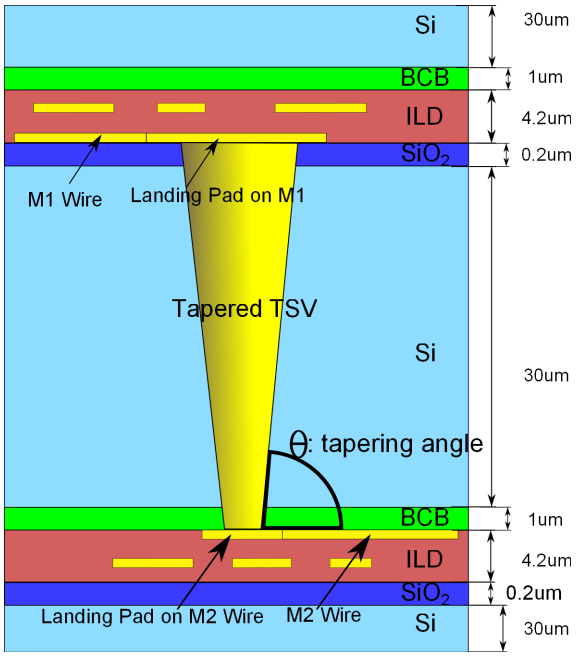


Figure 5.3: Tapered TSV's geometry.

5.3.3 Thermal Simulation

We also calculate the Joule heat produced by the DC current, and then use that as the heat sources to setup the heat conduction equations, for both tapered and cylindrical TSVs. It turns out the TSV tapering has little impact on the temperature increase. The primary Joule heating effect happens in the narrower signal wires. In fact, in both cases, the temperature rises from room temperature (298K) to 323K, increased by 25K. We also notice that the temperature is basically uniformly distributed inside the structure. This is due to the fact that the heat conductivity of the copper is high enough to spread the Joule heat around. In later sections, we'll use 323K as the working temperature of TSVs.

5.3.4 Impact of TSV Tapering on KOZ

Mechanical stress can affect devices in many ways. High magnitude of stress may induce cracks or voids in silicon, or change the devices' mobility thus driving current as well. High variations in stress may also lead to significant EM failures in interconnects. The primary purpose of KOZ is to guarantee that the device is free from unpredictable mechanical stress. The size of the KOZ is usually defined by a stress threshold such that stress magnitude outside the KOZ is smaller than that threshold. In this chapter, for simplicity, we use Von Mises stress as a measurement of thermal mechanical stress. We also assume that the KOZ is in the shape of a circle, and we define the threshold to be 200 MPa. Since the early simulation results show that the temperature gradient inside the TSV structure is not remarkable, we assume the temperature distribution is uniform. We set the annealing temperature to be 573K, and after annealing, the TSV is cooled to a normal working temperature, which is 323K.

The 3D view of the stress distribution of both tapered and cylindrical TSVs is shown in Figure 5.4. Significant high stress values are observed in the contact regions between the copper cone and the silicon substrate (silicon substrate is not shown for readability). This is due to the fact that the CTE of copper ($1.77 \times 10^{-5} K^{-1}$) is significantly larger than silicon ($3.05 \times 10^{-6} K^{-1}$), and when the whole 3D structure is cooled down from the annealing temperature, high compressive stress is formed inside the copper cone.

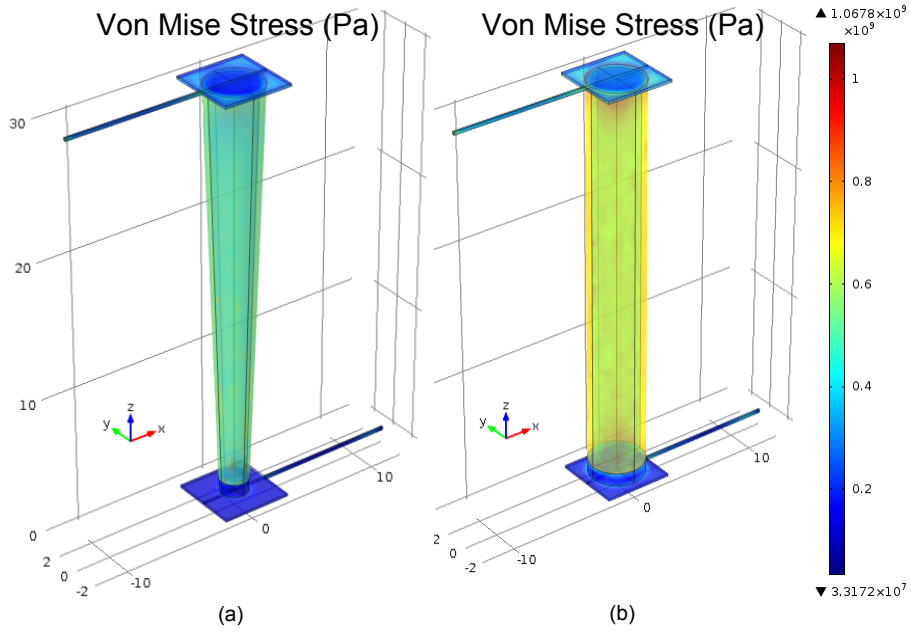


Figure 5.4: Thermal mechanical stress distribution in (a)*tapered* TSV and (b)*cylindrical* TSV.

Now let's quantify the size of the KOZ, using 200MPa Von Mises Stress as a threshold. In a tapered TSV structure, the radius of KOZs are 3.6 μm at the top pad (Figure 5.5(a)), and 0.87 μm at the bottom(Figure 5.5(b)), respectively. That will require an area of 98.5μm^2 and 3.8μm^2 at the top and the bottom.

In a cylindrical TSV structure, however, the radius of KOZs are 3.6 μm at the top pad (Figure 5.6(a)), and 2.65 μm at the bottom(Figure 5.6(b)), respectively. The area requirement at the top and bottom pad is of 98.5μm^2 and 22.1μm^2, respectively. From the simulation result we can clearly see that the tapering does not affect the stress distribution at the top pad, but for the bottom pad, it saves 83% of area, which could be used for wire routing and gate placement.

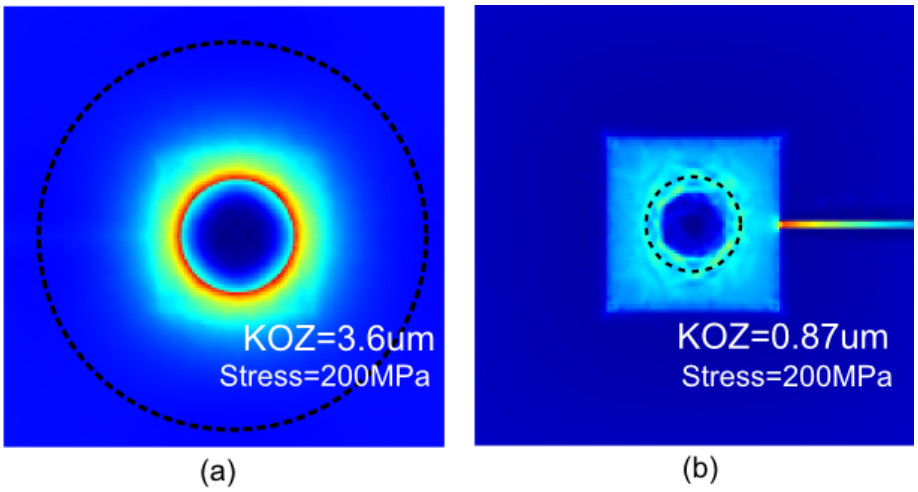


Figure 5.5: Thermal mechanical stress distribution in *Taper* TSV at (a) top pad and (b) bottom pad.

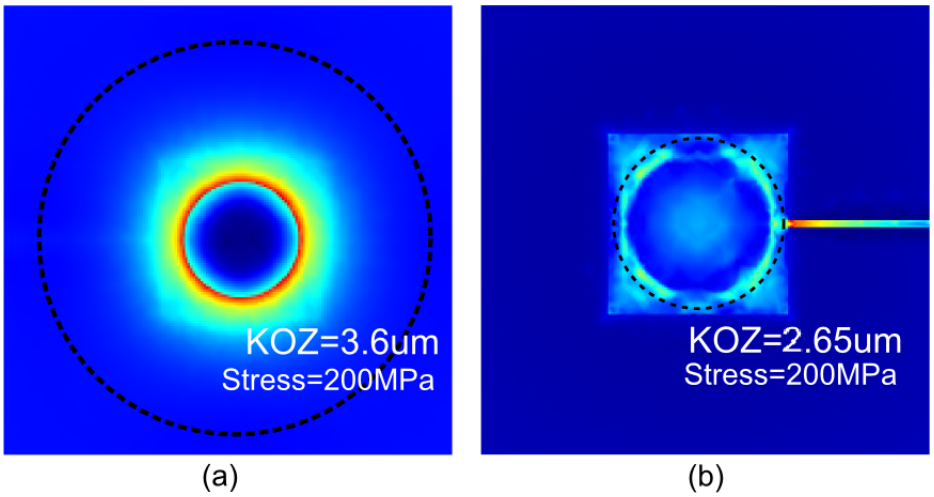


Figure 5.6: Thermal mechanical stress distribution in *cylindrical* TSV at (a) top pad and (b) bottom pad.

5.3.5 TSV's EM simulation using FEM

Equation 5.1 shows all four driving forces of EM: atomic concentration gradient, current density, thermal mechanical stress gradient, and temperature gradient. The atomic concentration c, is difficult to solve analytically. Instead, these mass

transportation equations for single TSV are solved by FEM in the work of [1, 71]. The authors in [1] uses atomic concentration change as a single TSV EM's measurement, while [71]'s authors monitor TSV's resistance change caused by void growth inside one single TSV.

In this section, we set up detailed FEM based solver in COMSOL [70] to numerically simulate a *single* TSV and adjacent planer wire's EM, induced by current density, the TSV's own stress gradient, and thermal gradient caused by Joule heating.

The structure of the TSV and its material property is as follows. TSV comprises a cylindrical cone and two landing pads. Each landing pad is connected to a metal wire. We assume that the TSV diameter is $4\mu m$, landing pad is $5\mu m \times 5\mu m$, and TSV's height is $31.2\mu m$. The wires connecting to both ends of TSV have a dimension of $0.2\mu m \times 0.2\mu m \times 10\mu m$ (width $\times$ height $\times$ length). Wires and TSVs are imposed with a symmetric current signal, with a magnitude of 0.4 mA. In this chapter, we assume that the asymmetrical current induced by asymmetrical rise/fall behavior of wiring buffers is negligible. The current density inside the wires is $1 \times 10^{10} A \cdot m^{-3}$. TSV's annealing temperature is set to be 573K. Wire's joule heating effect is considered in the model to calculate the temperature distribution. The initial atomic concentration is $1.5 \times 10^{28} cm^{-3}$. MTTF is defined as the length of time elapses when the atomic concentration has 5% deviation from the initial value [1, 72, 73]. Though MTTF varies depending on different failure criteria, the trend remains similar. Thus in this chapter we use a normalized MTTF to represent the TSV's lifetime subject to EM.

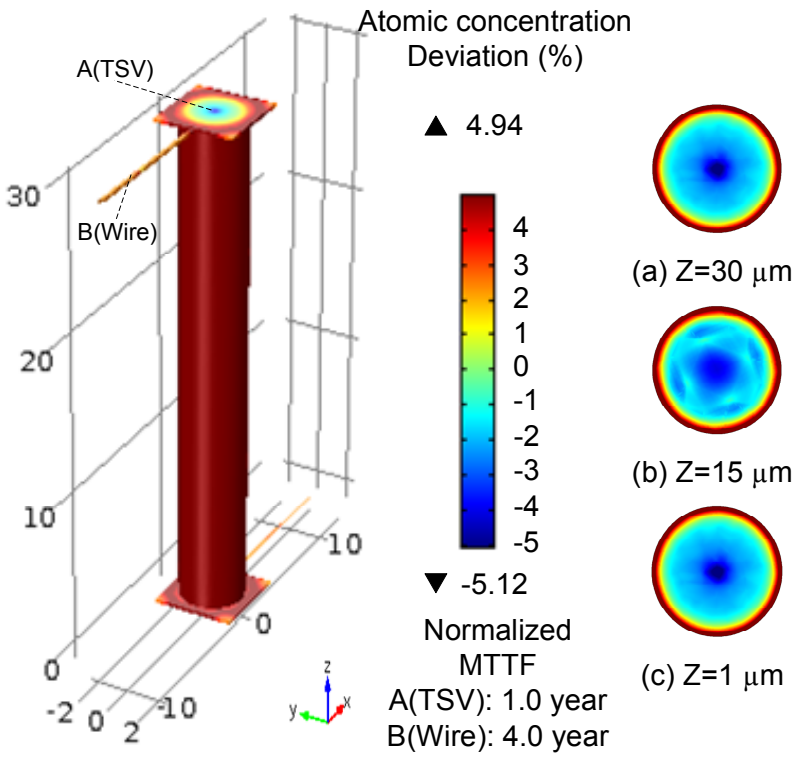


Figure 5.7: Atomic concentration deviation at $t = MTTF$ in a TSV and neighboring planer wire.

Figure 5.7 shows the simulation result. We have three observations from Figure 5.7. (1) Metal atoms migrate from TSV's center to its periphery. Our simulation result matches with some experimental TSV's EM testing data [74], where void appears in the central areas of the TSV. The radial movement of atoms can be explained by the mismatch of coefficient of thermal expansions (CTEs) between TSV filling materials (copper, for example) and the silicon substrate. Since $CTE_{Cu} = 1.77 \times 10^{-5} >> CTE_{Si} = 3.05 \times 10^{-6}$, when the TSV structure is cooled from annealing temperature, radical tensile stress is formed inside the TSV, pulling metal atoms from the center to the periphery. (2) Compared to the EM inside XY plane, the vertical migration of atoms is far less significant. This is because

the major stress gradient happens at the interface between TSV cone and the substrate, which is inside XY plane. Although current flows along Z direction, it is bi-directional thus its impact on the EM is less significant. Previous EM study [1] also supports our observation, where the vertical EM is far less significant than the EM in the XY plane. (3) TSV's MTTF is four times longer than the planar wire's lifetime, making the TSV, rather than the wire, become the most EM-vulnerable device in 3D ICs.

Besides the EM trends, we also observe that the temperature distribution inside the TSVs is quite uniform. This is because copper is an ideal heat conductor, which spreads wire and TSV's Joule heat evenly. The detailed thermal distribution is omitted for brevity.

5.4 TSV's EM Objective Function

Solving the differential equation (Equation 5.1) using FEM-based TSV's EM model takes about one hour for a single TSV. As thousands of TSVs with different temperature and stress distributions are practically used in real circuit designs, FEM model is not capable of handling chip-scale TSV's EM optimization. In this section, we propose a TSV's EM objective function that faithfully represents the intensity of TSV's EM, and is simple enough to be used in optimization engine, which is shown in Equation 5.12 [75]. The following two subsections derive our EM objective

function from Equation 5.1 and also validate it to FEM-based simulation results.

$$OBJ\ (SingleTSV) : \frac{1}{T \cdot exp(\frac{E_a}{kT})} \cdot \left(\frac{\partial^2 \sigma}{\partial x^2} + \frac{\partial^2 \sigma}{\partial y^2} \right) \tag{5.12}$$

5.4.1 Model Derivation

As described in Equation 5.1, atomic concentration gradient, current density, thermal mechanical stress gradient, and temperature gradient are four driving forces for EM. However, as current inside signal TSV is bi-directional, temperature is uniformly distributed inside the TSV, and the atomic concentration gradient is a "recovery force" once EM happens, we consider the thermal mechanical stress gradient as the primary EM driving force for TSV's EM. Meanwhile, as $\Omega\sigma << E_a$, Equation 5.1 can be simplified as below.

$$\frac{\partial c}{\partial t} + D_0 \cdot exp\left(\frac{-E_a}{kT} \right) \nabla \cdot \left[\frac{c\Omega}{kT} \cdot (\nabla(\sigma)) \right] = 0 \tag{5.13}$$

Both previous simulations work [1] and TSV's stress model [65] show that TSV's EM trend is more significant in the XY plane rather than along Z direction, therefore Equation 5.13 can be further simplified as follows.

$$\frac{\partial c}{\partial t} + \frac{D\Omega}{kT} \left[\frac{\partial c}{\partial x} \cdot \frac{\partial \sigma}{\partial x} + \frac{\partial c}{\partial y} \cdot \frac{\partial \sigma}{\partial y} + c \cdot \left(\frac{\partial^2 \sigma}{\partial x^2} + \frac{\partial^2 \sigma}{\partial y^2} \right) \right] = 0 \tag{5.14}$$

As we regard 5% derivation as the EM failure criteria, $\frac{\partial c}{\partial x}$, $\frac{\partial c}{\partial y} << c$, and finally we have:

$$\frac{\partial c}{\partial t} + \frac{D_0 \Omega}{kT} \cdot exp\left(\frac{-E_a}{kT}\right)\left[c \cdot \left(\frac{\partial^2 \sigma}{\partial x^2} + \frac{\partial^2 \sigma}{\partial y^2}\right)\right] = 0 \tag{5.15}$$

For a given TSV distribution and thermal profile, which result in a fixed σ and T, Equation 5.15 is a simple ODE, leading to the following MTTF solution (Equation 5.16).

$$MTTF = \frac{kT \cdot exp\left(\frac{E_a}{kT}\right)}{D_0 \Omega \cdot \left(\frac{\partial^2 \sigma}{\partial x^2} + \frac{\partial^2 \sigma}{\partial y^2}\right)} \cdot ln\left(\frac{c_0}{c_t}\right) \tag{5.16}$$

where c_0 is the initial atomic concentration, and c_t is the threshold concentration at which TSV fails.

From Equation 5.16 it is clear that the proposed objective function (Equation 5.12) can theoretically be an indicator of TSV's EM.

5.4.2 Model Validation

In this section, we set up FEM simulations to validate the proposed TSV's EM objective function (Equation 5.12). Five cases of TSV placement configuration are employed in our experiment. All five cases are performed in an FEM simulation framework implemented in COMSOL [70], using the multi-physics mass transportation equation as in Equation 5.1. As shown in Figure 5.8(a)-(e), the central TSV (yellow) is under influences of one or multiple neighboring TSV's (blue) thermal mechanical stress field. For each of the cases in Figure 5.8(a)-(d), we change the distance between central and neighboring TSV's center, d, from $9\mu m$ to $15\mu m$ (the

diameter of the TSV is set to be $8\mu m$). In Figure 5.8(e), neighboring TSVs are randomly placed. TSV's temperature is set to 298 K for Figure 5.8(a) to (d), while random thermal map is applied to Figure 5.8(e).

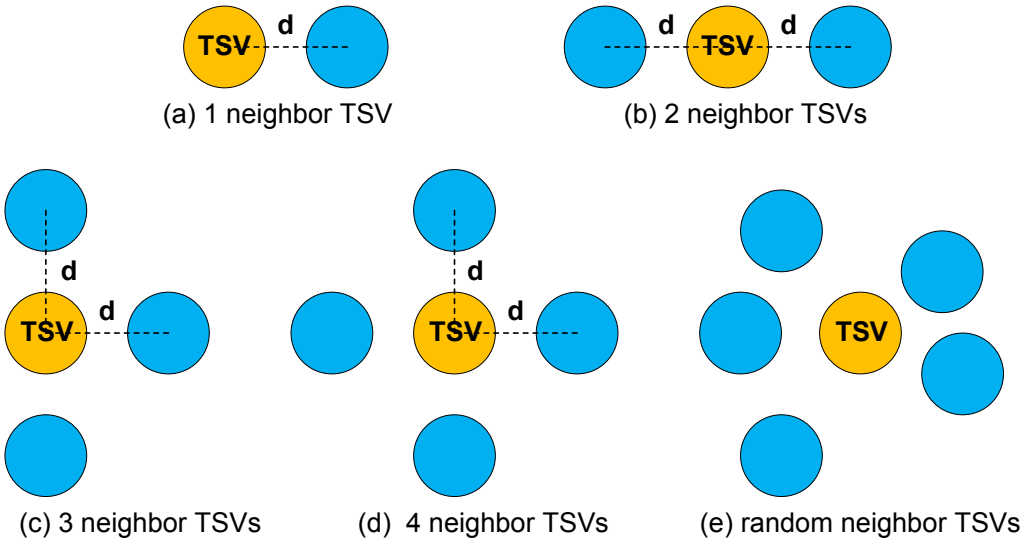


Figure 5.8: TSV placement configurations

The FEM based simulation results are summarized in Figure 5.9. As the distance (d) between central and neighboring TSVs uniformly increases, TSV's MTTF increases. However, when TSV distance (d) is fixed, increasing the number of neighboring TSVs does not always degrade TSV's MTTF. For instance, Figure 5.9(a) has the fewest neighboring TSVs but the shortest MTTF value in all cases, while Figure 5.9(d) possesses the most neighboring TSVs but the longest MTTF value. This is because the configuration of neighboring TSVs in Figure 5.9(d) produces the lowest stress gradients on the central TSV. These observations imply that when multiple TSVs are present, the angle and relative distance between the central TSV and the neighboring TSVs are vital in deciding the central TSV's EM lifetime. In

other words, the idea of simply keeping TSVs away from each other by applying KOZ might not guarantee the optimal EM reliability. A quantitative physics-based objective function, such as Equation 5.12 is fundamentally better than the KOZ approach to account for TSV's mutual influences on stress as well as EM.

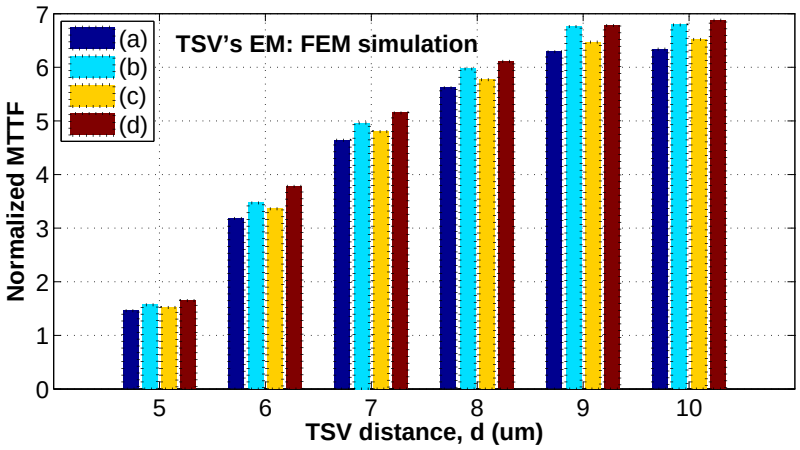


Figure 5.9: TSV's MTTF using FEM simulation for Figure 5.8(a)-(d), with various TSV distance.

Figure 5.10 presents the correlation between the normalized MTTF values calculated using Equation 5.16 using our objective function (Equation 5.12) and the normalized MTTF from FEM simulation.

We use the Pearson's Correlation Coefficient (PCC) to quantify the correlation between our objective function, and the FEM simulation result. PCC is the normalized covariance among two vectors, which is a mathematical way of capturing how good a model is to real data. The PCC in our case reaches 0.98 (near 1), meaning our TSV's EM objective function is highly correlated with FEM simulation results.

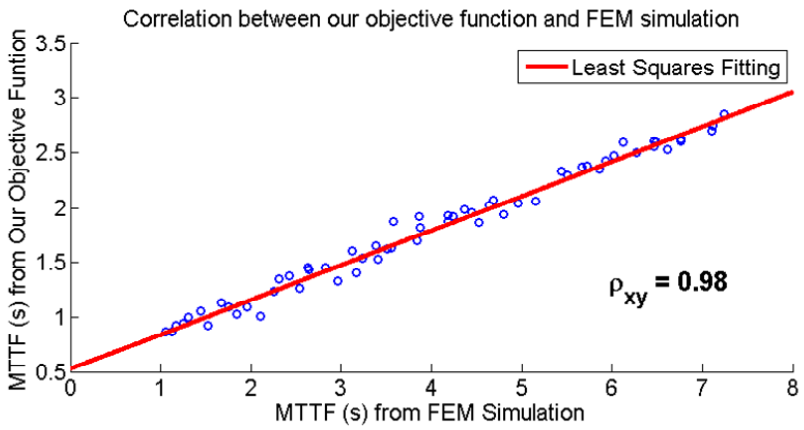


Figure 5.10: Correlation between the MTTF calculated using our objective function and using FEM simulation

5.5 3D IC's Material Fracture Objective Function

We propose a material fracture objective function (refereed as the fracture rate in later sections), which employs the square of the von Mises stress (Equation 5.8). The proposed material fracture objective function is shown in Equation 5.17 [76].

$$OBJ_f \ : 3 \cdot \sum_{j \in grid} \left[\left(\sum_{i \in TSV} \sigma_{rr}^{ij} cos2\theta_{ij} \right)^2 + \left(\sum_{i \in TSV} \sigma_{rr}^{ij} sin2\theta_{ij} \right)^2 \right]. \tag{5.17}$$

5.6 Problem Definition

As mentioned before, TSV's EM is a strong function of thermal mechanical stress and TSV's temperature. The stress profile depends on TSV locations as well as the thermal profile. Therefore the stress profile is impacted by both the gate placement as well as the TSV placement. In this chapter, we are interested

in placing both logic gates and TSVs in the 3D space such that a desired stress and thermal profile are obtained, which create the optimal balance between wire length and TSV's EM. Specifically, we investigate the problem of deciding the count and location of TSVs, the location of logic cells such that the TSV's average EM is minimized, while the wire length overhead is small, as compared to conventional EM-unaware 3D placers such as 3Dcraft [64]. The TSV's EM objective function (Equation 5.12) and material fracture objective function (Equation 5.17) are combined to form a unified objective function. The formal problem definition is presented as follows.

Problem Statement Given a circuit netlist (E) and each logic gate's power consumption (P), we'd like to find the number and locations of TSVs and the locations of logic gate in the 3D space, such that the TSV's average EM and overall wire length are minimized:

$$\begin{aligned} min: & \sum_{e \in E} \left[WL(e) + \alpha_{TSV(e)} \cdot TSV(e) \right] + \\ & + \beta_{EM} \left[\sum_{i \in TSV} \frac{1}{T_i \cdot exp(\frac{E_a}{kT_i})} \cdot (\frac{\partial^2 \sigma_i}{\partial x^2} + \frac{\partial^2 \sigma_i}{\partial y^2}) \right] \end{aligned} \tag{5.18}$$

where wire length $WL(e)$ and $TSV(e)$ are half-perimeter wire length (HPWL) of wires and TSVs, the coefficient α_{TSV}, β_{EM} are the weight for TSV usage, and TSV's EM, respectively, and the last term is the total EM for all the TSVs.

5.7 Overview of Reliability-aware Placer

In this section, we briefly cover the primary steps to solve the optimization problem (Equation 5.18). We assume 3Dcraft, proposed in [64], gives a high-quality placement solution that minimizes the total wire length. Then the locations of all logic gates and TSVs are perturbed in order to minimize the TSV's average EM, while ensuring small wire length overhead. Iterative optimizations are performed between the following three optimization engines: TSV placement engine, temperature estimation engine, and logic cell's re-placement engine.

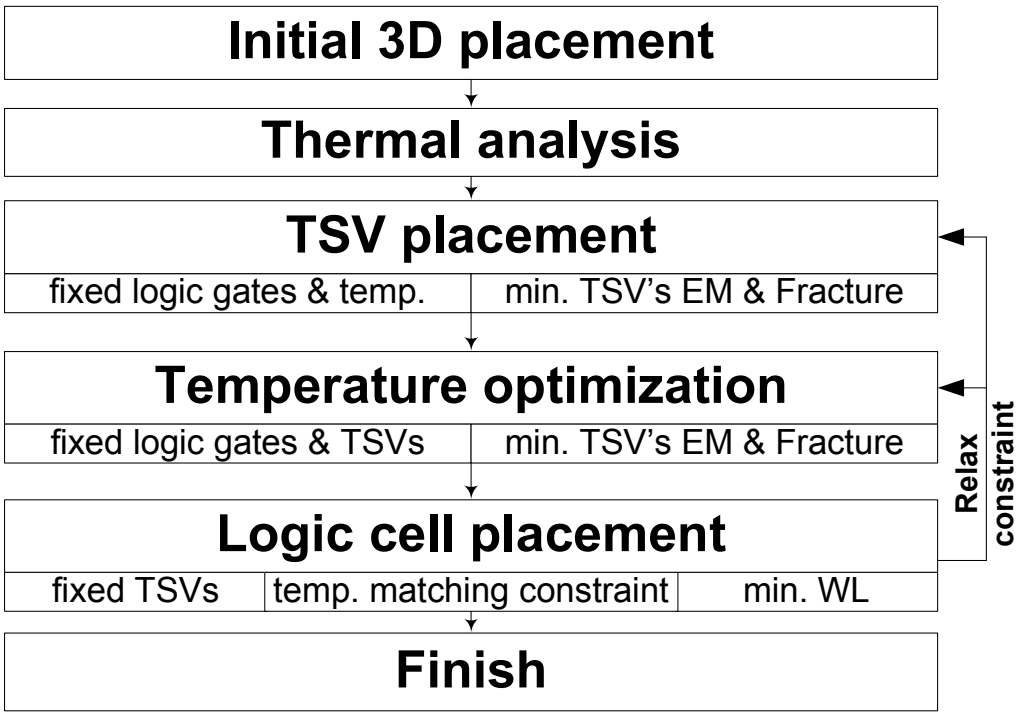


Figure 5.11: Overall design flow for Reliability-aware 3D placer

Figure 5.11 presents the overall design flow for the proposed 3D placement flow. First of all an initial 3D placement solution is generated using any of the conventional 3D placement tools, including analytical placers such as [64,67], force-directed approach based placers such as [68], and partition-based placer such as [69]. After initial placement generation, we obtain an initial chip-scale thermal profile by applying the thermal model described in [66]. The next step is to re-place all the TSVs to minimize both the average EM rate and the average von Mises stress, assuming logic gate's placement is fixed. Although the thermal profile would be affected by the location of TSVs, at this point this impact is ignored. A QP problem is formulated to handle this TSV placement problem. Then we fix the locations of all logic gates and TSVs, and the temperature estimation engine estimates a desirable temperature allocation for each TSV such that the both the average TSV EM rate and the average von Mises stress are minimized. An effective heuristic for the temperature estimation engine is proposed. This is followed by the logic cell's re-placement engine, which aims to match this desired thermal profile by perturbing logic cell's placement. The logic cell's re-placement engine is based on SA. The logic cell's re-placement engine feeds its output back to temperature estimation engine if it SA fails to find a feasible solution and the corresponding TSV temperature range is modified in the temperature estimation engine, otherwise the next iteration begins from the TSV re-placement engine again. The iteration runs until no significant reliability improvement is observed in consecutive runs or the a user-specified wire length overhead threshold has been is reached.

5.8 TSV Placement Engine

Given an initial logic gate's placement, an initial TSV distribution, and the resulting chip-scale thermal profile, the TSV re-placement engine aims to optimize the location of each TSV, such that the average von Mises stress value and the average EM rate are minimum.

We assume after initial placement, all the logic gates can only be moved horizontally, and cannot be relocated to another layer. Under this assumption, number of TSVs is a constant in all following optimization engines.

As TSV re-placement engine begins, each TSV is assigned with a bounding box (BB). A TSV's BB is the layout whitespace near TSV's initial location. The size of BB is decided by the amount of wire length penalty we are willing to tolerate, due to TSV reallocation. A TSV can only move inside its BB.

For the sake of explanation, we assume a 2-layer 3D IC in this section, although our algorithm can be extended to more than two layers as well. Our experimental section shows a 4-layer 3D IC. TSVs and their BBs are numbered from 1 to n, where n is the total number of TSVs. Each grid inside k^{th} BB has an unknown variable

$x_{k,a,b}$, where (a, b) is the grid coordinate. Note that one grid could have multiple variables if multiple BBs overlap at this grid. Specifically $x_{k,a,b}$ is a Boolean variable defined as follows.

$$x_{k,a,b} = \begin{cases} 1 & if\ k^{th}\ TSV\ is\ located\ at\ grid\ (a,b); \\ 0 & otherwise. \end{cases} \tag{5.19}$$

We denote the stress value at grid (i,j) as $\sigma_{i,j}$. We also denote single TSV's stress profile as σ_{xyz}, which can be pre-computed under different thermal loads using Equation 5.5 and Equation 5.6. As explained earlier, at this stage logic gate's locations are given therefore the thermal profile is fixed. Thus the TSV-induced stress profile σ_{xyz} is a constant for a given temperature. When multiple TSVs are present, $\sigma_{(i,j)}$ can be expressed according to Equation 5.9, as follows.

$$\sigma_{xx}^{(i,j)} = -\sigma_{yy}^{(i,j)} = \sum_{a=1}^{m}\sum_{b=1}^{m}\sigma_{xx}^{a,b,i,j}\sum_{k=1}^{n}x_{k,a,b}. \tag{5.20a}$$

$$\sigma_{xy}^{(i,j)} = \sigma_{yx}^{(i,j)} = \sum_{a=1}^{m}\sum_{b=1}^{m}\sigma_{xy}^{a,b,i,j}\sum_{k=1}^{n}x_{k,a,b}. \tag{5.20b}$$

$$\sigma_{zz}^{(i,j)} = \sigma_{xz}^{(i,j)} = \sigma_{yz}^{(i,j)} = 0. \tag{5.20c}$$

where m is the horizontal mesh size, n is the total number of TSVs, $\sigma_{a,b,i,j}$ is the stress value at the grid (i,j) induced by a single TSV centered at grid (a,b), σ_{xx} and σ_{xy} are the xx and xy stress components, and the term $\sum_{k=1}^{n} x_{k,a,b}$ denotes whether there is a TSV placed at grid (a,b) or not.

Then we formulate an optimization problem combining Equation 5.18, Equation 5.19 and Equation 5.20. Two constraints are imposed: (1) the k^{th} TSV is located inside its BB; (2) Each grid can have no more than one TSV. Considering multiple BBs might overlap, we allow multiple TSVs placed within the same BB.

$$
\begin{aligned}
Min: & \sum_{(i,j)\in BB} \frac{1}{T_{i,j}\cdot exp(\frac{E_a}{kT_{i,j}})}\cdot\left[\frac{\partial^2\sigma_{(i,j)}}{\partial x^2}+\frac{\partial^2\sigma_{(i,j)}}{\partial y^2}\right] \\
& + \frac{3\gamma_f}{\beta_{EM}} \sum_{(i,j)\in BB}\left[(\sigma_{xx}^{(i,j)})^2+(\sigma_{xy}^{(i,j)})^2\right] \\
s.t. & \sum_{(a,b)\in k'sBB} x_{k,a,b}=1, \quad \forall k; \\
& \sum_{k=1}^{m} x_{k,a,b}\le 1, \quad \forall a,b; \\
& x_{k,a,b}=0 \ or \ 1, \quad \forall k,a,b.
\end{aligned}
\tag{5.21}
$$

where the objective function is a weighted sum of the total EM rate and total fracture rate. $T_{i,j}$ is the temperature at the grid (i,j), and the coefficient β_{EM}, γ_f are the weights for EM, and material fracture, respectively.

The temperature distribution T is a fixed distribution under the assumption that the logic gate's placement is given, therefore, all the stress components, as shown in Equation 5.20, are linear functions of all $x_{k,a,b}$.

To solve the optimization problem (Equation 5.21) numerically, we further linearize the second order derivative of the stress term by applying forward differencing method, as shown in Equation 5.22. After linearization, the first term in Equation 5.21, the EM rate part, becomes a linear function w.r.t all $x_{k,i,j}$s. Its second term, the fracture rate part, is a sum of squares of a linear function w.r.t all $x_{k,i,j}$s. It's clear this is a QP problem w.r.t. all $x_{k,i,j}$s.

$$\frac{\partial^2 \sigma_{(i,j)}}{\partial x^2} + \frac{\partial^2 \sigma_{(i,j)}}{\partial y^2} = \frac{\sigma_{(i+1,j)} + \sigma_{(i-1,j)} - 2\sigma_{(i,j)}}{\triangle^2} + \frac{\sigma_{(i,j+1)} + \sigma_{(i,j-1)} - 2\sigma_{(i,j)}}{\triangle^2}. \tag{5.22}$$

where $\triangle$ is the mesh size (a constant).

After all the TSV locations have been determined, we perform a simple legalization step based on an algorithm in [77]. Generally, all cells (gates and TSVs) are sorted by their x coordinates and then packed sequentially into a row to minimize displacement. Although this is a simple legalization algorithm, we find it effective in removing cell overlaps.

Results in Sec.5.11.1 shows that the TSV re-placement engine effectively reduces average TSV EM rate and average von Mises stress value by globally redistributing all the TSVs, with little overhead in total wire length.

5.9 Temperature Estimation Engine

After the TSV locations are perturbed by the TSV re-placement engine, we try to estimate the "best" temperature at these TSVs so that the average EM and fracture rate are minimized. This thermal profile would be provided as a matching objective to the subsequent gate re-placement engine.

We rewrite the reliability part of the objective function (Equation 5.18) and denote its value as $F(\mathbf{T})$. Basically $F(\mathbf{T})$, as shown in Equation 5.23a, is determined by three components: $G(i)$ in Equation 5.24a, $H(i)$ in Equation 5.24b, and $L(i)$ in Equation 5.24c.

All $G(i)$, $H(i)$ and $L(i)$ are functions of temperature. On the one hand, $G(i)$ is an exponentially increasing function of temperature. On the other, $H(i)$ and $L(i)$ are linearly decreasing functions of temperature. In other words, decreasing the i^{th} TSV's temperature will decrease $G(i)$, at the cost of increasing its $H(i)$ and $L(i)$. The increasing $H(i)$ and $L(i)$ accelerate material fracture and other TSV's EM.

Thus the problem of minimizing $F(\mathbf{T})$ in Equation 5.23a is a global optimization problem, where every TSV's temperature affects not only its own reliability, but also all other TSVs' and substrate's reliability as well.

$$\begin{aligned} F(\mathbf{T}) &= \sum_{i \in TSV} \frac{1}{\mathbf{T_i} \cdot exp(\frac{E_a}{k\mathbf{T_i}})} \cdot \left(\frac{\partial^2 \sigma_i}{\partial x^2} + \frac{\partial^2 \sigma_i}{\partial y^2} \right) \\ &+ \frac{3\gamma_f}{\beta_{EM}} \sum_{i \in BB} \left[(\sigma_{xx}^i)^2 + (\sigma_{xy}^i)^2 \right] \\ &= \sum_{i \in TSV} G(i) \cdot H(i) + \frac{3\gamma_f}{\beta_{EM}} \sum_{i \in BB} L(i) \end{aligned} \tag{5.23a}$$

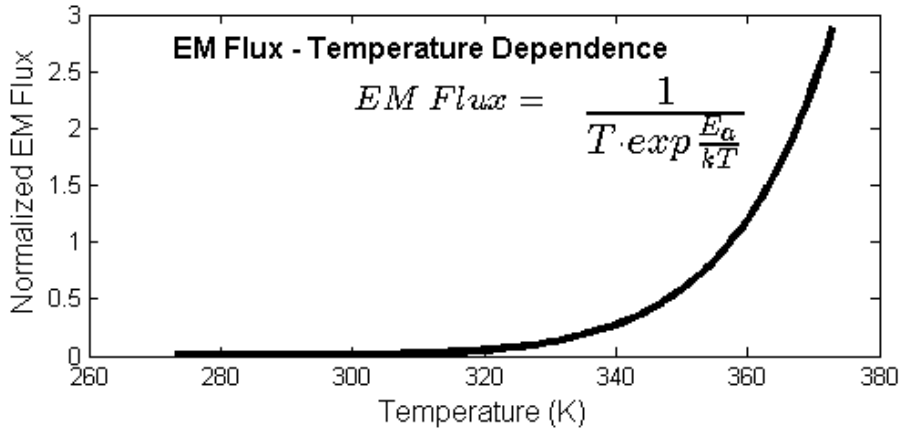
$$G(i) = \frac{1}{\mathbf{T_i} \cdot exp(\frac{E_a}{k\mathbf{T_i}})} \tag{5.24a}$$

$$\begin{aligned} H(i) &= \left(\frac{\partial^2 \sigma_i}{\partial x^2} + \frac{\partial^2 \sigma_i}{\partial y^2} \right) \\ &= \sum_{\substack{j \in TSV \\ j \neq i}} c_0 (T_a - \mathbf{T_j}) \cdot \nabla^2 (\frac{1}{r_{i,j}^2}) \end{aligned} \tag{5.24b}$$

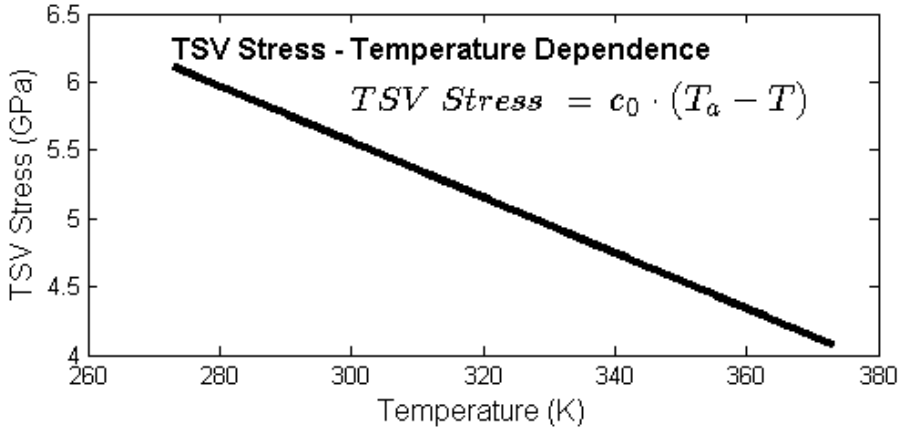
$$\begin{aligned} L(i) &= (\sum_{j \in TSV} \frac{c_0}{r_{i,j}^2} [(T_a - \mathbf{T_j}) cos 2\theta_{ij}])^2 \\ &+ (\sum_{j \in TSV} \frac{c_0}{r_{i,j}^2} [(T_a - \mathbf{T_j}) sin 2\theta_{ij}])^2 \end{aligned} \tag{5.24c}$$

where n is the total number of TSVs, $c_0 = \frac{-E(\alpha_{Cu} - \alpha_{Si})}{2(1-\nu_{Cu})} R^2$, $\nabla^2(\frac{1}{r^2}) = \frac{\partial^2(\frac{1}{r^2})}{\partial x^2} + \frac{\partial^2(\frac{1}{r^2})}{\partial y^2}$, T_a is the annealing temperature, $r_{i,j}$ is the distance from the measured point (j) to the center of the i^{th} TSV, and c_0, $\nabla^2(\frac{1}{r^2})$, and T_a are all constant scalars in this section.

$G(i)$, $H(i)$, and $L(i)$ are all functions of temperature, and their temperature dependencies are plotted in Figure 5.12.



(a)



(b)

Figure 5.12: TSV's EM flux (a) increases when heated, while the stress value (b) drops.

As shown in Figure 5.12(a), $G(i)$, the TSV's EM flux, is an exponentially increasing function of temperature. However, $H(i)$, the TSV's stress, shown in Figure 5.12(b), is a linearly decreasing function of temperature. In other words, decreasing the i^{th} TSV's temperature will decrease the i^{th} TSV's EM, at the cost of increasing all other TSVs' EM due to increasing stress value contributed by the

i^{th} TSV to other TSVs. Thus the problem of minimizing $F(\mathbf{T})$ in Equation 5.23a is a global optimization problem, where every TSV's temperature affects not only its own EM, but also all other TSVs' EM as well.

As indicated earlier, our objective to find an optimal allocation of TSV temperatures such that the TSV's average EM is minimized. However, unfortunately Equation 5.23a is not a convex function w.r.t. all $\mathbf{T}$. Therefore considering the characteristics of $G(i)$ and $H(i)$, we develop the following heuristic, as shown in Algorithm 5.

Input:
1.Fixed placement for logic gates and TSVs;
2.Initial temperature ($T^{(0)}$) and temperature range for each TSV.
Output:
Desired temperature allocation for all TSVs.
Objective:
Minimize TSV's average EM (Equation 5.23a).
0.k = 0;
1.**Repeat**
2. $T_i^{k+1} = T_i^k - \delta T \cdot [\nabla G(i)] \cdot H(i)$;
3. $T_i^{k+1} = T_i^{k+1} - \delta T \cdot \sum_{j \in TSV, j \neq i} G(j) \nabla H(j)$;
4. Check each T_i is within the given temperature range;
5. k = k +1.
6.**Until** EM objective met.

Algorithm 5: Heuristic for temperature estimation

Algorithm 5's input includes (1) the placement for logic gates from either conventional 3D placer during step zero, or the logic cell's re-placement engine that we'll introduce in Sec.5.10, (2) the TSV placement generated by the TSV placement engine, and (3) initial TSV's temperature allocation obtained by applying the compact thermal model (Sec.5.2.6) or temperature range from feedback of the logic cell's re-placement engine. Algorithm refalg:temperature iteratively updates the temperature for all TSVs by moving in a direction that helps in the highest

reduction in $G(i)$ and $H(i)$. The update's step size is denoted as δT. After each temperature update, we check if all current temperature values are within the given temperature range. Then iteration proceeds until a desired TSV's EM objective is reached.

5.10 Logic Cell Replacement Engine

The primary purpose of the logic cell replacement engine is to perturb the current logic gate's placement solution such that the TSV's temperature is as close as possible to the desired temperature allocation generated from Algorithm 5, while minimizing the increase in wire length. The logic cell's re-placement engine is based on multi-level simulated annealing (SA), as shown in Algorithm 6 [78].

Input:
1.Fixed placement for TSVs, and initial placement for logic gates;
2.Power consumption of each logic gate;
3.Desired temperature allocation for TSVs, $T = [T_1^d, \ldots, T_n^d]'$.
Output:
Placement of logic cells.
Begin
1. Setup initial mesh.
2.**Repeat**
3. Calculate the objective grid temperature, T^g by averaging the desired TSV temperature inside each grid.
4. **Begin SA**
5. Move: exchange logic gates between two mesh grids.
6. Update TSV temperature, $T^{new} = T^{cur} + \Delta T$.
7. Objective: $WL + \eta[(T^{new} - T^g)'(T^{new} - T^g)]$.
8. **End SA**.
9. Update mesh size.
10.**Until** Each mesh grid contains only one TSV.

Algorithm 6: Heuristic for SA based cell's placement

The SA begins with a relatively coarse mesh, and iteratively shrinks the mesh size until only one TSV is contained inside each mesh grid. For a particular mesh, we assume the temperature distribution inside each mesh grid is uniform. The desired temperature for each mesh grid is set to be the average temperature of all the residing TSV. During the SA process, logic gates are exchanged between two random mesh grids. Then 3D temperature distribution is incrementally updated, as shown in Equation 5.25.

$$\Delta T = R \cdot [0, \ldots, 0, \delta P_i, 0, \ldots, 0, \delta P_j, 0, \ldots, 0]'. \tag{5.25}$$

where R is the thermal transfer matrix, which is constant since we fixed all TSVs' locations, $\delta P_i, \delta P_j$ are the power changes of grid i and j after logic gates in grid i and j are exchanged during the SA process. The temperature update is incremental thus can be finished in constant time.

The SA's objective function is the weighted sum of total wire length and the mean square error (MSE) between the current and the desired TSV temperature. Again, cell legalization is performed after cell replacement. Finally, the logic cell's replacement engine feeds its output back to temperature estimation engine if it fails to converge and corresponding temperature range is modified in the temperature estimation engine in the hope that the it will produce a more reasonable temperature allocation next time.

5.11 Experimental Results

In this section, we implement our algorithms in Matlab and C++, by an Intel Core i5 3.1GHz CPU and 12GB RAM. We use the IWLS 2005 benchmarks [79], for the purpose of a fair comparison with conventional EM-unaware 3D placer such as 3Dcraft. Detailed statistics of the benchmarks used in our simulation in presented in Table 5.2. The circuit is synthesized with a standard cell library for the MIT Lincoln Lab 130nm 3D SOI technology. The diameter of the TSV is $8\mu m$. The power dissipation of each cell is randomly generated. And for the compact thermal model, conventional air cooling is applied at the top of the 3D-IC while all other sides are adiabatic.

Table 5.2: Benchmark Statistics

Circuit	#Layer	#Gate	#TSV	Power(W)	Area(mm^2)
usb_phy	4	546	110	0.1580	0.053
simple_spi	4	821	146	0.0592	0.077
des_area	4	3132	1071	1.7015	0.318
spi	4	3227	813	1.4683	0.265
usb_funct	4	12808	2128	6.4070	1.076
aes_core	4	20795	4435	6.9717	1.396
ethernet	4	46771	12482	36.9108	4.420
des_perf	4	98341	13112	42.3551	6.871

We use the TSV weight, α_{TSV} as its default value (8000) in 3D-Craft for all benchmarks. This produces 13112 TSVs for 98341 gates (TSV:gate = 13:100) in the largest"des_perf" benchmark. This is comparable to several papers in the literature,

for example, [80] uses 5352 TSVs for 50K gates (TSV:gate = 11:100), and [67] uses 1.63K TSVs for 11K gates (TSV:gate = 15:100). The weights for TSV's EM and material fracture, β_{EM} and γ_f, are experimentally determined. For example, for benchmark "des_perf", we find $\beta_{EM} = 500$ and $\gamma_f = 0.01$ produces a good balance between reliability and total wire length.

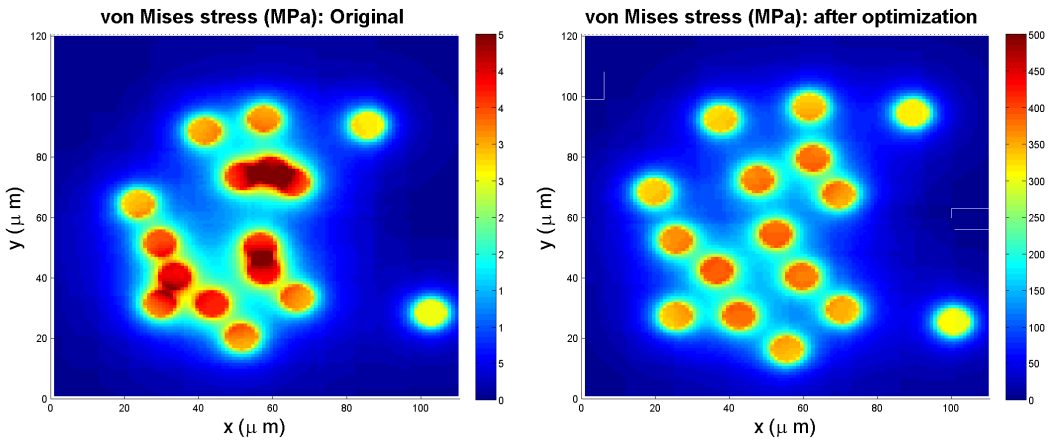
5.11.1 TSV re-placement engine's result

In this section, we compare the wire length and TSV's average EM rate (represented as MTTF value using Equation 5.16) and the von Mises stress between the initial placement result and our TSV re-placement engine. As mentioned earlier in Sec.5.8, the TSV re-placement engine re-distributes all TSVs based on a given initial gate and TSV placement by solving an convex QP problem (Equation 5.21). The bounding box is set to be $10\mu m \times 10\mu m$, by which we can obtain good reliability improvement within reasonable run time. A larger bounding box size results in longer time, but produces limited improvement in reliability metrics. This is because the magnitude of TSV-induced stress is inversely proportional to the square of distance (Equation 5.5), thus a TSV has a small stress impact on a faraway location. We relax the integer constraint in Equation 5.21 to $0 \leq x_{k,a,b} \leq 1$ and solve the resulting continuous QP problem for shorter computational time and then round the solutions. Each simulation for one benchmark is able to finish within one hour. The resulting wire length (in μm) and normalized TSV's average MTTF are presented in Table 5.3.

Table 5.3: TSV re-placement result

Circuit	Initial Placement			TSV Re-placement		
	WL ($\times 10^3$)	von Mises (MPa)	MTTF (norm.)	WL ($\times 10^3$)	von Mises (MPa)	MTTF (norm.)
usb_phy	10.7	485	7.22	10.7	441	10.4
simple_spi	20.0	432	1.32	20.0	395	1.82
aes_area	184.2	556	1.92	184.2	514	2.34
spi	117.2	523	2.22	117.4	481	2.76
usb_funct	506.8	486	1.54	507.3	444	1.97
aes_core	910.9	558	2.42	911.0	519	2.86
ethernet	4298.0	495	1.00	4299.0	447	1.38
des_perf	4148.0	468	6.11	4148.7	431	8.05
Average	1.0**x**	1.0**x**	1.0**x**	1.0**x**	0.92**x**	1.26**x**

As shown in Table 5.3, the TSV re-placement engine is able to achieve 26% longer TSV lifetime on average (1.26**x**), 8% reduction in von Mises stress (1.0**x** - 0.92**x**) with negligible wire length (0.1%) overhead. The gate placement at this stage is unchanged, therefore each TSV has a fixed temperature value. The 0.1% wire length overhead comes from the local change in TSV's locations in order to minimize the TSV's mutual influence on each other's EM rate and thermal stress. Figure 5.13 shows a 3D IC's von Mises stress distribution before and after the TSV replacement. The TSV-replacement engine spreads out the TSVs in order to minimize the average thermal stress. The average TSV displacement is 5.6μm, for $10\mu m \times 10\mu m$ bounding boxes.



(a) (b)

Figure 5.13: (a) von Mises stress distribution before optimization. (b) after optimization.

5.11.2 TSV and gate re-placement's result

A more aggressive approach in addition to the TSV re-placement is to re-place all the TSVs and logic gates such that a desired TSV distribution as well as a desired thermal profile are obtained, while keeping the wire length overhead small. The resulting wire length, peak temperature, and TSV's average MTTF are compared to initial placement, as summarized in Table 5.4.

As Table 5.4 indicates, compared to the reliability-unaware initial placement, our post-placement gate & TSV re-placement enhances the average MTTF (EM caused) by 2.44**x**, and reduces the average von Mises stress by 23% (1.0**x** - 0.77**x**) with 1% wire length overhead.

Table 5.4: TSV & gate re-placement result

Circuit	Initial Placement			TSV & Gate re-placement		
	WL ($\times 10^3$)	von Mises (MPa)	MTTF (norm.)	WL ($\times 10^3$)	von Mises (MPa)	MTTF (norm.)
usb_phy	10.7	485	7.22	11.2	360	19.9
simple_spi	20.0	432	1.32	20.9	329	4.59
aes_area	184.2	556	1.92	186.9	434	4.01
spi	117.2	523	2.22	119.9	406	4.68
usb_funct	506.8	486	1.54	518.4	365	3.26
aes_core	910.9	558	2.42	928.4	440	5.26
ethernet	4298.0	495	1.00	4342.6	364	2.63
des_perf	4148.0	468	6.11	4185.8	392	13.15
Ratio	1.0**x**	1.0 **x**	1.0**x**	1.03**x**	0.77**x**	2.44**x**

5.12 Summary

This chapter presents a TSV and logic gate's post-placement redistribution method to minimize the material fracture and EM for 3D ICs. A unified objective function to model both the material fracture and TSV SM is successfully integrated into existing 3D placement tool. Two studies are performed: (1) locally changing the TSV's locations decouples the TSV-induced stress field thus improves 3D IC's reliability. (2) re-placing both gates and TSVs to achieve a desirable TSV distribution and thermal profile is a more effective approach, as material fracture and TSV SM are both highly temperature dependent. Simulation results show the proposed

TSV & gate re-placement approach achieves 2.44**x** improvement in SM MTTF, 23% reduction in von Mises stress, with only 1% wire length overhead, compared to a reliability unaware initial 3D placement.

Chapter 6: Voltage Noise Induced DRAM Soft Error Reduction Technique for 3D-CPUs

Three-dimensional integration enables stacking DRAM on top of CPU, providing high bandwidth and short latency. However, non-uniform voltage fluctuation and local thermal hotspot in CPU layers are coupled into DRAM layers, causing a non-uniform bit-cell leakage (thereby bit flip) distribution. We propose a performance-power-resilience simulation framework to capture DRAM soft error in 3D multi-core CPU systems. A dynamic resilience management (DRM) scheme is investigated, which adaptively tunes CPU's operating points to adjust DRAM's voltage noise and thermal condition during runtime. The DRM uses dynamic frequency scaling to achieve a resilience borrow-in strategy, which effectively enhances DRAM's resilience without sacrificing performance.

6.1 Introduction

Three-dimensional integration of stacked memory and CPUs has received many attentions recently for its potential to overcome the "Memory Wall" problem - the memory access time (in CPU cycles) has increased to an extent that the memory access latency has become the bottleneck. Vertical stacking enables heterogeneous

integration of multiple DRAM chips on top of a microprocessor, and the high-speed and wider memory bus interfaces (i.e. through-silicon-vias) between the two significantly reduce memory latency.

However, several recent publications have shown that transient fault is one of the common forms of DRAM failures in modern computing and storage systems. One important cause of DRAM transient faults is the power-delivery-network (PDN) noise on DRAM wordline (WL). A noisy WL makes DRAM transistor's gate unable to shut off, causing significant sub-threshold leakage from bit cell's capacitor. As volatile memory, a DRAM bit cell cannot retain its data permanently as the bit capacitor gradually loses its charge. If significant leakage occurs such that DRAM sense amplifier can no longer read the correct data as written last time, a DRAM transient fault happens.

The WL noise induced DRAM transient fault is a severe problem in 3D-CPU systems for the ever-increasing PDN noise and 3D integration's thermal issue. From PDN noise perspective, as more and more devices are integrated and stacked, the power consumption increases volumetrically. The increase of power combined with decrease of supply voltage with technology scaling result in a huge increase in current need. This causes even higher current demand per unit area, because power pins can only be placed at the bottom surface of 3D CPUs. As interconnect resistance increases with technology scaling (narrower planar wires and extra resistance of TSVs), the resistive voltage drop (IR drop) becomes more significant. As CPU frequency increases, the inductive transient voltage droop (Ldi/dt noise) also increases. From thermal perspective, the stacking structure lacks heat conduction

path to reach the ambient. A large amount of heat generated by the ever-increasing CPU power, is often coupled into the DRAM layers, affecting the leakage rate of DRAM transistors.

The architectural solutions for mitigating stacked DRAM's transient fault seem non-trivial. An intuitive idea is to enable shorter DRAM refresh period. However, refreshing operation blocks DRAM access, therefore degrades the performance and energy efficiency [81, 82]. Another approach is to throttle CPU performance to cool the chip or/and reduce PDN noise, which again hurts the performance. In our opinion, a promising opportunity lies at the moment when both low and high power tasks enter the system, and designers can exploit the reliability margin of these tasks and allocate different operating points to maximize the performance while achieving certain long-term resilience. A "borrow-in" strategy can be applied, and high-power tasks might borrow resilience margins to boost performance, and the resilience "loan" will be paid off during a low-power task period.

To the best of our knowledge, although there have been many field studies about the transient DRAM fault recently [83–85], simulation frameworks that investigate its underlying causes, its implications in 3D CPUs, and corresponding mitigation techniques have been lacking. Several publications use fault-injection technique based on constant DRAM failure rate to investigate DRAM resilience [86, 87], however, the fact that different CPU workload causes a temporal and spacial DRAM fault rate variation has been ignored. In 3D CPU architectures, CPU activity causes fluctuation in PDN voltage and temperature, both of which gets coupled into DRAM layers. The integration of multi-dimensional inter-dependent simulation

analysis (performance, power, thermal and voltage) is a challenging task. A more fundamental dilemma for simulation setup is the gap between CPU cycle-level performance / power simulation and long-term DRAM resilience simulation. Although CPU performance / power simulation is crucial in determining the thermal and PDN transients, it's not practical to perform cycle accurate simulation across the entire DRAM lifetime (often in years). This gap needs to be filled for accurate DRAM resilience analysis within practical run-time.

In this chapter, we focus on the WL voltage noise induced stacked DRAM transient fault problem. Although there are other transient fault causes, such as particle-induced upset, they are heavily investigated in the literature, and their trends seem irrelevant with 3D integration. We propose a performance - power - voltage - resilience simulation framework, which effectively captures the interaction between multi-core CPU layer's activity, and voltage noise induced DRAM transient fault. We observe significant correlation between CPU core's activity and DRAM voltage and thermal profiles, both in time and space. As the DRAM leakage strongly depends on voltage noise and temperature, we further observe significant temporal and spacial variation in bit-cell's leakage rate. These two observations imply that the DRAM failure probability strongly depends on 3D CPU's operating points. In our simulation experiments, we investigate an off-line task assignment problem, and propose a two-stage DRAM resilience management approach, which enables frequency allocation for CPU cores and refresh rate adjustment for DRAMs, in order to optimize 3D-CPU's performance while accommodating DRAM resilience requirement.

Specifically our contributions are summarized as follows:

1. Contrary to common assumptions that DRAM transient error is random in time and space, we demonstrate that stacked DRAM's transient error is strongly correlated with 3D CPU activities, through voltage and thermal coupling.

2. Using SPICE simulation, we show that DRAM bit cell leakage is an accumulative effect, indicating it's more related to IR drop rather than transient droop.

3. We propose an effective off-line operating point tuning approach to solve a task assignment problem, which allocates frequency and refresh rate for CPU cores and DRAM controllers, such that the system's performance is maximized, while ensuring DRAM resilience. Compared to a simplistic frequency throttling method, significant performance enhancement is achieved.

6.2 Related Work

In this section, we provide a brief overview of recent work on DRAM transient error field experiment and simulation, and we also cover prevailing management methods.

6.2.1 Background: DRAM Transient Fault

Transient fault in DRAM is a very common form of hardware failure in modern computing clusters and data centers. DRAM's transient fault can be caused by voltage noise and external particles (*i.e.* alpha particles and neutrons). Voltage

noise on the WL causes transistor sub-threshold current and loss of charges on the bit capacitor. The sources of external particles that include alpha particles emitted by packaging materials, and neutrons from cosmic radiation [88].

Recently, several publications from major data servers have shown field experimental studies for detecting and measuring DRAM transient fault. Google [84], AMD [85], and Facebook [83] have reported significant transient error rate during years-long studies. Schroeder *et al.* from Google collected its server fleet's failure record for nearly 2.5 years, and observed a high annual incidence of uncorrectable errors even with strong error correction code (ECC) such as ChipKill [84]. Sridharan *et al.* from AMD also performed field study based on two supercomputers and suggested that transient DRAM errors will become a major concern in the future [85, 89]. Meza *et al.* investigated Facebook's server fleet over fourteen months, and observed an increasing trend of DRAM failure rate with new DRAM fabrication generations [83]. The authors also suggested that the DRAM failure rate is correlated with DRAM PDN noise, and DRAM access patterns. El-Sayed *etc.* published a study on the temperature effect on the DRAM failure rate in data center environment [90].

Statistical modeling approaches such as Monte Carlo (MC) methods are adopted to analyze the probability of DRAM failure under different resilience schemes [86, 87]. The lifetime of DRAM into equal-sized time intervals and transient faults are injected into memory array based on their probability of occurrence. Error detection and correction are invoked periodically to determine whether the injected faults are correctable or not.

In 3D-CPU architectures, power is delivered from external power sources to CPU and DRAM layers, therefore, CPU activity has a strong impact on DRAM layer's voltage fluctuation. An active CPU core draws more current from PDN, causing higher IR drop (and possibly more Ldi/dt voltage droop) in the DRAM layer. We are especially interested in the voltage noise on DRAM WL, as it causes sub-threshold leakage over time. In summary, this chapter focuses on the WL voltage noise induced DRAM transient faults.

6.2.2 Existing Mitigation Techniques

A variety of architectural techniques exist to mitigate the DRAM transient fault problem. Process solutions such as developing purer materials, using better particle strike shielding, employing silicon-on-insulator devices exist, but in this chapter we focus on effective architectural methods. The most effective and prevailing method is hardware-based parity or ECC [91]. Parity is typically an XOR of all data bits and is able to detect any single-bit error. As a more powerful scheme, ECCs such as ChipKill [92] and BCH [93] enable detection and correction of one or multiple bit errors. Hardware-based ECC usually has certain hardware overhead, for example, BCH has an overhead of 8 bits per 64 bits of data.

Reducing the refreshing time interval is an intuitive method to mitigate DRAM's transient fault. However, memory refresh wastes energy and degrades performance by interfering with memory access [81, 82]. The energy and performance overheads of DRAM refresh increase as DRAM device capacity expands.

Since DRAM transient faults can accumulate over time, another option is to perform memory scrubbing that scans throughout the memory and then write back the corrected bit information before single-bit fault mounts to multi-bit uncorrectable fault [94]. However, such a heavy task also incurs significant performance and energy overhead.

Software-based technique includes completely mirroring CPUs or threads. Identical copies of the same program are run simultaneously and the operating system can determine the correct data based on majority voting [95–97].

Another possibility is to change the operating point for dynamic reliability management. Mercati *etc.* proposed a sensor based CPU reliability control scheme, where voltage and frequency operating points are dynamically allocated to improve the performance while meeting the long-time CPU reliability constraint [98]. Shevgoor *etc.* proposed an IR-drop-aware memory controller and a page migration method to cope with IR-drop constraint [99]. While these approaches are effective in handling CPU core reliability degradation, their implications on DRAM resilience are unclear.

6.3 Simulation Framework

In order to analyze the correlation between 3D CPU activity and stacked DRAM resilience, an integrated analysis of how CPU's performance and power transients influences DRAM layer's thermal and voltage behavior is of vital importance. One also needs to quantify how WL noise induces bit flip. Our performance / power / voltage / resilience simulation and optimization flow is summarized in Figure 6.1.

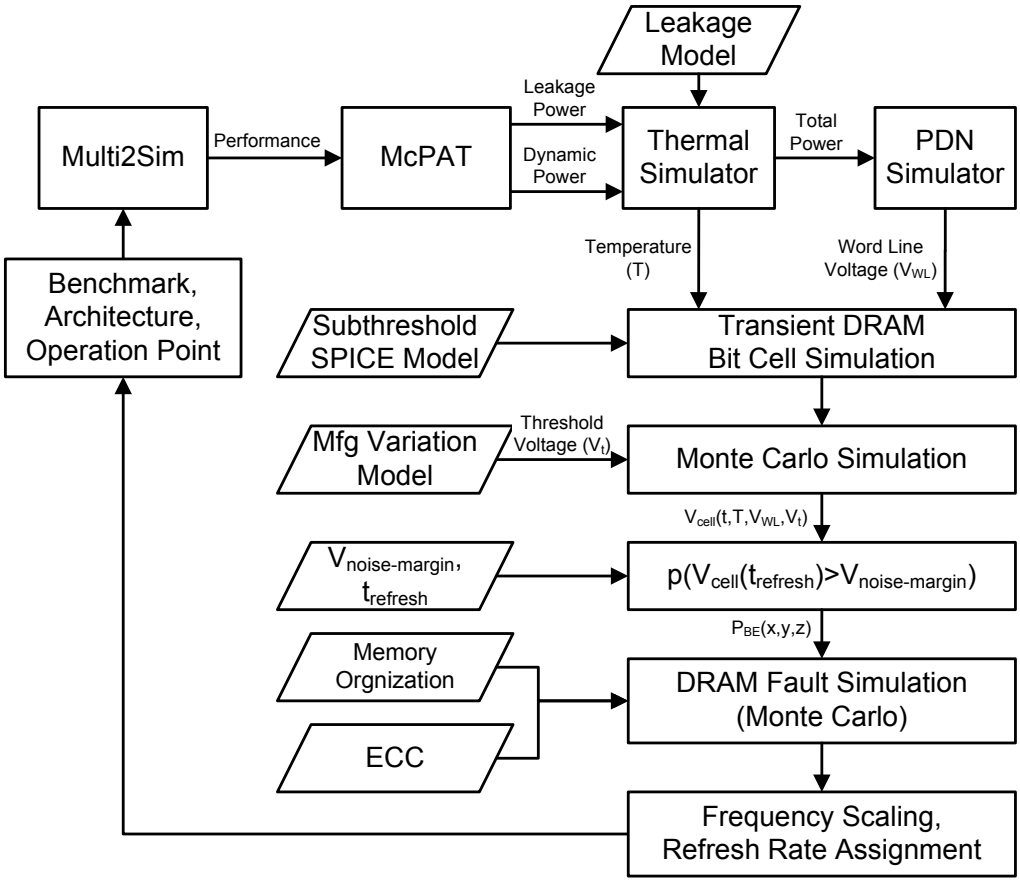


Figure 6.1: Overview of Performance / Power / Voltage / Resilience Simulation and Optimization flow for 3D-CPU.

Multi2Sim is used to produce performance data (instructions executed per nanosecond) and number of memory access [100]. Dynamic and leakage powers of 3D-CPU are simulated with McPAT [101]. The power data are fed into in-house thermal and PDN simulator to generate transient thermal and WL voltage noise distribution. This is followed by a DRAM bit cell SPICE simulation, which captures the sub-threshold leakage current within one DRAM refresh period. Once the noise margin of DRAM bit information is known, we use MC simulation to obtain the probability of a bit cell error (P_{BE}), for a given random manufacturing variation. The P_{BE} value is a function of three dimensional space, which is the input of a DRAM resilience simulator. The DRAM resilience simulator maps the P_{BE} value into memory space, and a high P_{BE} value at certain (x, y, z) location will have a higher probability of transient error in its corresponding memory address space. ECC scheme such as SECDED is evoked periodically. The DRAM fault simulator returns the probability of uncorrectable errors after thousands of MC iterations. After all the benchmarks and all the operating points have been profiled, a feedback loop is created to adjust the operating points of 3D-CPUs. Specifically we consider CPU core frequency scaling and DRAM refresh rate adjusting as two tuning knobs for optimizing CPU performance and DRAM resilience.

6.3.1 Performance/Power/Thermal/Voltage Simulator

6.3.1.1 Performance Simulation

Performance of a software workload on a target 3D CPU is estimated using Multi2Sim [100], a cycle-accurate multi-core CPU simulator. Architectural parameters of the target system including memory and cache architectures and latencies, operating frequency, network-on-chip (NOC) topology, pipeline width, function unit count *etc.* are given to the simulator and the simulator reports the total run time, number and type of executed instructions, and access counts of the various CPU components. The performance metric considered in this work is the number of instructions executed per nanosecond (IPnS).

6.3.1.2 Power/Area Estimation

Power estimates are calculated using McPAT [101]. The tool considers the architectural parameters of the target system and the component access counts generated by the cycle-accurate performance simulator (Section 6.3.1.1). The architectural parameters are used to determine the energy-per-access and total area of each CPU components. The performance counters determine the number of accesses to each component, thus yielding the total energy consumption and average power consumed during the simulation period. The estimation tool outputs a hierarchical

list of dynamic and leakage power estimates for each CPU component. The power distribution is used to analyze the chip temperature (Section 6.3.1.3) and IR drop (Section 6.3.1.4).

6.3.1.3 Thermal Analysis

A 3D grid thermal resistance model of the target 3D CPU is generated based on dimensions and material properties. The power located in each grid is modeled by a current source. This technique is similar to other proposed thermal simulation techniques such as HotSpot [102]. The generated circuit model is solved to yield a voltage distribution which is representative of the thermal distribution of the target 3D stack. Thermal estimates are used to re-estimate leakage power using the baseline estimates from McPAT (which assumes a uniform user-selected temperature) and a thermal-leakage scaling model extracted from the McPAT source code [103]. This thermal-leakage feedback repeats until the convergence. Final thermal estimates are used to model the DRAM bit cell leakage rates and the associated probability of bit errors (Section 6.3.2).

6.3.1.4 PDN Modeling

Our PDN model is an RLC circuit which includes on-chip and off-chip components. Each on-chip grid is connected to its neighbor through a resistor and a inductor in series, and is also attached to a current load. On-chip decoupling capacitance is distributed uniformly on-chip. Power/ground C4 bumps are modeled

as series of resistors and inductors between on-chip and off-chip networks. Power/ground TSVs establish vertical power delivery. We only model the power rail because the ground rail has symmetric property. Our PDN modeling is similar to [104].

6.3.2 DRAM Bit Flip Probability Modeling

Noise induced DRAM bit cell's failure probability is estimated by MC simulation and a transistor level SPICE simulation. During one DRAM refresh period, WL noise induces sub-threshold leakage current, which causes charge transfer from the bit-line capacitor to the DRAM capacitor. The bit-line is connected to VDD/2 when memory access happens, or is floating when the memory array is idle. Once the change of charges exceeds the noise margin, a bit fault occurs.

We assume DRAM manufacturing process causes random variation in bit-cell transistor's gate oxide thickness. A transistor with thin gate oxide leaks faster, and the probability of bit flip is calculated by the total number of flipped bit divided by the number of MC simulations.

The transistor leakage model, Eqn. (6.1a), is from [105]. The temperature-dependent leakage scaling model, Eqn.(6.1b) is extrapolated from the McPAT [101] source code, as in [103].

$$I(t) = I_0(T(t))e^{(V_{gs}(t)-V_{th})q/nkT(t)}(1 - e^{(-V_{ds}(t)q/kT(t))}) \tag{6.1a}$$

$$I_0(T(t)) = I(T_0) \times (5.121\frac{T(t)^2}{T_0} - 6.013\frac{T(t)}{T_0} + 1.892) \tag{6.1b}$$

$$\frac{dV_{cell}(t)}{dt} = \frac{I(t)}{C_{cell}} \tag{6.1c}$$

where I is DRAM transistor leakage, t is time, T is temperature, V_{gs} is gate-to-source voltage, V_{th} is threshold voltage, q is electron charge, k is Boltzmann constant, V_{ds} is drain-to-source voltage, T_0 is nominal temperature, V_{cell} is bit-cell's voltage, and C_{cell} is cell capacitance.

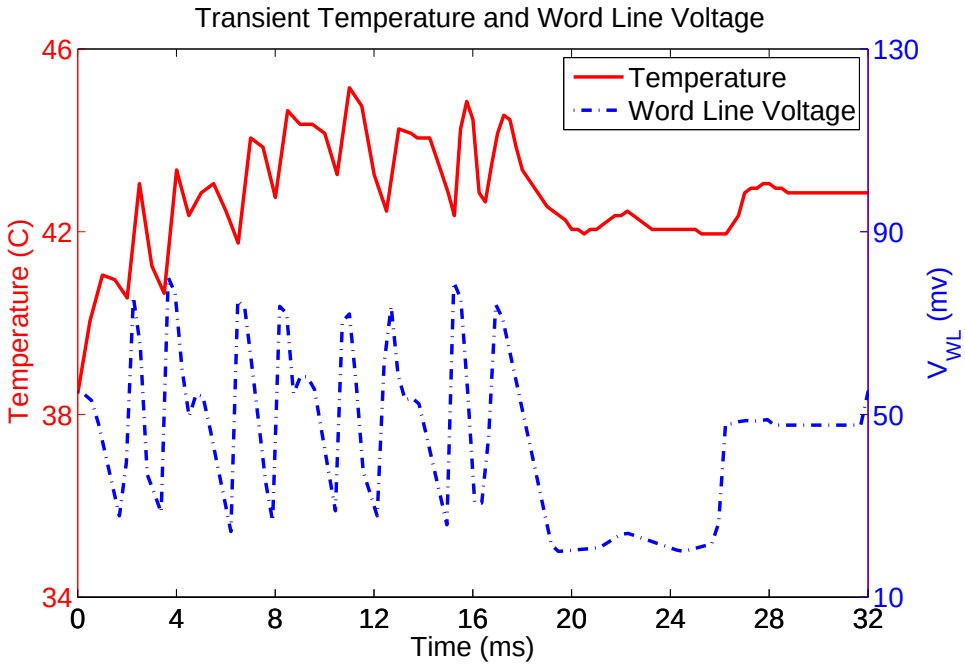


Figure 6.2: Transient WL noise and temperature change in DRAM layer during a 32 ms refresh cycle of "bodytrack" execution.

Figure 6.2 shows the transient of voltage and temperature changes in DRAM layers during the first 32 ms refresh period when executing benchmark "bodytrack". Oscillation in CPU activity is coupled into the DRAM layer, causing thermal and

voltage fluctuations. The transient of bit-cell voltage (with WL noise) is shown with the solid black line in Figure 6.3. The supply voltage is 1 Volt. The transient voltage and temperature data in Figure 6.3 are sampled at 280 μs, and the adaptive time step used by the DEQ solver to calculate bit cell voltage is roughly 10x small than the data sampling rate. For comparison, an ideal WL voltage (perfectly grounded) is applied for the same bit-cell, and the voltage on its capacitor is plotted with the black dotted line. As can be seen in the figure, the original bit information "0" gets corrupted much fast due to WL noise.

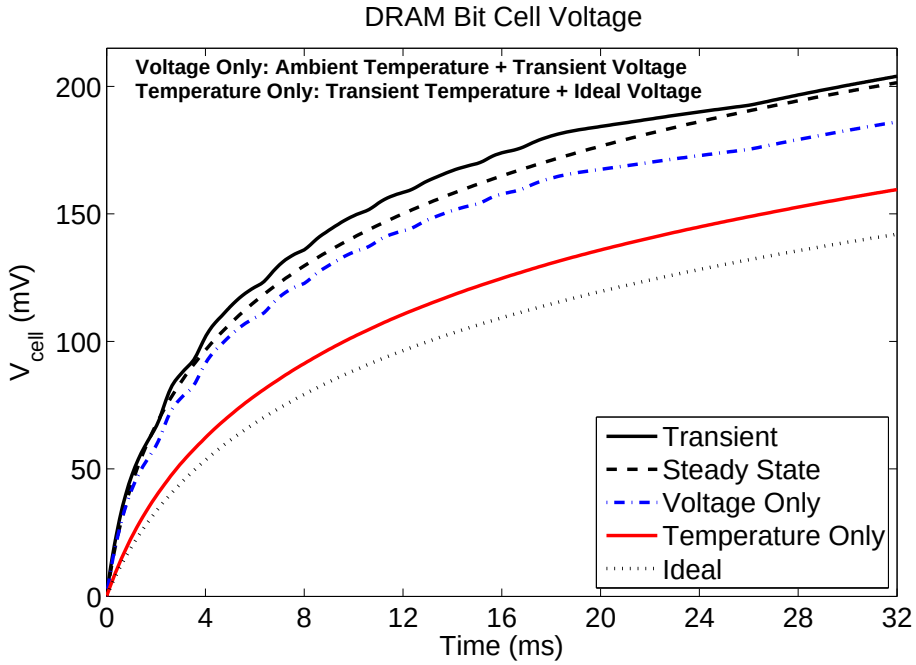


Figure 6.3: DRAM bit transistor's sub-threshold leakage charges the bit capacitance over time. The original bit information "0" gets corrupted when capacitor's voltage exceeds the noise margin.

Another transient is performed with the average voltage noise and temperature as inputs, and the bit-cell's voltage trace is solved similarly. We refer this as "V-T steady state" simulation. The resulting bit cell voltage is plotted with the dashed

black curve. The "V-T steady state" simulation converges with the real transient simulation at the end of the refresh cycle. Note that in Eqn.(6.1), the leakage current is an exponential function of V_{gs}, where $V_{gs} = V_{WL} - V_{Cell}$. However, the reason that "V-T steady state" simulation achieves high accuracy is that within the temperature and voltage noise range (35 °C - 45 °C and around 100 mV WL noise), the bit-cell voltage as function of temperature and WL noise is operating at a linear region. This effect has important implications on operating point tuning techniques such as dynamic frequency and voltage scaling (DVFS). When the time constant of DVFS is in the same order of magnitude with DRAM refresh period (V-F selection in roughly every 50 ms in [106]), taking the average of voltage noise and temperature and then simulate the transient of bit cell voltage will be fast and accurate.

We also perform two sensitivity analyses, one with ambient temperature and transient voltage noise, and another with transient temperature and ideal WL voltage (*i.e.* grounded). The results are plotted with blue dashed line and red solid line, respectively. The data show that when executing "bodytrack", WL noise seems to be a more dominating factor than temperature for sub-threshold leakage.

6.3.3 DRAM Resilience Simulator

The DRAM resilience simulator uses MC method to obtain the probability of DRAM failure under a given ECC scheme. Based on DRAM layer's thermal and voltage noise distribution generated by thermal and PDN simulations for a

particular CPU activity transient, we obtain the physical distribution of P_{BE}. A DRAM block with a thermal hotspot and a noisy WL has higher probability of bit flip. A random number of bit flips per grid is generated according to binomial error distribution. The spacial distribution of bit flip numbers is then mapped into memory address space, for a given memory organization and physical layout. We use the data structure of Faultsim [87] to represent transient faults in memory address space. Finally ECC is evoked and the simulator returns whether the voltage noise induced transient fault causes an uncorrectable DRAM fault. Thousands of MC simulations are performed with random manufacturing variation until a reliable probability of uncorrectable DRAM failure (POF) is obtained.

6.4 DRAM Resilience Management

Using the simulation framework discussed in Section 6.3.1 we are able to obtain the spatial distribution of P_{BE} and the corresponding POF when executing different phases of a set of software workloads. With this profiling data we are able to estimate the optimal core frequency and DRAM refresh rate for each phase of each application using the scheme shown in Figure 6.4.

In Figure 6.4, the optimization goal is to maximize the system performance while meeting a specified DRAM resilience target (*i.e.* the long-term probability of an uncorrectable memory error). The long-term period is divided into several short-term periods. The long term controller (LTC) takes a set of applications awaiting execution from the application queue, and determines at which operating point (*i.e.*

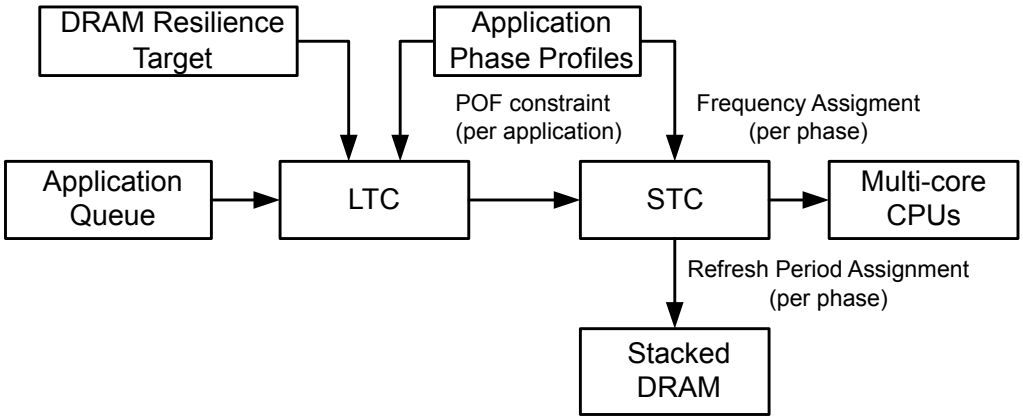


Figure 6.4: Control scheme for frequency and refresh rate assignment across applications (long term control) and application phases (short term control).

frequency/refresh rate setting) each application should be run to minimize runtime while meeting the long term DRAM resilience target R_t averaged across all considered applications. The short term controller (STC) optimizes the runtime of each application by assigning an operating point to each phase of the application. The POF estimate output by LTC serves as a constraint on the POF of the application during STC optimization.

One limitation of our control scheme is it relies on the availability of profile data, which assumes that the workloads can be predicted off-line. While this assumption may not be applicable to all systems, it applies well to embedded systems [107] and high performance computing (HPC) systems [108]. These are two paradigms where the stacked DRAM processor shows great potential due to its small form factor and low power (embedded systems) and/or massive memory bandwidth (HPC).

6.4.1 Long Term Controller

During one long term period, we assume DRAM bit errors do not accumulate from one application to the next. Any errors accumulated in one application are effectively corrected when the application completes and its memory is deallocated. Thus the POF of each application is independent and the DRAM resilience (R) is calculated using Equation (6.2).

$$R = \prod_i (1 - POF_i) \tag{6.2}$$

The DRAM resilience constraint can be linearized using Equation (6.3), allowing the assignment problem to be formulated as an integer linear program (ILP) in Equation (6.4). $\delta_{i,j,k}$ is a binary variable such that it is set when application i is assigned to run at frequency j and DRAM refresh rate k. Likewise $t_{i,j,k}$ and $POF_{i,j,k}$ are the runtime and POF of application i running at frequency j and DRAM refresh rate k.

$$\prod_i (1 - POF_i) \geq R_t \rightarrow \sum_i \ln(1 - POF_i) \geq \ln(R_t) \tag{6.3}$$

Decision variable : $\delta_{i,j,k}$

$$
\begin{aligned}
\max \quad & \sum_{i,j,k} \delta_{i,j,k} \\
s.t. \quad & 1. \quad \sum_{i,j,k} t_{i,j,k} \cdot \delta_{i,j,k} \leq t_{LTC} \\
& 2. \quad \sum_{i,j,k} -\ln(1 - POF_{i,j,k}) \cdot \delta_{i,j,k} \leq -\ln(R_t) \\
& 3. \quad \sum_{j,k} \delta_{i,j,k} \leq 1 \, \forall_i \\
& 4. \quad \sum_{j,k} \delta_{i,j,k} - \sum_{j,k} \delta_{i-1,j,k} \leq 0 \, \forall_i
\end{aligned}
\tag{6.4}
$$

The ILP problem objective maximizes the number of assigned applications and Constraint 1 ensures that the total runtime of all assigned applications is less than the LTC control window t_{LTC}. By maximizing the number of applications assigned in a fixed period of time we maximize performance since applications must be executed in order. Constraints 3 and 4 respectively ensure that each application is assigned no more than one operating point and applications are executed in order. Constraint 2 uses the linearized resilience constraint from Equation (6.3) to generate operating point assignments that meet the resilience constraint across the LTC control window.

The assignment of operating points to each application yields an expected POF and runtime value for each application such that total runtime is minimized and the DRAM resilience target is met. However the assignment of a single operating point to each application is overly constrained. We improve our assignment

solution by allowing different operating points to be assigned to each phase of a single application using the short term controller such that the application's POF is less than the POF estimated by the LTC but the total runtime is reduced.

6.4.2 Short Term Controller

The short term controller (STC) allocates operating points to each phase of one application. The POF estimate output by LTC is a POF constraint during STC optimization. Within a single application DRAM transient fault from phase to phase do compound, which means the POF of an application cannot be described as a linear combination of the POF of each phase. The problem could be solved by expanding the size of the ILP problem to include all possible sequences of operating points, but this would be computationally infeasible for non-trivial problem sizes. Moreover the profiling effort to generate POF values for all combinations of operating points would be unnecessarily detailed. So we instead solve the STC optimization problem using a simulated annealing (SA) heuristic. Since the initial solution given by the LTC (*i.e.* run all phases of one application at the given operating point) is already a good solution, low temperature SA does well to improve the solution.

The annealing function used in our SA formulation moves the operating point of one phase to a neighboring operating point (*i.e.* increase or decrease the frequency or refresh rate by one level). The SA objective function evaluates the runtime and POF of each considered sequence of operating points and evaluates Equation (6.5).

r_i is the runtime of phase i running at the assigned operating point. POF is the POF of the entire application running at the considered sequence of operating points. Evaluation of POF is explained below. POF_{LTC} and RT_{LTC} are the POF and runtime estimate generated by the LTC for this application, assuming all phases were run at a single operating point.

$$OBJ = c_1 \cdot \frac{\sum_i r(i)}{RT_{LTC}} + c_2 \cdot \max(1, \frac{POF}{\sum_i r(i)} / \frac{POF_{LTC}}{LT_{LTC}}) \tag{6.5}$$

The first objective term tries to reduce the total runtime of an application. However if the runtime of an application is reduced the probability of failure must reduce by the same amount to maintain to long term POF constraint. The second term serves to constrain the optimization by assigning a large penalty if the considered ratio between POF and runtime increases above the one assigned by the LTC. Since the second term is a constraint, c_2 is much greater than c_1.

Note that the runtime of each phase at each operating point is already available in our profile data, and we can simply sum up all phases to evaluate the runtime of the entire application.

6.4.3 Fast Conversion from P_{BE} to POF

In Eqn.6.5, the evaluation of the POF (of an entire application) requires simulating the sequence of associated P_{BE} distributions using our DRAM resilience simulator (Section 6.3.3), which takes significant runtime. In order to speed up the optimization we propose a fast conversion method to convert a sequence of P_{BE} distributions into a POF estimation, based on the observation in Figure 6.5.

Figure 6.5 shows a plot of the POF value estimated from our DRAM resilience simulator versus the average P_{BE} over 3D space. As the average P_{BE} increases, more bit flips occur therefore it becomes harder for ECC detection. Each red data point represents running one benchmark at a specific combination of core frequency and DRAM refresh rate. We observe that the data fits an error function very well, and conclude that such the fitted function can be used to convert an arbitrary average P_{BE} over space into a scalar POF value. The fitted error function is shown in Figure 6.5.

The data in Figure 6.5 is derived using the DRAM resilience simulator with a specific P_{BE} distribution. In our SA objective function we must convert a sequence of P_{BE} distributions to the application's POF, since each application consists of multiple phases with varying operating points. We deduce that a sequence of P_{BE} distributions can be converted to an equivalent single distribution by taking the maximum value across all members of the sequence for each point in the spatial distribution. Because the underlying source of randomness (*i.e.* manufacturing variations) is not changing over time, it is in fact the phase with the highest P_{BE}

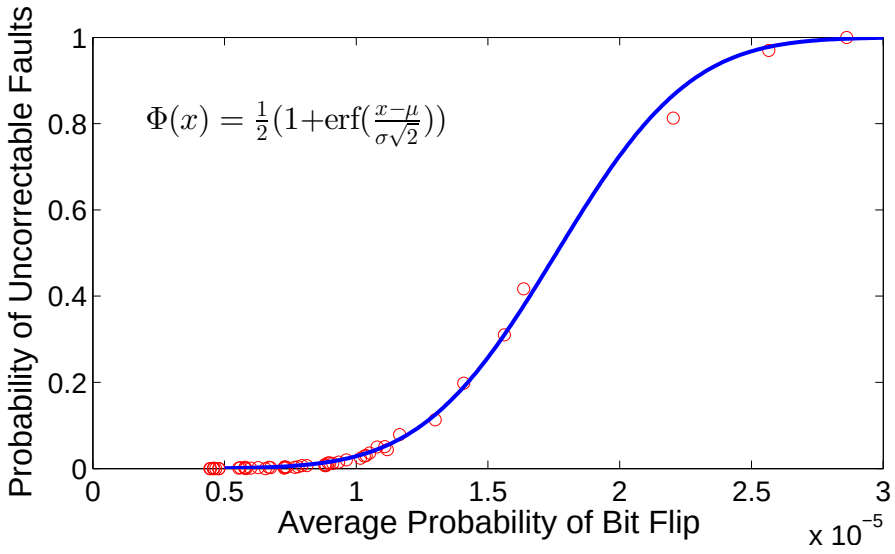


Figure 6.5: Probability of uncorrectable fault (POF) versus probability of bit flip (P_{BE}) averaged over three dimensional space.

that determines if bit flip occurs at that point. If the bit error does not occur when the probability is the highest (*i.e.* when the leakage current is the highest) it is not possible that the bit error would have occurred when the probability was less (*i.e.* when the leakage current was lower). Thus to evaluate an application's POF, we first generate an equivalent P_{BE} distribution by taking the maximum P_{BE} across all members of the sequence at each point in space, and then evaluate POF based on the mapping in Figure 6.5.

6.5 Simulation Results

6.5.1 Experimental Setup

We study a 3D CPU architecture with four layers of DRAM (4GB total capacity) and one layer of CPU. Each DRAM layer represents one rank. One DRAM rank is partitioned into four banks, and within each bank, there are 131072 rows and 16384 columns. Each DRAM rank is connected to the CPU layer with a 512 bit bus implemented with TSVs. The memory organization we consider is similar to previous stacked DRAM publications [106,109]. Distributed refresh mode supported by JEDEC is used. The CPU layer containing 16 cores is stacked above the DRAM layers and adjacent to the heat sink to prevent thermal violations. We mesh the 3D CPU into multiple grids, with the size of $200\mu m \times 200\mu m$ for each grid. As the power estimates provided by McPAT are at the function unit granularity which is on the order of 0.1 mm^2, the $200\mu m \times 200\mu m$ grid size is sufficient to capture the thermal and voltage variance in both the multi-core processor and DRAM layers. BCH [93] is used as ECC.

For PDN modeling, the on-chip decoupling capacitance density is $33nF/cm^2$ [110]. Other parameters of PDN is calculated based on its physical dimension and material property, similarly to the techniques proposed in [104].

For DRAM bit flip probability P_{BE} modeling, we assume a normal distribution of gate oxide thickness. The DRAM capacitance is 30 fF. The DRAM cell's noise margin depends on the strength of the DRAM sense amplifier, and here we assume

that a bit error occurs if the bit cell voltage degrades more than 10% VDD from ideal. For a given DRAM refresh period, a hundred thousand MC simulations are run for each WL voltage and temperature point to obtain accurate P_{BE} distribution. The run time of the total MC simulations is about 12 hours for a PC with an Intel Core i5 3.1GHz CPU and 12GB RAM. We only need to run the MC simulations once to obtain P_{BE} as a function of PDN noise and temperature.

We use part of SPLASH-2 and PARSEC benchmarks in our simulation [111, 112]. These benchmarks are commonly used to profile highly parallel computing systems, and we assume them to be representative of the different computing operations to be encountered in real applications. We profile our proposed control scheme across a sequence of applications, where each application is composed of different computing phases which are represented by the benchmarks. Each application is randomly assigned a nominal runtime (*i.e.* runtime if it were assigned the nominal operating point) and a random distribution of how much runtime is consumed by each type of phase. It is likely that a single application only contains a subset of all possible phases, and even applications containing the same subset can have drastically different distributions of runtime across phases.The specific benchmarks we use are: bodytrack, dedup, fft, fluidanimate, radix and swaptions.

Performance, power, voltage and temperature for each benchmark at each operating point are generated using the simulation framework described in Section 6.3.1. DRAM refresh rate affects memory latency due to less availability of the DRAM. Memory latency is scaled from the nominal value proportional to the increase in DRAM overhead (refresh time divided by refresh period). Although the

DRAM refresh rate determines the DRAM memory latency, it affects performance differently for different benchmarks as a function of their memory usage: memory bound workloads will gain a large performance improvement if memory refresh period is increased whereas CPU bound workloads will see little improvement.

We set the long term control window (t_{LTC}) to 30 minutes and assign application runtime uniformly between 1.33 to 8 seconds. The POF constraint within a long term control window is 0.3%. The controller assigns each phase of each application a frequency value from the set {3.0, 2.5, 2.0, 1.5} GHz and a refresh period from the set {8, 12, 16, 32, 64} ms.

6.5.2 Results

6.5.2.1 Probability of Bit Flip

We use the SPICE simulation setup described in Sec. 6.3.2 to determine the probability of DRAM bit flip (P_{BE}) across the voltage noise, temperature and refresh rate ranges. The resulting P_{BE} at $16\mu s$ refresh period is shown in Fig.6.6. The P_{BE} values are normalized to the POF value at $75\,^oC$ and $100mV$ WL voltage noise.

As Figure 6.6 shows, increasing the WL noise and temperature both increase P_{BE} exponentially, where the WL noise has a more dominating effect.

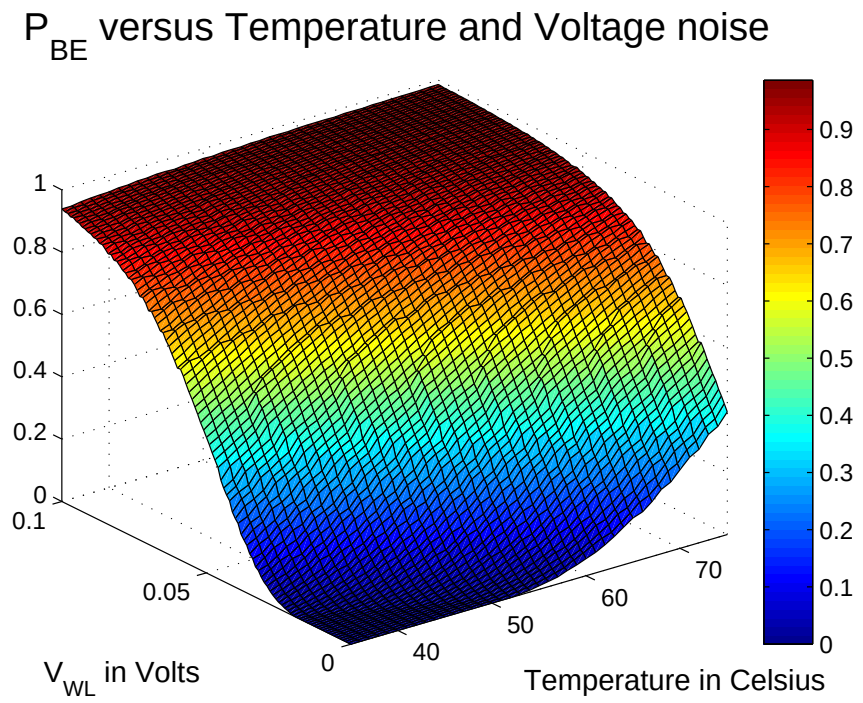


Figure 6.6: Probability of bit flip (P_{BE}) versus WL noise level (V_{WL}) and temperature, at 16μs refresh period.

6.5.2.2 DRAM Resilience Management

We analyze the distribution of operating points assigned to our controller across a set of 520 applications. The nominal operating point is 3.0 GHz clock frequency and 8 ms refresh period. The results of this analysis are shown in Table 6.1. Although 20 combinations of frequency and refresh are possible, 13 of them are never assigned. The 32/64 ms refresh rates are never chosen by the scheduler. This is because the DRAM IR drop can be as high as 0.1 VDD (shown in Figure 6.2), which leads to significant increases to DRAM transistor leakage and that is why 32/64 ms are used for 2D ICs but only shorter refresh periods are feasible for 3D ICs.

Figure 6.7 plots the 12 operating points (32/64 ms refresh rates are excluded in this figure) and their corresponding runtime and *POF* for a representative application from the optimized set. It can be seen that four operating points for this application are not on the Pareto optimal front, and thus will never be chosen by the controller. This explains the data in the upper right corner of Table 6.1. In this distribution the nominal operating point is chosen a majority of the time, but significant throttling of both frequency and refresh rate are used to optimally trade off performance and DRAM resiliency. For example, an application with high power (which causes PDN noise and temperature increase) may require frequency to be reduced, whereas a lower power application puts less stress on the DRAM resiliency constraint and can be configured to run with a longer refresh period and get a performance boost.

Table 6.1: Distribution of operating points assigned by our controller

		Refresh Rate				
		8 ms	12 ms	16 ms	32 ms	64 ms
Freq.	1.5 GHz	1.8%	0.0%	0.0%	0.0%	0.0%
	2.0 GHz	1.8%	0.0%	0.0%	0.0%	0.0%
	2.5 GHz	7.2%	0.0%	0.5%	0.0%	0.0%
	3.0 GHz	59.3%	27.5%	2.0%	0.0%	0.0%

We compare the performance subject to *POF* constraint of our control scheme to the baseline case of running at nominal frequency and refresh rate. Our technique is able to improve system throughput (jobs completed per second) by 27%. Our

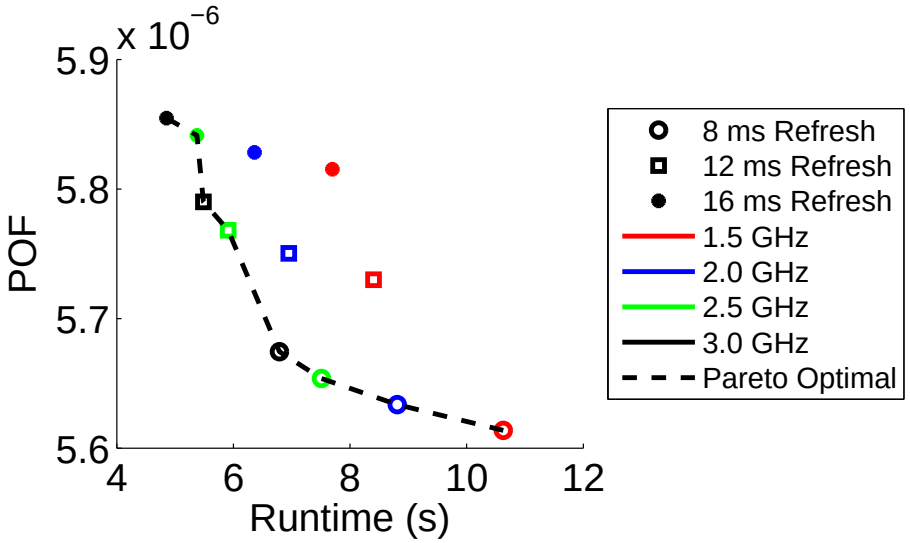


Figure 6.7: Pareto optimal frequency/refresh rate points for a representative application.

controller improves performance in two ways: 1) it allows applications to trade off resiliency and performance with one another and 2) it allows slack in the POF constraint to be converted to performance improvements.

Table 6.2: Throughput improvement using our frequency/refresh rate controller scheme

	Optimal	Nominal
POF	0.30%	0.23%
Throughput (jobs/second)	0.29	0.23
Normalized Throughput	1.27	1.00

6.6 Conclusion

This chapter presents a detailed analysis and optimization scheme for 3D CPUs on voltage noise induced DRAM transient fault. Significant correlations between CPU activities and DRAM layer thermal and voltage noise behaviors have been observed. We show that under certain DRAM resilience target, the currently off-

the-shelf DRAM refresh period (32 or 64 ms) is not sufficient, however arbitrarily applying faster DRAM refresh rate inevitably hurts the performance. We propose a dynamic DRAM resilience management technique, which tunes CPU frequency and DRAM refresh rate in order to maximize performance while meeting long-term resilience target. Simulation results show that our management scheme achieves 27% increase in performance when comparing to running at the nominal operating point.

Chapter 7: Conclusions and Future Work

3D ICs have shown promising improvements in performance and energy efficiency independent of costly transistor scaling. However, the expanded design space brought on by 3D integration imposes extra design complexities to the physical design domain, including the 3D clock tree synthesis. Furthermore, vertical vias introduce new sources of reliability degradations. In this dissertation, we present novel clock tree synthesis flow for 3D ICs. We also develop a simulation framework to capture the trend of EM degradation and thermal mechanical stress. These physical design methodologies are necessary enhance 3D IC's performance, power and reliability, and push 3D ICs into full commercialization in the near future.

In order to deliver clock signal throughout the three dimensional space, we develop a clock tree synthesis algorithm for 3D ICs. Clock gating is applied in order to minimize the clock tree power. Different than clock gating a 2D IC, sending "enable" signal to shutdown gates requires control TSVs, which compete with clock TSVs for placement resources. In contrast with conventional clock tree synthesis flow which constructs the clock tree based only on the clock sinks' geometric information, our flow aims to cluster clock sinks with similar switching behaviors, thus one shutdown gate can control multiple clock sinks at the same time. Besides, our

clock tree synthesis flow accounts for placement whitespaces for TSVs, and is able to optimally allocate clock TSVs and control TSVs such that the overall clock power is minimum.

We also investigate several reliability aware physical design methodologies. The first case study we have performed is about EM-aware delay optimization for a TSV-involved timing path. We develop a fast and accurate meshing strategy to predict the peak current density inside one TSV. The meshing strategy avoids time-consuming numerical simulation. To prevent EM degradation caused by high magnitude of current, wire sizing technique is used to control the peak current density, while optimizing for wire delay. We use dynamic programming method to solve the EM-constrained delay optimization problem, and successfully minimize interconnect delay while meeting the EM constraint. In addition, we recognize the TSV's tapering effect, which is a byproduct of TSV manufacturing, and quantitatively summarize that TSV tapering causes higher magnitude of current in side TSVs.

Furthermore, a more advanced and accurate TSV EM model is investigated. We set up a simulation framework using FEM to investigate the impact of electrical current, thermal stress, and temperature on TSV's EM. Due to the large mismatch of CTEs between copper and silicon, TSV and neighboring substrate area endure high magnitude of thermal stress, which dramatically accelerates the migration of atoms inside TSVs. Besides, high magnitude of thermal stress leads to delamination and cracks of the substrate. We develop accurate analytical models for TSV's EM and TSV-induced material fracture, which replace time-consuming FEM simulations.

These reliability models are then embedded into the 3D placement design flow, and we show that from locally rearranging the locations of TSVs and logic gates, circuit lifetime can be substantially increased, and thermal stress can be reduced, with litter degradation on circuit's performance.

At last, we present a detailed simulation framework for analyzing 3D CPUs' voltage noise induced DRAM transient fault. Significant correlations between CPU activities and DRAM layer thermal and voltage noise behaviors have been observed. We show that under certain DRAM resilience target, the currently off-the-shelf DRAM refresh period (32 or 64 ms) is not sufficient, however arbitrarily applying faster DRAM refresh rate inevitably hurts the performance. Based on our simulation framework, we propose a dynamic DRAM resilience management technique, which tunes CPU frequency and DRAM refresh rate in order to maximize performance while meeting long-term resilience target. Simulation results show that our management scheme achieves higher throughput when comparing to running at the nominal operating point.

7.1 Future Work

In Chapters 3 through Chapters 6 we have discussed our physical design methodologies on 3D clock tree synthesis, wire delay optimization, reliability-aware 3D placement flow, and techniques for voltage noise induced soft error mitigation. We have proposed effective models and algorithms for solving these problems. One of our future thesis works is on the powe delivery design and mangement.

7.1.1 Power Delivery Design and Management

Suppressing voltage noises is the most important PDN design objective to ensure a quality and stable voltage level. Voltage noises are especially severe in 3D ICs. On one hand, the total power consumption increases as more chips are vertically stacked. On the other hand, the power I/O pin count is proportional to 3D IC's footprint area, which is relatively fixed due to manufacturing constraints. As functional units are drawing large amounts of currents from the power supply, significant power is lost during vertical transportation through IR drop. In addition, the surge of current can induce significant Ldi/dt voltage droop.

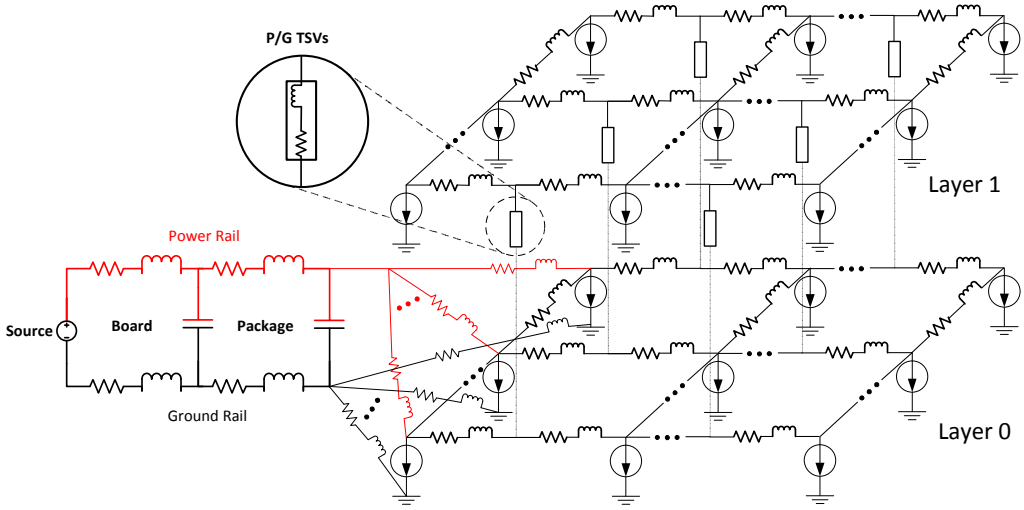
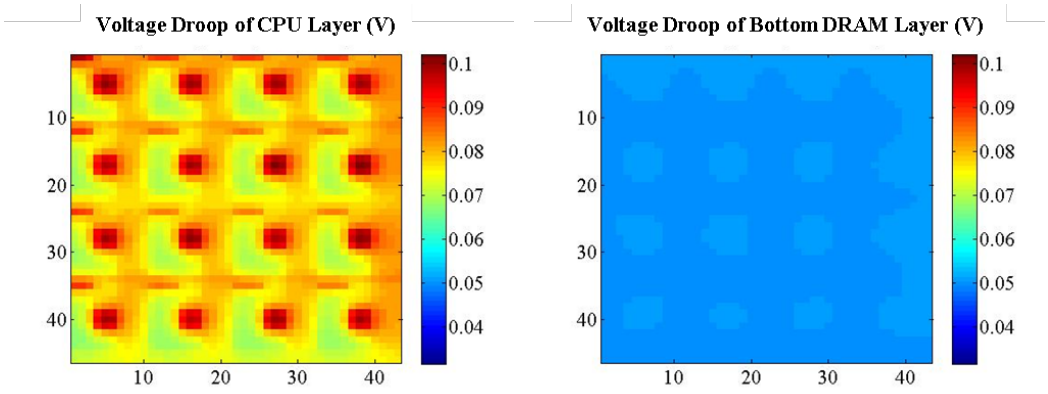


Figure 7.1: A two-layer 3D IC's PDN model

Figure 7.1 shows a two-layer 3D IC's PDN model, which consists of RLC components in the PCB board, package, and the stacking chips. The stacking chips are meshed into multiple grids, where current draws are modeled as current sources attached to each of the grids. Between grids there are RL components for planar wires and power TSVs.

Several properties of 3D power network are essential for designers to suppress the voltage noises in a 3D IC's power line. First, power consumption possesses significant spatial variations in both planar plane and along the vertical direction. Inside the planar plane different functional units dissipate various powers (i.e. an execution unit dissipates more power than a load-and-store unit). Along the vertical direction, where chips with different technologies are bonded, power consumptions might also be different. For example, a DRAM layer requires much lower power than a CPU layer.

Second, the stacking structure makes the power noise to be coupled to adjacent layers. For example, during a period when CPU cores are active, significant voltage noise are observed in adjacent DRAM layers as well. Figure 7.2 shows the static state voltage map for a 3D CPU where 4 DRAM layers and 1 16-core CPU layer are stacked, and power I/O pins are placed next to the DRAM layer. The nominal VDD is 1 Volt.



(a) (b)

Figure 7.2: Voltage map of (a) the processor layer (furthest away from power I/O) and (b) the bottom DRAM layer (closest to power I/O)

In Figure 7.2, the CPU layer has larger voltage droop and also larger voltage droop variation, comparing to the DRAM layer. This is primarily because the CPU layer consumes large amounts of power, therefore large amounts of currents are drawn from the power I/Os and transported to the CPU layer, causing significant IR drop. DRAM layers consume much less power, therefore, the IR drop is much less severe.

Third, power consumptions vary dramatically in time. Heavy computational task results in higher power consumption, and run-time approaches such as dynamic voltage and frequency scaling (DVFS) also impacts the power profile dynamically. The transient voltage droop for the same 3D CPU when simulating the "BodyTrack" parallel benchmark from [111] is shown in Figure 7.3. We observe large temporal voltage droop variation in the CPU layer, and in addition, the voltage droop from the CPU layer is coupled into adjacent DRAM layers.

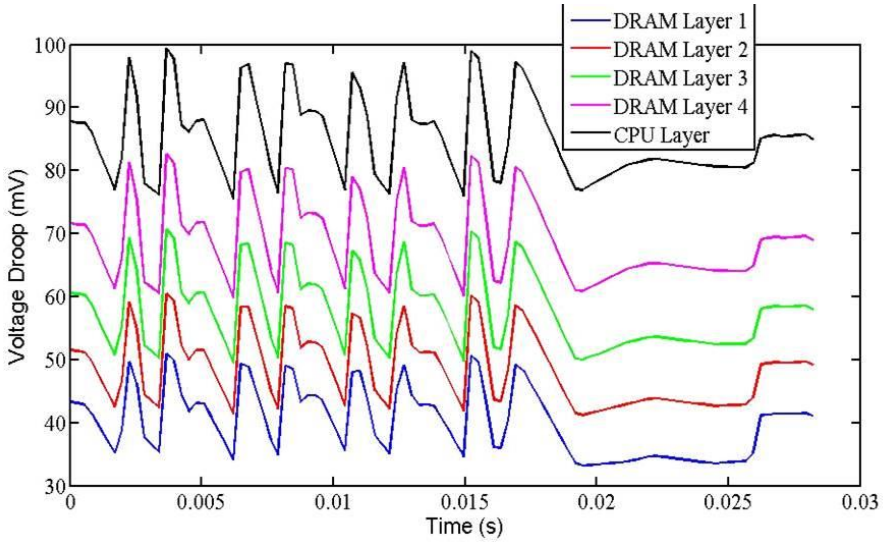


Figure 7.3: The transient voltage droop in CPU and DRAM layers when simulating the "BodyTrack" parallel benchmark

Bibliography

[1] J.S. Pak, M. Pathak, Sung-Kyu Lim, and D.Z. Pan. Modeling of electromigration in through-silicon-via based 3d ic. In *Electronic Components and Technology Conference*, pages 1420–1427, 2011.

[2] Xin Zhao, J. Minz, and Sung Kyu Lim. Low-power and reliable clock network design for through-silicon via (tsv) based 3d ics. *IEEE Trans. Compon. Packag. Manuf. Technol.*, 1(2):247–259, Feb 2011.

[3] Xin Zhao, M. Scheuermann, and Sung Kyu Lim. Analysis of dc current crowding in through-silicon-vias and its impact on power integrity in 3d ics. In *Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE*, pages 157 –162, 2012.

[4] Caleb Serafy, Tiantao Lu, and Ankur Srivastava. Thermal-reliability physical co-optimization during architectural design space exploration of 3d-cpus. In *GOMACTech*, 2016.

[5] James R. Black. Mass transport of aluminum by momentum exchange with conducting electrons. In *Reliability Physics Symposium, 2005. Proceedings. 43rd Annual. 2005 IEEE International*, pages 1–6, 2005.

[6] Kuang-Yao Lee, Cheng-Kok Koh, Ting-Chi Wang, and Kai-Yuan Chao. Fast and optimal redundant via insertion. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 2197–2208, Dec 2008.

[7] K.H. Lu, Suk-Kyu Ryu, Qiu Zhao, Xuefeng Zhang, Jay Im, Rui Huang, and Paul S. Ho. Thermal stress induced delamination of through silicon vias in 3-d interconnects. In *ECTC*, pages 40–45, June 2010.

[8] K.H. Lu, Xuefeng Zhang, Suk-Kyu Ryu, Jay Im, Rui Huang, and Paul S. Ho. Thermo-mechanical reliability of 3-d ics containing through silicon vias. In *ECTC*, pages 630–634, May 2009.

[9] Jing Zhang, Max O. Bloomfield, Jian-Qiang Lu, R.J. Gutmann, and Timothy S. Cale. Modeling thermal stresses in 3-d ic interwafer interconnects. *Semiconductor Manufacturing, IEEE Transactions on*, pages 437–448, Nov 2006.

[10] Xi Liu, Q. Chen, Pradeep Dixit, R. Chatterjee, R.R. Tummala, and S.K. Sitaraman. Failure mechanisms and optimum design for electroplated copper through-silicon vias (tsv). In *ECTC*, pages 624–629, May 2009.

[11] J. Mitra, Moongon Jung, Suk-Kyu Ryu, Rui Huang, Sung-Kyu Lim, and D.Z. Pan. A fast simulation framework for full-chip thermo-mechanical stress and reliability analysis of through-silicon-via based 3d ics. In *Electronic Components and Technology Conference*, pages 746–753, May 2011.

[12] J.W. Joyner, P. Zarkesh-Ha, and J.D. Meindl. A stochastic global net-length distribution for a three-dimensional system-on-a-chip (3d-soc). In *ASIC/SOC Conference, 2001. Proceedings. 14th Annual IEEE International*, pages 147–151, 2001.

[13] Qing K. Zhu. *High-Speed Clock Network Design*. Kluwer Academic Publishers, Norwell, MA, USA, 2002.

[14] Rupesh S. Shelar and Marek Patyra. Impact of local interconnects on timing and power in a high performance microprocessor. In *Proc. ACM Int. Symp. Phys. Design*, 2010.

[15] ITRS. Itrs. 2011. international technology roadmap for semiconductors. http://www.itrs.net/, 2011.

[16] Mark A. Franklin and Donald F. Wann. Asynchronous and clocked control structures for vlsi based interconnection networks. *SIGARCH Comput. Archit. News*, 10(3):50–59, April 1982.

[17] Michael A. B. Jackson, Arvind Srinivasan, and E. S. Kuh. Clock routing for high-performance ics. In *Proc. 27th Design Automation Conf. (DAC)*, pages 573–579, 1990.

[18] J. Cong, A. B. Kahng, and G. Robins. Matching-based methods for high-performance clock routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 12(8):1157–1169, Aug 1993.

[19] R. S. Tsay. An exact zero-skew clock routing algorithm. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 12(2):242–249, Feb 1993.

[20] Ting-Hai Chao, Yu-Chin Hsu, Jan-Ming Ho, and A.B. Kahng. Zero skew clock routing with minimum wirelength. *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, 39(11):799–814, Nov 1992.

[21] A.H. Farrahi, Chunhong Chen, A. Srivastava, G. Tellez, and M. Sarrafzadeh. Activity-driven clock design. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 20(6):705–714, Nov 2001.

[22] Jaewon Oh and M. Pedram. Gated clock routing for low-power microprocessor design. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 20(6):715–722, Jun 2001.

[23] M. Donno, A. Ivaldi, L. Benini, and E. Macii. Clock-tree power optimization based on rtl clock-gating. In *Proc. 40th Design Automation Conf. (DAC)*, 2003.

[24] Hai Li, S. Bhunia, Yiran Chen, K. Roy, and T. N. Vijaykumar. Dcg: deterministic clock-gating for low-power microprocessor design. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 12(3):245–254, Mar 2004.

[25] Wei-Chung Chao and Wai-Kei Mak. Low-power gated and buffered clock network construction. *ACM Trans. Des. Autom. Electron. Syst.*, 13(1):20:1–20:20, Feb 2008.

[26] Jingwei Lu, Wing-Kai Chow, and Chiu-Wing Sham. Fast power- and slew-aware gated clock tree synthesis. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 20(11):2094–2103, Nov 2012.

[27] Weixiang Shen, Yici Cai, Xianlong Hong, and Jiang Hu. An effective gated clock tree design based on activity and register aware placement. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 18(12):1639–1648, Dec 2010.

[28] L. Benini, G. De Micheli, E. Macii, M. Poncino, and R. Scarsi. Symbolic synthesis of clock-gating logic for power optimization of synchronous controllers. *ACM Trans. Des. Autom. Electron. Syst.*, 4(4):351–375, October 1999.

[29] Leticia Bolzani, Andrea Calimera, Alberto Macii, Enrico Macii, and Massimo Poncino. Enabling concurrent clock and power gating in an industrial design flow. In *Proc. Des. Autom. Test in Euro.*, pages 334–339, 2009.

[30] Tak-Yung Kim and Taewhan Kim. Clock tree embedding for 3d ics. In *Proc. 15th Asia South Pac. Des. Autom. Conf. (ASP-DAC)*, pages 486–491, Jan 2010.

[31] Tak-Yung Kim and Taewhan Kim. Clock tree synthesis for tsv-based 3d ic designs. *ACM Trans. Des. Autom. Electron. Syst.*, 16(4):48:1–48:21, Oct 2011.

[32] Jaeseok Yang, J.S. Pak, Xin Zhao, Sung Kyu Lim, and D.Z. Pan. Robust clock tree synthesis with timing yield optimization for 3d-ics. In *Proc. 16th Asia South Pac. Des. Autom. Conf. (ASP-DAC)*, pages 621–626, Jan 2011.

[33] Chiao-Ling Lung, Yu-Shih Su, Hsih-Hsiu Huang, Yiyu Shi, and Shih-Chieh Chang. Through-silicon via fault-tolerant clock networks for 3-d ics. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, pages 1100–1109, July 2013.

[34] Heechun Park and Taewhan Kim. Synthesis of tsv fault-tolerant 3-d clock trees. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 34(2):266–279, Feb 2015.

[35] Tiantao Lu and Ankur Srivastava. Gated low-power clock tree synthesis for 3d-ics. In *Proceedings of the 2014 International Symposium on Low Power Electronics and Design*, ISLPED '14, pages 319–322, 2014.

[36] Moongon Jung, J. Mitra, D.Z. Pan, and Sung Kyu Lim. Tsv stress-aware full-chip mechanical reliability analysis and optimization for 3-d ic. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 31(8):1194–1207, Aug 2012.

[37] Chang Liu, Taigon Song, Jonghyun Cho, Joohee Kim, Joungho Kim, and Sung-Kyu Lim. Full-chip tsv-to-tsv coupling analysis and optimization in 3d ic. In *Proc. 48th Design Automation Conf. (DAC)*, pages 783–788, 2011.

[38] S. Lloyd. Least squares quantization in pcm. 28(2):129–137, 1982.

[39] Sematech. Concerns of 3d integration technology using tsv. http://www.sematech.org/meetings/archives/symposia/9028/Session2_3D/Lee_KangWook.pdf, 2010.

[40] ITRS. Itrs. 2010. international technology roadmap for semiconductors. http://www.itrs.net/, 2010.

[41] M. Edahiro. A clustering-based optimization algorithm in zero-skew routings. In *Proc. 30th Design Automation Conf. (DAC)*, pages 612–616, Jun 1993.

[42] N. Viswanathan and C.C.N. Chu. Fastplace: efficient analytical placement using cell shifting, iterative local refinement,and a hybrid net model. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 24(5):722–733, May 2005.

[43] C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55, January 2001.

[44] ISPD. Ispd 2009 clock network synthesis contest. http://ispd.cc/contests/09/ispd09cts.html, 2009.

[45] Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted voronoi diagrams and randomization to variance-based k-clustering: (extended abstract). In *Proceedings of the Tenth Annual Symposium on Computational Geometry*, SCG '94, pages 332–339, New York, NY, USA, 1994. ACM.

[46] M. Puech, Thevenoud, and et al. Fabrication of 3d packaging tsv using drie. In *Design, Test, Integration and Packaging of MEMS/MOEMS, 2008.*, pages 109–114, 2008.

[47] C. Huyghebaert, Van Olmen, and et al. Cu to cu interconnect using 3d-tsv and wafer to wafer thermocompression bonding. In *Interconnect Technology Conference (IITC), 2010 International*, pages 1–3, 2010.

[48] Tiantao Lu and Ankur Srivastava. Modeling and layout optimization for tapered tsvs. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 23(12):3129–3132, Dec 2015.

[49] Tiantao Lu and A. Srivastava. Detailed electrical and reliability study of tapered tsvs. In *3D Systems Integration Conference (3DIC), 2013 IEEE International*, pages 1–7, Oct 2013.

[50] Tiantao Lu and Ankur Srivastava. *Detailed Electrical and Reliability Study for Tapered TSVs*. Dec 2015.

[51] A. Klumpp, P. Ramm, and R. Wieland. 3d-integration of silicon devices: A key technology for sophisticated products. In *Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1678–1683, 2010.

[52] Xiaowu Zhang, T.C. Chai, and et al. Development of through silicon via (tsv) interposer technology for large die (21 x 21mm) fine-pitch cu/low-k fcbga package. In *Electronic Components and Technology Conference.*, pages 305–312, 2009.

[53] Imec's scientific report 2010,tsv processing. http://www.imec.be/ScientificReport/SR2010/2010/1159081.html. Accessed: 2013-08-10.

[54] Fumihiro Inoue and et al. Highly adhesive electroless barrier/cu-seed formation for high aspect ratio through-si vias. *Microelectron. Eng.*, pages 164–167, 2013.

[55] Nagarajan and et al. Development of a novel deep silicon tapered via etch process for through-silicon interconnection in 3-d integrated systems. In *ECTC*, page 5 pp., 2006.

[56] B. Kim, C. Sharbono, Tom Ritzdorf, and D. Schmauch. Factors affecting copper filling process within high aspect ratio deep vias for 3d chip stacking. In *Electronic Components and Technology Conference, 2006. Proceedings. 56th*, pages 6 pp.–, 2006.

[57] Zheng Xu and Jian-Qiang Lu. High-speed design and broadband modeling of through-strata-vias (tsvs) in 3d integration. *Components, Packaging and Manufacturing Technology, IEEE Transactions on*, pages 154 –162, 2011.

[58] Ansys workbench. http://www.ansys.com/.

[59] A.B. Kahng and S. Muddu. An analytical delay model for rlc interconnects. *TCAD*, pages 1507–1514, 1997.

[60] S.S. Sapatnekar. Wire sizing as a convex optimization problem: exploring the area-delay tradeoff. *TCAD*, pages 1001–1011, 1996.

[61] Jason Cong and Cheng-Kok Koh. Interconnect layout optimization under higher-order rlc model. In *ICCAD*, pages 713–720, 1997.

[62] Moongon Jung, J. Mitra, D.Z. Pan, and Sung Kyu Lim. Tsv stress-aware full-chip mechanical reliability analysis and optimization for 3-d ic. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1194–1207, Aug 2012.

[63] Cheng fu Chen. Characterization of in-plane stress in tsv array - a unit model approach. In *Electronic Components and Technology Conference*, pages 2020–2026, May 2014.

[64] Guojie Luo, Yiyu Shi, and J. Cong. An analytical placement framework for 3-d ics and its extension on thermal awareness. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 510–523, April 2013.

[65] Suk-Kyu Ryu, Kuan-Hsun Lu, Xuefeng Zhang, Jang-Hi Im, Paul S. Ho, and Rui Huang. Impact of near-surface thermal stresses on interfacial reliability of through-silicon vias for 3-d interconnects. *IEEE Transactions on Device and Materials Reliability*, pages 35–43, March 2011.

[66] Ching-Han Tsai and Sung-Mo Kang. Cell-level placement for improving substrate thermal distribution. *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, pages 253–266, November 2006.

[67] Meng-Kai Hsu, Yao-Wen Chang, and V. Balabanov. Tsv-aware analytical placement for 3d ic designs. In *Design Automation Conference*, pages 664–669, June 2011.

[68] Brent Goplen and Sachin Sapatnekar. Efficient thermal placement of standard cells in 3d ics using a force directed approach. In *Proceedings of the IEEE/ACM International Conference on Computer-aided Design*, ICCAD '03, pages 86–, 2003.

[69] B. Goplen and S. Sapatnekar. Placement of 3d ics with thermal and interlayer via considerations. In *Design Automation Conference*, pages 626–631, June 2007.

[70] Comsol multiphysics. http://www.comsol.com/.

[71] R.L. de Orio, H. Ceric, and S. Selberherr. Electromigration failure in a copper dual-damascene structure with a through silicon via. *Microelectronics Reliability*, pages 1981 – 1986, 2012.

[72] Tiantao Lu and Ankur Srivastava. Electromigration-aware clock tree synthesis for tsv-based 3d-ics. In *Proceedings of the 25th Edition on Great Lakes Symposium on VLSI*, GLSVLSI '15, pages 27–32, New York, NY, USA, 2015.

[73] Lihua Liang JianPing Jing and Guang Meng. Electromigration simulation for metal lines. *Journal of Electronic Packaging*, 2010.

[74] T. Frank, S. Moreau, C. Chappaz, P. Leduc, L. Arnaud, A. Thuaire, E. Chery, F. Lorut, L. Anghel, and G. Poupon. Reliability of tsv interconnects: Electromigration, thermal cycling, and impact on above metal level dielectric. *Microelectronics Reliability*, pages 17 – 29, 2013.

[75] Tiantao Lu and Ankur Srivastava. Electrical-thermal-reliability co-design for tsv-based 3d-ics. In *ASME 2015 International Technical Conference and Exhibition on Packaging and Integration of Electronic and Photonic Microsystems*, pages 1–10, 2015.

[76] Tiantao Lu, Zhiyuan Yang, and Ankur Srivastava. Post-placement optimization for thermal-induced mechanical stress reduction. In *2016 IEEE Computer Society Annual Symposium on VLSI*, pages 1–6, 2016.

[77] Cheng-Kok Koh Chen Li. On improving recursive bipartitioning-based placement. *Technical Report TR-ECE-03-14, Purdue University ECE*, 2003.

[78] Tiantao Lu, Zhiyuan Yang, and Ankur Srivastava. Electromigration-aware placement for 3d-ics. In *International Symposium on Quality Electronic Design*, page pp, 2016.

[79] C. Albrecht. Iwls benchmark effort. In *Int. Workshop Logic Synthesis*, June 2005.

[80] K. Athikulwongse, A. Chakraborty, Jae seok Yang, D.Z. Pan, and Sung Kyu Lim. Stress-driven 3d-ic placement with tsv keep-out zone and regularity study. In *ICCAD*, pages 669–674, Nov 2010.

[81] Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu. Raidr: Retention-aware intelligent dram refresh. *SIGARCH Comput. Archit. News*, 40(3):1–12, June 2012.

[82] I.S. Bhati, Mu-Tien Chang, Z. Chishti, Shih-Lien Lu, and B. Jacob. Dram refresh mechanisms, trade-offs, and penalties. *Computers, IEEE Transactions on*, PP(99):1–1, 2015.

[83] J. Meza, Qiang Wu, S. Kumar, and O. Mutlu. Revisiting memory errors in large-scale production data centers: Analysis and modeling of new trends from the field. In *Dependable Systems and Networks (DSN), 2015 45th Annual IEEE/IFIP International Conference on*, pages 415–426, June 2015.

[84] Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber. Dram errors in the wild: A large-scale field study. *SIGMETRICS Perform. Eval. Rev.*, 2009.

[85] Vilas Sridharan and Dean Liberty. A study of dram failures in the field. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '12, 2012.

[86] Xun Jian, N. DeBardeleben, S. Blanchard, V. Sridharan, and R. Kumar. Analyzing reliability of memory sub-systems with double-chipkill detect/correct. In *Dependable Computing (PRDC), 2013 IEEE 19th Pacific Rim International Symposium on*, pages 88–97, Dec 2013.

[87] D Roberts and P Nair. Faultsim: A fast, configurable memory-resilience simulator. In *The Memory Forum: In conjunction with ISCA*, volume 41.

[88] R. Baumann. Soft errors in advanced computer systems. *Design Test of Computers, IEEE*, 2005.

[89] Vilas Sridharan, Nathan DeBardeleben, Sean Blanchard, Kurt B. Ferreira, Jon Stearley, John Shalf, and Sudhanva Gurumurthi. Memory errors in modern systems: The good, the bad, and the ugly. *SIGPLAN Not.*, 2015.

[90] Nosayba El-Sayed, Ioan A. Stefanovici, George Amvrosiadis, Andy A. Hwang, and Bianca Schroeder. Temperature management in data centers: Why some (might) like it hot. *SIGMETRICS Perform. Eval. Rev.*, 2012.

[91] C. L. Chen and M. Y. Hsiao. Error-correcting codes for semiconductor memory applications: A state-of-the-art review. *IBM J. Res. Dev.*, 1984.

[92] Timothy J Dell. A white paper on the benefits of chipkill-correct ecc for pc server main memory.

[93] M.Y. Hsiao. A class of optimal minimum odd-weight-column sec-ded codes. *IBM Journal of Research and Development*, 14(4):395–401, July 1970.

[94] Shubhendu S. Mukherjee, Joel Emer, Tryggve Fossum, and Steven K. Reinhardt. Cache scrubbing in microprocessors: Myth or necessity? In *Proceedings of the 10th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC'04)*, PRDC '04, 2004.

[95] T.J. Slegel, III Averill, R.M., M.A. Check, B.C. Giamei, B.W. Krumm, C.A. Krygowski, W.H. Li, J.S. Liptay, J.D. MacDougall, T.J. McPherson, J.A. Navarro, E.M. Schwarz, K. Shum, and C.F. Webb. Ibm's s/390 g5 microprocessor design. *Micro, IEEE*, 19(2):12–23, Mar 1999.

[96] T.M. Austin. Diva: a reliable substrate for deep submicron microarchitecture design. In *Microarchitecture, 1999. MICRO-32. Proceedings. 32nd Annual International Symposium on*, pages 196–207, 1999.

[97] S.S. Mukherjee, M. Kontz, and S.K. Reinhardt. Detailed design and evaluation of redundant multi-threading alternatives. In *Computer Architecture, 2002. Proceedings. 29th Annual International Symposium on*, pages 99–110, 2002.

[98] Pietro Mercati, Andrea Bartolini, Francesco Paterna, Tajana Simunic Rosing, and Luca Benini. Workload and user experience-aware dynamic reliability management in multicore processors. In *Proceedings of the 50th Annual Design Automation Conference*, DAC '13, pages 2:1–2:6, 2013.

[99] Manjunath Shevgoor, Jung-Sik Kim, Niladrish Chatterjee, Rajeev Balasubramonian, Al Davis, and Aniruddha N. Udipi. Quantifying the relationship between the power delivery network and architectural policies in a 3d-stacked memory device. In *Proceedings of the 46th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-46, pages 198–209, 2013.

[100] Rafael Ubal, Byunghyun Jang, Perhaad Mistry, Dana Schaa, and David Kaeli. Multi2sim: a simulation framework for cpu-gpu computing. In *Proceedings of the 21st international conference on Parallel architectures and compilation techniques*, pages 335–344. ACM, 2012.

[101] Sheng Li, Jung Ho Ahn, Richard D Strong, Jay B Brockman, Dean M Tullsen, and Norman P Jouppi. Mcpat: an integrated power, area, and timing modeling framework for multicore and manycore architectures. In *Microarchitecture, 2009. MICRO-42. 42nd Annual IEEE/ACM International Symposium on*, pages 469–480. IEEE, 2009.

[102] Wei Huang, Shougata Ghosh, Siva Velusamy, Karthik Sankaranarayanan, Kevin Skadron, and Mircea R Stan. Hotspot: A compact thermal modeling methodology for early-stage vlsi design. *TVLSI*, 14(5):501–513, 2006.

[103] C. Serafy, A. Bar-Cohen, A. Srivastava, and D. Yeung. Unlocking the true potential of 3-d cpus with microfluidic cooling. *TVLSI*, PP(99):1–1, 2015.

[104] Jun So Pak, Joohee Kim, Jonghyun Cho, Kiyeong Kim, Taigon Song, Seungyoung Ahn, Junho Lee, Hyungdong Lee, Kunwoo Park, and Joungho Kim. Pdn impedance modeling and analysis of 3d tsv ic by using proposed p/g tsv array model based on separated p/g tsv and chip-pdn models. *TCPM*, 1(2):208–219, Feb 2011.

[105] K. Roy, S. Mukhopadhyay, and H. Mahmoodi-Meimand. Leakage current mechanisms and leakage reduction techniques in deep-submicrometer cmos circuits. *Proceedings of the IEEE*, 91(2):305–327, Feb 2003.

[106] Jie Meng, Katsutoshi Kawakami, and Ayse K. Coskun. Optimizing energy efficiency of 3-d multicore systems with stacked dram under power and thermal constraints. In *Proceedings of the 49th Annual Design Automation Conference*, DAC '12, pages 648–655, 2012.

[107] Ramkumar Jayaseelan and Tulika Mitra. Temperature aware task sequencing and voltage scaling. In *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design*, ICCAD '08, pages 618–623, 2008.

[108] D. Skinner and W. Kramer. Understanding the causes of performance variability in hpc workloads. In *Workload Characterization Symposium, 2005. Proceedings of the IEEE International*, pages 137–149, Oct 2005.

[109] Gabriel H. Loh. 3d-stacked memory architectures for multi-core processors. In *Proceedings of the 35th Annual International Symposium on Computer Architecture*, ISCA '08, pages 453–464, 2008.

[110] Gang Huang, M. Bakir, A. Naeemi, H. Chen, and J.D. Meindl. Power delivery for 3d chip stacks: Physical modeling and design implication. In *Electrical Performance of Electronic Packaging, 2007 IEEE*, pages 205–208, Octl 2007.

[111] Steven Cameron Woo, Moriyoshi Ohara, Evan Torrie, Jaswinder Pal Singh, and Anoop Gupta. The splash-2 programs: Characterization and methodological considerations. *SIGARCH Comput. Archit. News*, 23(2):24–36, May 1995.

[112] Christian Bienia, Sanjeev Kumar, Jaswinder Pal Singh, and Kai Li. The parsec benchmark suite: Characterization and architectural implications. In *Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques*, PACT '08, pages 72–81, 2008.

[113] Bing Shi, A. Srivastava, and Peng Wang. Non-uniform micro-channel design for stacked 3d-ics. In *DAC'11*, 2011.

[114] T. Chantem, Y. Xiang, X. S. Hu, and R. P. Dick. Enhancing multicore reliability through wear compensation in online assignment and scheduling. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, pages 1373–1378, March 2013.

[115] W. Song, S. Mukhopadhyay, and S. Yalamanchili. Architectural reliability: Lifetime reliability characterization and management ofmany-core processors. *IEEE Computer Architecture Letters*, 14(2):103–106, July 2015.

[116] P. Mercati, A. Bartolini, F. Paterna, T. Simunic Rosing, and L. Benini. Workload and user experience-aware dynamic reliability management in multicore processors. In *Design Automation Conference (DAC), 2013 50th ACM/EDAC/IEEE*, pages 1–6, May 2013.

[117] C. Zhuo, D. Sylvester, and D. Blaauw. Process variation and temperature-aware reliability management. In *2010 Design, Automation Test in Europe Conference Exhibition (DATE 2010)*, pages 580–585, March 2010.