RUN-TIME THERMAL MANAGEMENT

BASED ON TASK MIGRATION TECHNIQUES

IN 3D CHIP MULTIPROCESSORS

by

Sulaiman Obaid Aljeddani

BSc., Umm Al-Qura University, Makkah, Saudi Arabia, June 2013.

A thesis presented to Ryerson University
in partial fulfillment of the requirements for the degree of
Master of Applied Science
in the program of
Electrical and Computer Engineering.

Toronto,Ontaio, Canada, 2018.

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

DEDICATION

I would love to dedicate this thesis ***to my dearest father and my dearest mother****, who have ever been supportive and encouraging throughout my life. I would also like to dedicate this work to my* ***siblings****, whose support and guidance were key factors to my success. I would also like to dedicate this thesis to* ***my dearest wife****, who led me through the heart of darkness with the light of support and success. Last but not least, I would like to dedicate this thesis to* ***my lovely daughter, Rateel****, who has been a source of inspiration and hope.*

ACKNOWLEDGEMENTS

First and foremost, I must acknowledge my greatest gratitude to Allah, the most gracious and the most merciful, for his help and blessings. I am totally sure that this work would never have become a success without his guidance.

It is with great appreciation and gratitude that I take this opportunity to thank the individuals and the organizations who contributed to the accomplishment of this thesis.

First, I would like to express my deepest gratitude and appreciation to my **Supervisor: Prof. F. Mohammadi**, who provided me with generous, unlimited guidance and support without which I would not have been able to complete this work. She spared no effort to ensure the success of this thesis, and she closely monitored my progress from the very beginning to the very end.

In addition, I would also like to express my gratitude to Ryerson University for providing me with this opportunity to complete this work. I also owe deep gratitude to the **Professors Lev Kirischian**, **Vadim Geurkov** and **Kaamran Raahemifar** whose graduate courses were very critical to my understanding of the basics of the Computer Engineering field that, in turn, put me on the right track from the very beginning.

Furthermore, I would like to take this opportunity to express my sincere appreciation to the **Government of Saudi Arabia** represented by the **Ministry of Education**, **Saudi Cultural Bureau in Canada**, for the unlimited financial support and academic guidance.

Last but not least, I am wholeheartedly thankful to **my family** for the effort and support they have provided me with every step of the way. Their generous encouragement and assistance made this dream accomplishable. I am so grateful and indebted to them all for everything I have become today.

ABSTRACT

Runtime Thermal Management Based on Task Migration Techniques in 3D Chip Multiprocessors

By

Sulaiman Obaid Aljeddani

Master of Applied Science, Electrical and Computer Engineering

Ryerson University, Toronto, 2018.

The industry trend of Chip Multiprocessors (CMPs) architecture is to move from 2D CMPs to 3D CMPs architecture for obtain higher performance, more reliability, and reduced memory access latency. However, one key challenge in designing the 3D CMPs is the thermal issue as a result of maximizing the throughput . Therefore, applying Runtime Thermal Management (RTM) has become crucial for controlling thermal hotspots. In this thesis, two methods of run-time task migration are presented to balance the temperature and reduce the number of hotspots in 3D CMPs. The proposed techniques consider hotspots both in the core and the memory layers simultaneously to make the optimum run-time task migration decisions. The first proposed approach is divided into two algorithms working in parallel, which aim at maximizing the throughput on the 3D CMPs while satisfying the peak temperature constraints. Experimental results show that the proposed architecture yields up to 60% reduction in overall chip energy. The proposed architecture improves the IPC for *canneal* and *fluidanimate* applications by 18% and 14%, respectively. In the second method, the proposed technique migrates the hottest core with the optimal coldest core instead of the coldest core in the core layer. The optimal coldest core is selected by considering hotspots

DRAM banks in the memory layer. The simulation results indicate up to 33℃ (on average 24℃) reduction in the cores' temperature of the target 3D CMPs. Finally, the proposed techniques are efficiency clarified in the simulation results that the maximum temperature of cores in the core and memory layers are both less than the maximum temperature limit; 80°C.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1: Introduction

1.1. Introduction

With the advanced of transistor technologies, processors have become very advanced and complicated since 2005 [49]. In 1965, Gordon Moore mentioned that “the number of transistors in a single chip are doubled approximately every 18 months” [32]. This definition is known as Moore's law which is made an evolution in the trend of processors fabrication. Moore's law predicts that the number of transistors will increase by double, therefore many advantages can be achieved. For example, increasing the chip performance will be achieved as a result of the increasing in the power consumption; thereby, the highly increased demand of high performance can be achieved by increasing the number of transistors. Figure 1.1, shows the increase in the number of transistors between 1971 to 2011 [34].

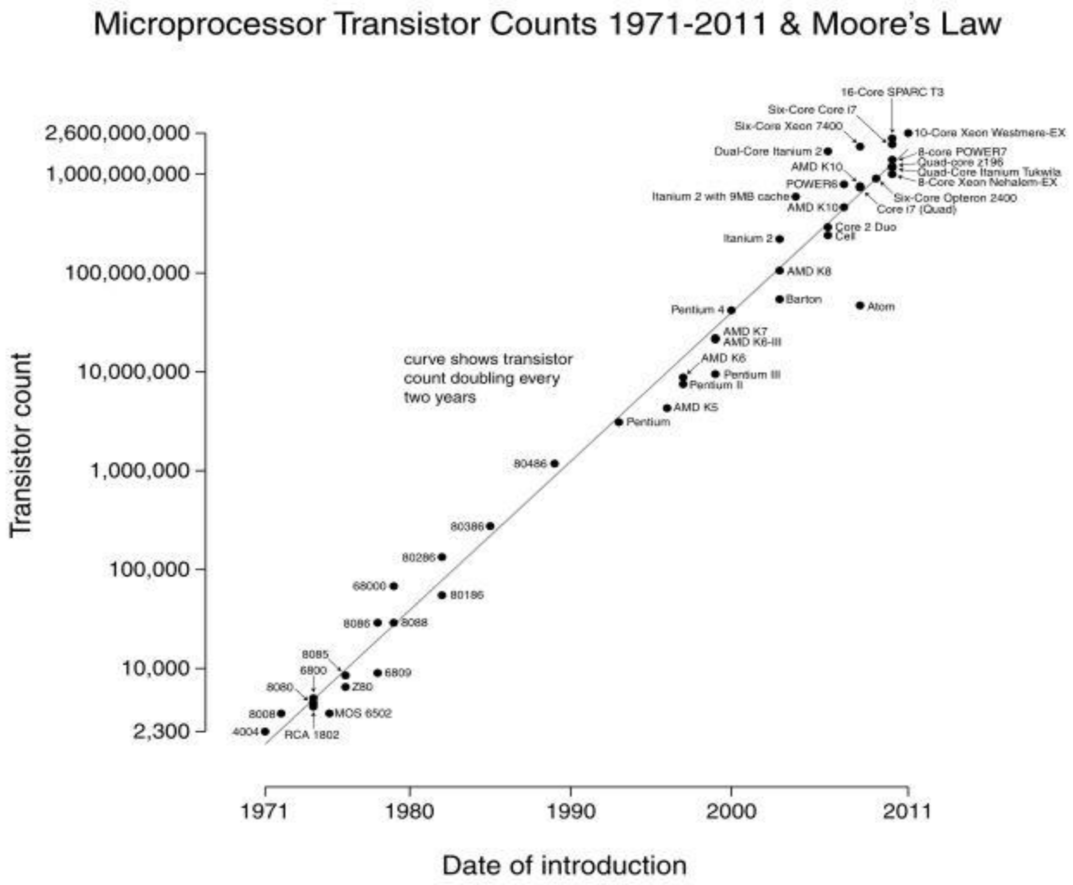


Figure 1.1: Number of transistors from 1971 to 2011.

In this context, this change makes it possible since 2005 to design sub-processors on the chip instead of a single processor which is known as multi-core processors that lead to a significant improvement in the processors' performance. In multi-core processors, each core can perform identical functions to those performed by a single core processor.

On the other hand, the memory in the new generation of multi-core processors plays an important role. In fact, to operate a large number of operations in multi-core processors, the need to more memory spaces in the entire chip becomes an urgent need and an important challenge as a result of the high demand to store the huge amount of data and instructions. Therefore, it leads to the integration of on-chip memories ranging from Megabytes to Gigabytes in processors. Figure 1.2, demonstrates how embedded memories are increasing on the chip from 20% in 1999 to around 70% in 2017 [31].

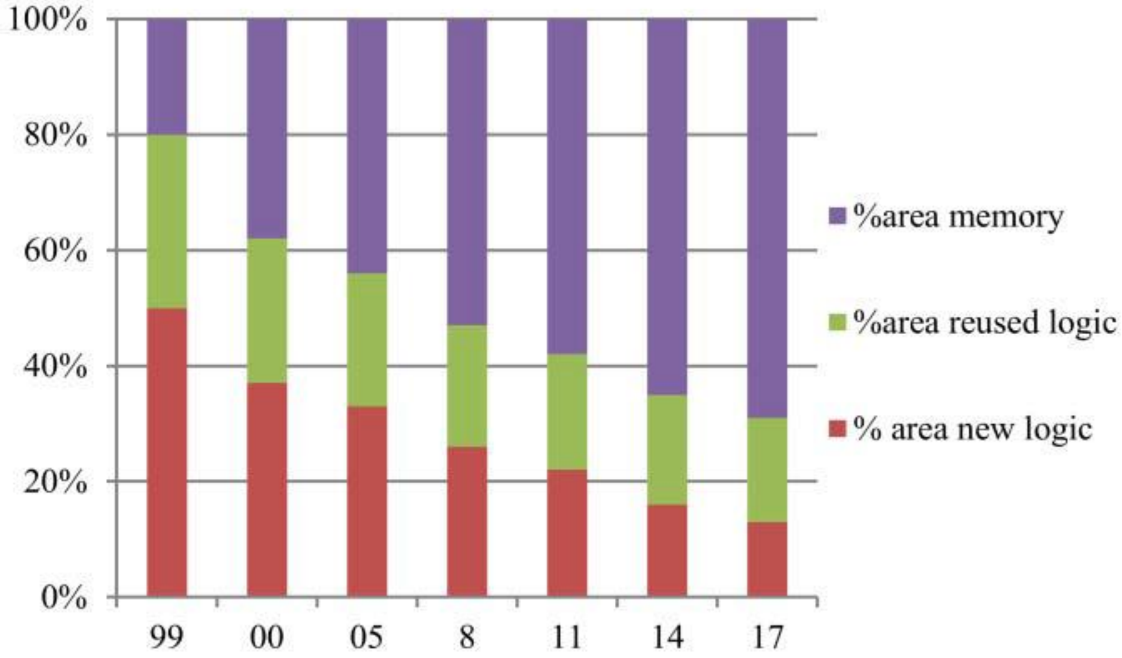


Figure 1.2: Embedded memory space on the chip between 1999 to 2017.

Over time, embedded systems have moved away from Two-Dimensional Integrated Circuits (2D ICs) to Three-Dimensional Integrated Circuits (3D ICs). The 3D ICs, when compared with 2D ICs designs, reduce interconnection wire length which, results in lower power consumption and shorter communication latency. Therefore, by combining the 3D ICs technology and Chip Multiprocessors (CMPs) such that they become 3D CMPs, they result in higher performance higher efficiency and reliable systems.

In this context, to combine the increased demand of having higher number of cores and more memory space in a chip, CMPs tend to move from 2D CMPs to 3D CMPs architecture. The 3D CMPs provide many features when compared with 2D CMPs designs, they increase the chip performance and reliability, reduce memory access latency, increase memory bandwidth, and also reduce interconnection wire length which results in lower power consumption and shorter communication latency [28]. The 3D CMPs architecture contains different layers which are core layer and memory layers as shown in Figure 1.3 [33]. Moreover, the architecture of CMPs has been extended to the 3D CMPs architecture by using Through Silicon Vias (TSVs).

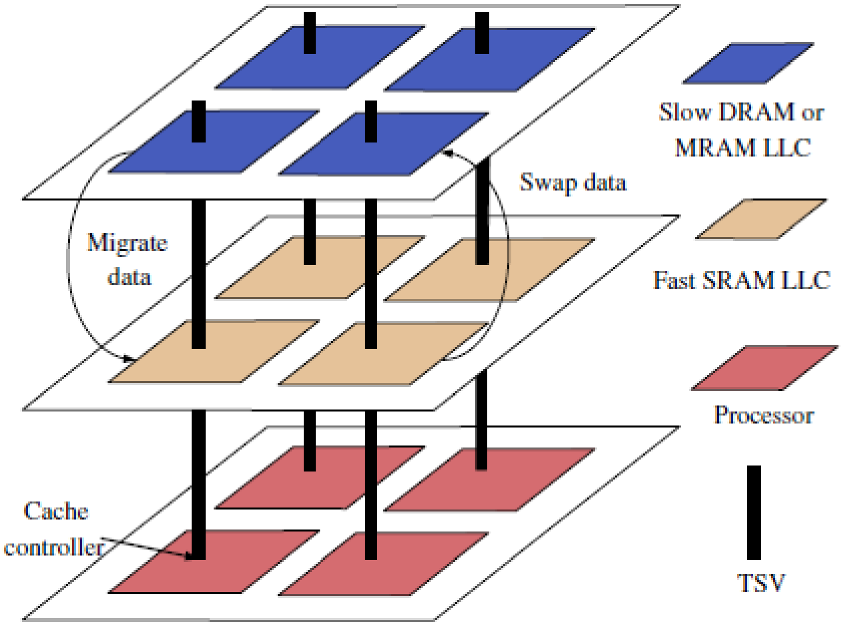


Figure 1.3: Architecture of 3D CMPs with three layers.

Even though the 3D CMPs achieve a significant improvement in the processors' performance by increasing the chip's power consumption, the drawbacks of the above introduced advantages result in high temperature spots, which are called thermal hotspots. In fact, the existence of high temperature spots in a chip is normal as long as they do not reach the critical temperature which is generally around 80℃, as far as available cooling technologies are concerned [48]. However, the highly increasing in thermal hotspots have become a major challenge for the 3D CMPs since hotspots can cause performance degradation, reducing reliability, decreasing chip life span, and eventually lead to system failure.

In this context, providing a solution to face these challenges has become crucial to controlling and distributing thermal hotspots in a chip. Therefore, applying Run-time Thermal Management (RTM) techniques can play an important role not only to balance the thermal hotspots on the 3D CMPs, but also to enable the 3D CMPs to operate at a favorable performance without any emergence fear of new hotspots. One of the most common techniques for RTM is run-time task migration technique, which is intended to migrate tasks from a hot core to a low temperature core to prevent the hot core from becoming a hotspot.

In this thesis, two new run-time task migration techniques for the 3D CMPs have been proposed. In this regard, in the 3D CMPs, migrating the task to a low temperature core in the core layer without considering hotspots in the stacked memory layer has the potential to cause the emergence of new hotspots. Therefore, in this thesis, the keynote of the proposed techniques is to consider hotspots on the two stacked layers simultaneously.

In this context, it is crucial that the system must consider hotspots on the two stacked layers, both the core layer and the memory layer, simultaneously; thus, making the optimum task migration decision more efficient. Therefore, it is crucial that the system analyzes the gathered cores' temperature information and then selects the optimal coldest core to be migrated with a hotspot rather than selecting the coldest core. The optimal coldest core refers to a cold core in the core layer that is not located under a hotspot in the memory layer. Therefore, this procedure can lead to make the optimum task migration decisions; thereby improve the 3D CMPs performance.

1.2. Thesis Overview

1.2.1. Overview

In this section, we introduce the recent major problems in the area of 3D CMPs. Also, the thesis objectives are presented to show the main goals of the thesis.

1.2.2. Major Problems in the Area of 3D CMPs

Nowadays, with the increased demand of high performance, 3D CMPs has led to the increase in the power consumption of the chip; which causes the emergence of hotspots. Therefore, from this point, the chip is facing a new major challenge. This challenge regards the increased temperature of the 3D CMPs plays a critical role in the performance degradation.

1.2.3. Thesis Objectives

The objective of this thesis is to solve the problem of thermal hotspots challenge, by applying RTM techniques. In this thesis, two approaches of run-time task migration techniques in the 3D CMPs are used. These two proposed approaches aim to proceed run-time task migration techniques in order to satisfy the peak temperature constraint on the 3D CMPs, which leads to increased performance and reliability of 3D CMPs.

1.3. Thesis Contributions

This dissertation makes the following contributions:

1. The proposed migration techniques including two approaches aim to proceed run-time task migration techniques to achieve a balanced thermal distribution on the 3D CMPs.
2. The presented approaches consider hotspots on two stacked layers simultaneously both in the core and memory layers, by analyzing hotspots' information to make optimum task migration decisions.
3. In this work, a centralized hardware named Migration Control Unit (MCU) is presented. MCU analyzes the gathered cores' information on the 3D CMPs and then makes the optimum decision for selecting the optimal coldest core in the core layer rather than the coldest core to be migrated with a hotspot.
4. In the first approach, the proposed migration technique gathers the temperature of cores in each layer using performance-counters instead of thermal sensors.

1.4. Thesis Organization

The remaining of the thesis is organized as follows:

Chapter 2

Chapter 2 gives a background review about processors' architectures and thermal management techniques. In this chapter, we review the preliminary and necessary materials and architectures to continue with this thesis. For instance: thermal problems and power wall, multi-core and many-core processors, memory and on-chip memory systems, 2D CMPs and 3D CMPs, and lastly, trends on power aware CMPs are all reviewed in this chapter.

Chapter 3

Chapter 3 discusses a summary of related work in the RTM techniques on multi-core processors. In these days, managing the power consumption under the specified power and the thermal budget is a hot topic. These techniques are required to keep the CMPs working state below the maximum temperature and power constraints.

Chapter 4

In chapter 4, we present the first approach of the run-time task migration that is used. This approach aims to balance the temperature and the number of hotspots in the 3D CMPs without any performance degradation. The proposed approach is divided into two algorithms that aim at maximizing the throughput on the 3D CMPs while satisfying the peak temperature constraint. Finally, at the end of this chapter, experiments are performed, and results are evaluated.

Chapter 5

In chapter 5, we proposed a new run-time task migration technique to control hotspots in the 3D CMPs. The second proposed technique migrates the hottest core on the core layer in the 3D CMPs with the optimal coldest core rather than the coldest core. In this chapter, we explained in detail the proposed technique, and presented experimental results.

Chapter 6

Finally, chapter 6 gives a summary of the given chapters, draws the main conclusions of the proposed run-time task migration techniques, and presents the future work section. In addition, the reference section cavers all the resources; which are used in this thesis.

Chapter 2: Background Review on Processors Architectures and Thermal Management Techniques

2.1. Introduction

Computer architectures have improved rapidly in the last five decades in terms of computational power and architecture complexity mainly due to the fast development of semiconductor fabrication techniques. The number of on-chip transistors used has increased steadily and doubled every eighteen months according to Moore's law [32]. Shrinking technology size reduced the channel length, and consequently improved the latency problem, and scaling clock frequency up. However, the latter has become no longer feasible option after 2005. Since the mid of the seventies, the scale of performance and energy went together with scaling transistors' numbers, which is known as Dennard

Constant Field Scaling [30]. The latest process technology brought Dennard Scaling to its end in 2005, where the emerging power density forces the chip with a limited power budget. Chips are required to be more energy-efficient to work within this limited power.

The fabrication of multiple cores in a single processor improved the chip performance with fixed or even lower clock frequency, which was proposed to Network on Chip (NoC) as a solution to deal with on-chip communication issues. NoC communication architecture was introduced as a novel technology to effectively utilize tens, rather hundreds of simple and lower power cores that can be integrated in a single CMPs. This utilization of those many cores in a single chip requires more on-chip memory as well as NoCs resources to handle the emerging on-chip communications. In those architectures, i.e. many-core systems, a significant portion of chip area is allocated to on-chip memories and memory systems, which leads to a major contribution to the overall chip power consumption.

Enhancing energy efficiency for the uncore components, such as memory systems and on-chip interconnection in parallel with cores, is a critical task in new CMPs designs. To avoid the major degradation in new design, the effect of the system performance in parallel with power management must be taken into account. The performance degradation happens in a processor CMOS-based regarding to the increased amount of clock frequencies per cycle while a huge number of transistors in the chip that results in increasing the power density which leads to increase leakage power and then cause higher temperature. In this context, it is required to monitor the system’s performance at run-time and apply new power management techniques accordingly. The proposed techniques in this work aim at proposing run-time migration algorithms in 3D CMPs to manage power in both memory and core components and to manage thermal constraints.

2.2. Thermal Problem and Power Wall

Figure 2.1, shows that the rate of general purpose processors' performance growth measured by operating frequency has slowed down since 2005 [34]. The slow-down resulted due to the fact that performance enhancements had been steadily increasing at the expense of increasing power density as shown in Figure 2.2 [34]. When power density hits 1W/mm^2 in 2005, chip designers could not trade power for performance anymore as they approach the limit of efficiency in air-cooling facilities.

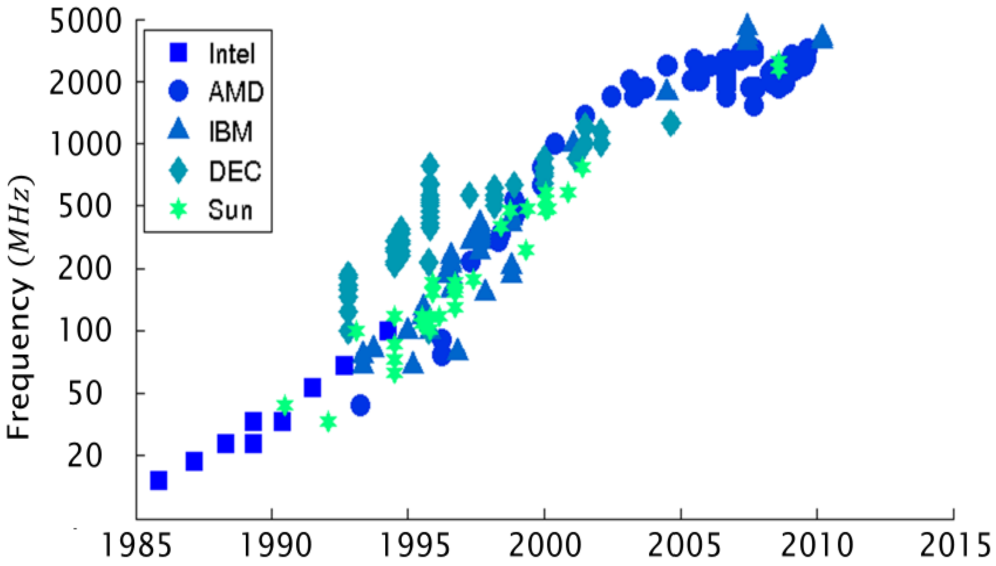


Figure 2.1: The frequency scaling of processor designs over time.

In this context, technology scaling had been steadily moving down until 2005 when it slowed down due to end of Dennard Constant Field Scaling [30]. The latter states that if all of the physical dimensions and the threshold voltage of a transistor are scaled down by a factor of α, the energy required to switch a transistor scales down by a factor of 3α [34]. This meant that as feature size has dropped by a factor of √2 with each technology node, chips were able to have the double number of transistors, increasing the processor's operating frequency, while still maintaining a constant power density. Due to leakage power concerns, however, threshold voltages (V_{th}) are no longer scaling at the same rate as operating voltage (V_{dd}) and the rest of the transistor, as depicted in Figure 2.3 [36], and thus V_{dd} scaling has dramatically slowed as well.

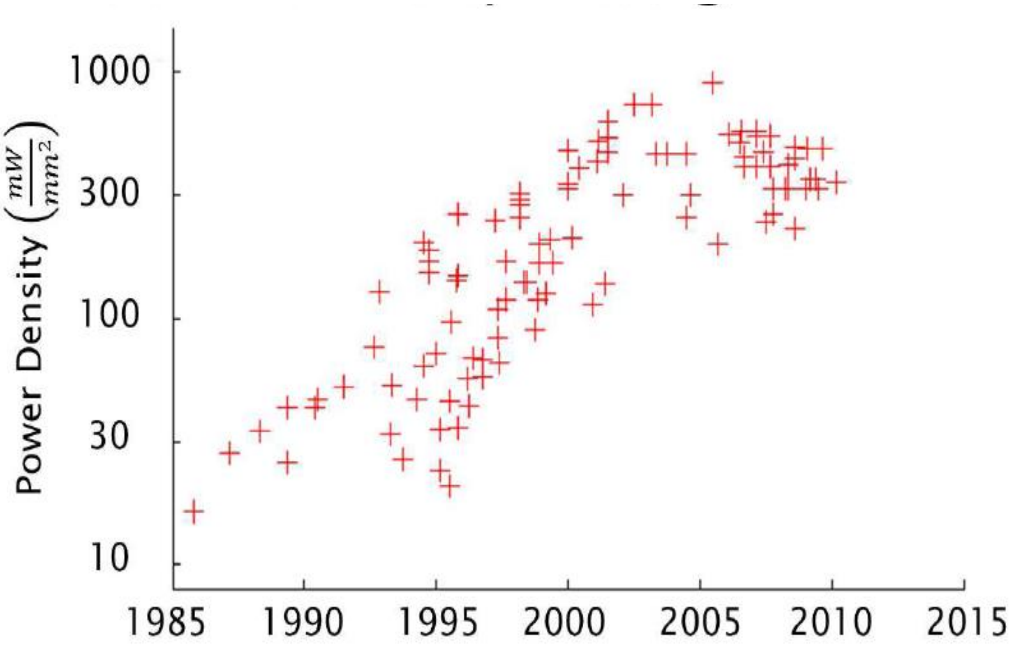


Figure 2.2: The power density of processor designs over time.

Dark Silicon Phenomenon states that every new chip generation will have a limited power budget which cannot be exceeded [35]. In the trend of technology scaling, the fraction of transistors that can operate at the full frequency is decreased. International Technology Roadmap for Semiconductors (ITRS) projected that 21% and 50% of the chip will be dark (off) at technologies 22nm and 8nm, respectively [36].

The system performance is measured by the number of operations per second (Op/Sec) and power is related to the product of consumed energy per operation ($Energy/Op$). For a limited chip power budget, the only way to increase the performance is by reducing the consumed energy per operation as shown in Equation 2.1.

$$Performance\left(\frac{Op}{s}\right) \propto \frac{Power}{\left(\frac{Energy}{Op}\right)} \tag{2.1}$$

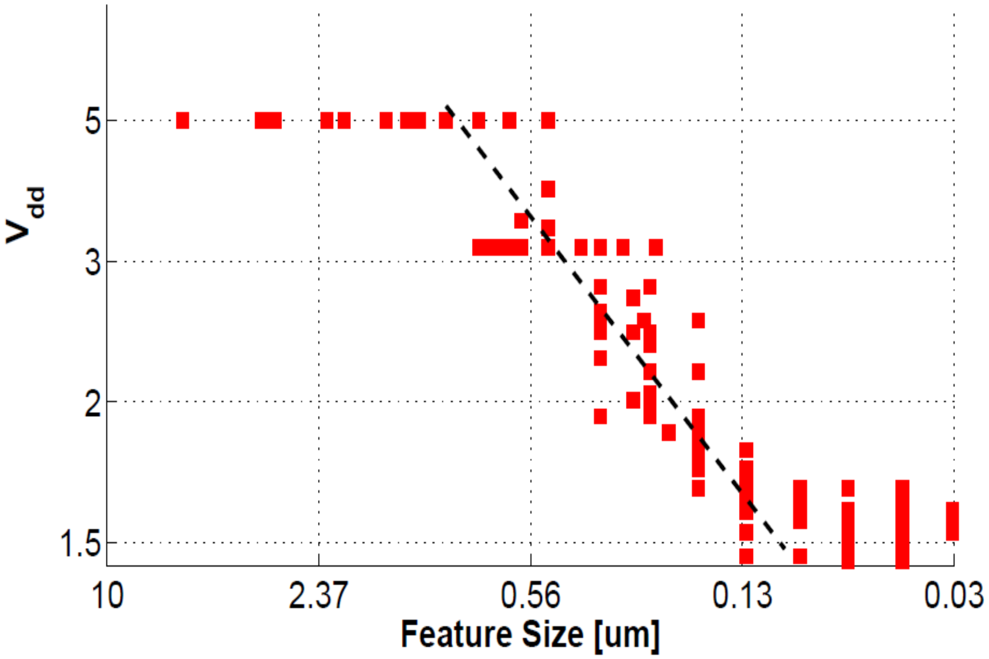


Figure 2.3: Voltage versus feature size.

2.3. Multiprocessor to Many Core Processor Extension

Multiple core processors successfully replaced the traditional approach of increasing the clock frequency by enhancing parallel computing. The nature of the task and the capability of the operating system which is responsible for dividing and scheduling tasks over available resources are two factors that determine the performance enhancement of the parallel computing.

Moving away from the general purpose processor to optimizing the data path for a certain class of algorithms and custom accelerators can achieve around 1000 higher energy efficiency than general purpose processors [34]. Figures 2.4, depicts the System on Chip (SoC) benefits compared to other processors' technology with normalized data at 32nm technology [34].

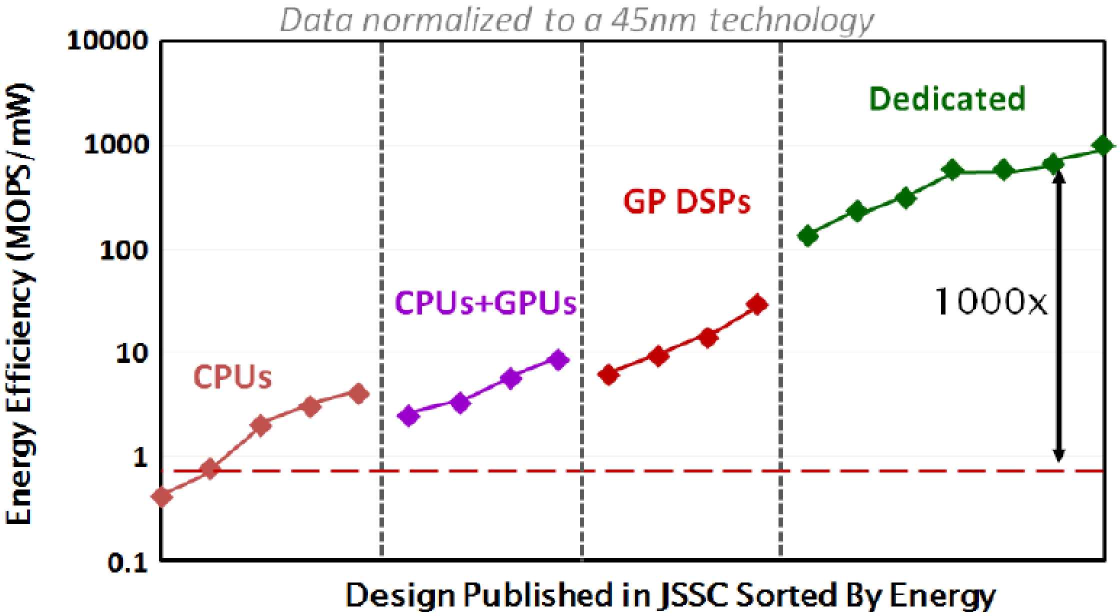


Figure 2.4: Energy efficiency for algorithms implemented on different platforms.

Incorporating several heterogeneous components such as different types of cores, memory banks, and communication resources make the on-chip communication a challenging task, especially the limitations of bus architectures like scalability and high-power consumption makes busses a bottleneck in SoC communication.

The success of computer networks and switch-based interconnection has motivated researchers to propose the NoC architecture as a viable solution in creating complex SoCs [37,40]. In multi/many core systems, simple low power cores can be connected based on packet-based on-chip interconnection including routers and links. These routers and links are the backbone of the components connecting on the chip. The number of cores can be in order tens or hundreds, whereas the numbers of routers is decided based on the used network topology. Figure 2.5, shows a two-dimensional 4×4 NoC in Mesh topology [37]. Each of these cores are called Processing Elements (PE) or Intellectual Property (IP) or they can be assigned to be a memory bank.

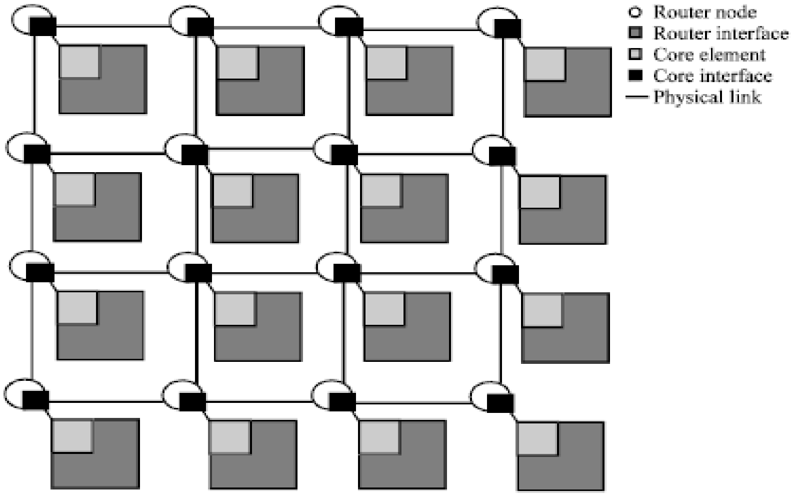


Figure 2.5: Generic 4×4 NoC architecture.

2.4. Memory Gap and Bandwidth Wall

Due to the throughput and performance gain with increasing the number of cores, CMPs will be able to alleviate of many barriers of single core processors. Therefore, with increasing the number of cores in CMPs, designers aim to extract performance and throughput at each generation. The exponentially increasing of core counts is confronted with the obstacles imposed by some performance-critical components that are less scalable. One of these obstacles is Memory Wall which presents a huge performance bottleneck for CMPs. The memory wall challenge restricts the CMPs performance with both long memory latencies and limited memory bandwidth.

In the trend of technology progress, processors' operating frequencies were scaled much faster than Dynamic Random-Access Memory (DRAM) that dominates the main memory system. The difference in speeds of the on-chip and off-chip memories pushed the chip designers to increase the on-chip memory and memory systems. However, growth in the number of cores and the amount of storage components in CMPs consequents a corresponding increase of off-chip memory traffic.

There are several factors lagging the memory bandwidth scaling from transistor density scaling described by Moore's Law such as power constraints, pin-limitations, and packaging costs [32].

According to ITRS [36], the number of on-chip cores is expected to be doubled every 18 months while the total number of pins will be increased by about 10% per year. The fundamental result shows that the rate at that memory traffic is generated by an increasing number of cores is growing faster than the rate that it can be serviced.

In these days, designers encounter to the bandwidth wall in addition to memory and power wall problems. Therefore, the total performance and throughput of the system are increasingly limited regarding to the available amount of off-chip bandwidth. In case the provided off-chip memory bandwidth is unable to maintain the rate of requested memory, the performance of the cores will decline due to the extra queuing delay for memory requests. The performance of the cores will have to match the available off-chip bandwidth. In this context, providing more number of cores is not efficient any more to improve the throughput or performance of the chip. The ideal solution is to increase the on-chip memory with more energy efficient architecture.

Several approaches have been investigated to mitigate the memory bandwidth, such as integrating the whole system on a chip (SoC) increasing the performance within the limits of power density. Memory compression is another approach which increases the memory size without increasing their area reducing the off-chip traffic [41].

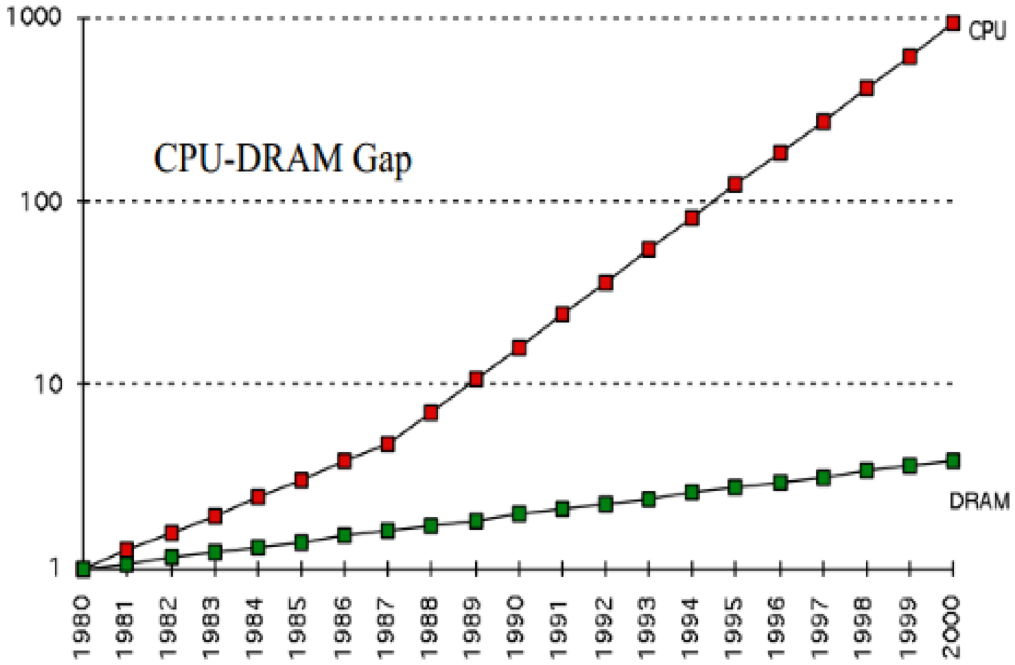


Figure 2.6: Gap growth between processor and memory performance.

2.5. On-Chip Memory

In the early decades of computing, memory systems were extremely slow and expensive. Central Processing Units (CPU) were not particularly fast either. Starting from the eighties, the gap between processors and memory components began to widen rapidly. Processor clock speeds took off, leaving memory access times lagging far behind as shown in Figure 2.6 [34]. As it shown in the figure that a new generation of fast memory is needed in order to bridge the gap.

As far as cashes architecture, processor cashes, which are identified as small pools of memory, store data that will be probably needed next by the processor. Sophisticated algorithms as well as certain assumptions related to programming code determine the amount of data loaded in the cache. As a matter of fact, the memory system ensures that the needed amount of data loaded into the memory is fully supplied to the processor. Once the processor unit finds required data, it is called memory hit. A memory miss, however, refers to the process of the CPU rushing off to find the data somewhere else, where the L2 comes into play. L2 represents a different type of on-chip caches which is slower, yet larger than L1. If the processor fails to find the data in either L1 or L2, it continues the process until it reaches the main memory (DRAM). It is noted that modern L1 caches have hit a higher rating than the 50% which is theoretically suggested in this paper. Both AMD and Intel hit a typical field cache rating of 95% [34].

The cache hierarchy including of small and fast L1, larger and slower L2 and most larger and most slower L3 comes to increase the memory hit rate and reduce the off-chip memory access events. Figure 2.7, shows a memory hierarchy in a multicore system [45].

In order to estimate the memory's hit rate importance, a difference of 2% in the L1 cache hit rate may double the total time for code execution [45]. The reasons for the differences in memory speed and size are traced to the design tradeoff between the memory performance with memory high cost of power and chip area. In CMPs, each PE has its own L1 and L2, and all PEs share the same Last Level Cache (LLC).

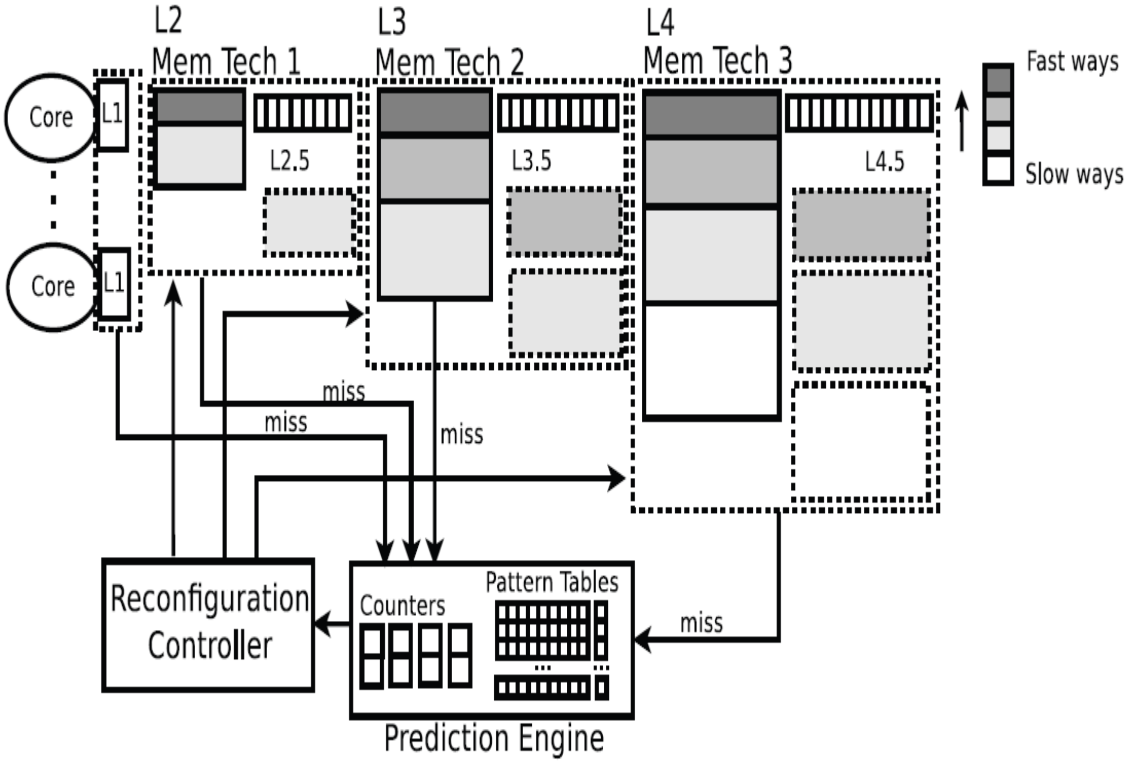


Figure 2.7: A memory hierarchy in a multicore processor.

2.5.1. Heterogenous On-Chip Architecture

As explained in [34], on-chip memory hierarchy bridges the gap between the processing units and the main Random-Access Memory (RAM). Shared LLC is the slowest and largest part of the on-chip memory as they are fabricated from smaller transistors to reduce the cost of power and chip area. Static RAM (SRAM) and DRAM are dominant as memory technologies in the face of arising scalability problems resulted from the limitations of their device cell size and power dissipation. Increasing the leakage power of SRAM and DRAM and the refresh power of DRAM contributes significantly to the overall system energy [35]. To enhance the 3D CMPs performance, an urgent need for an energy-efficient memory subsystem is inevitable. Figure 2.8 [47], illustrates a heterogenous memory architecture in a many-core system.

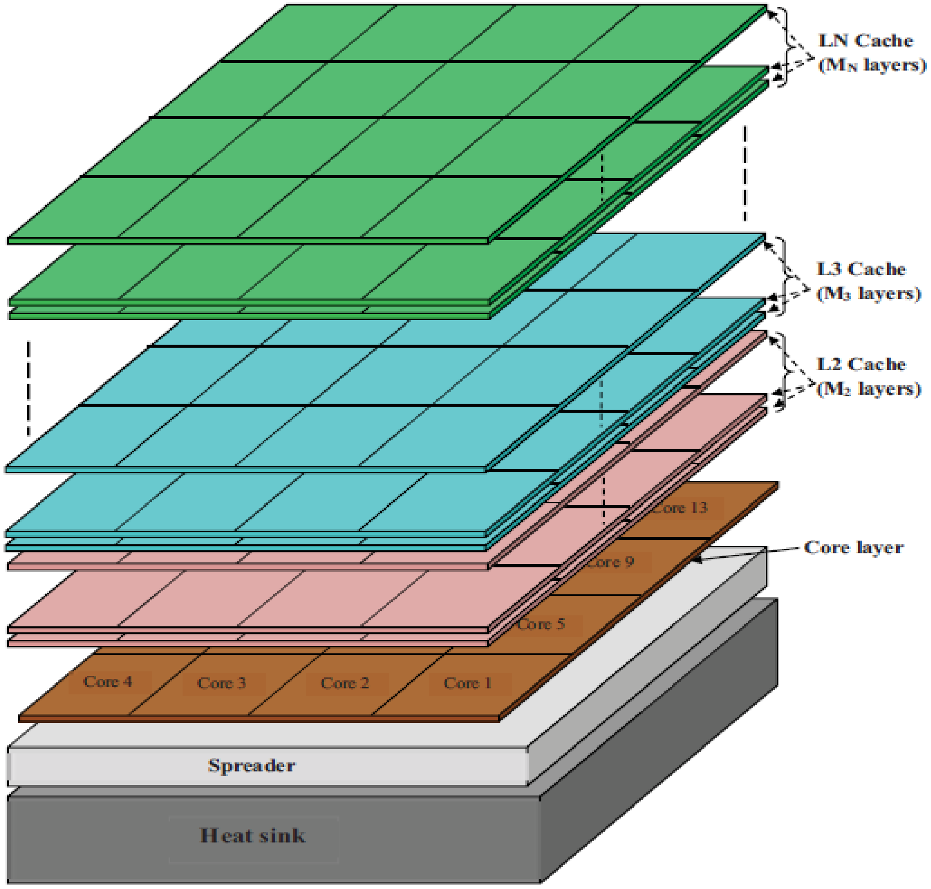


Figure 2.8: A heterogenous memory hierarchy in a many-core system.

2.6. Two to the Three-Dimensional CMPs

Larger number of PEs increases the distance among these PEs and the shared LLC, which causes latency problem in the system performance. The 3D integration technology is an emerging solution to solve the increasing on-chip communication latency and power consumption. In 3D CMPs, stacked layers shorten the distance between the horizontal layers with TSVs. TSVs facilitate stacking another active layer on the 3D layer. Utilizing the stacked layers of on-chip memory components in 3D CMPs is popular approach to tackle the memory bandwidth challenge and energy consumption [42]. Figure 2.9, shows a 3D chip as an example of 4×4×2 architecture, each core has its own network interface [42].

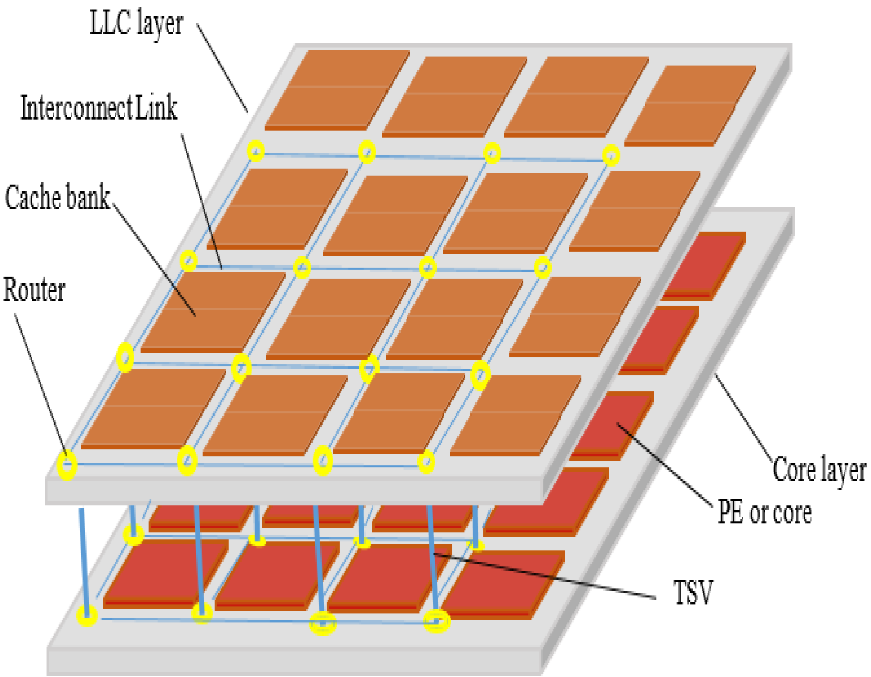


Figure 2.9: The 4×4×2 3D CMPs architecture.

In addition to the previous advantages, 3D architectures increase the cost efficiency and specifically, some facilitate such as mixed-technology stacking as special process required to fabricate Non-volatile Memories (NVMs) with conventional CMOS circuits [43]. The 3D CMPs is a natural evolution to the 2D CMP, when routers will have north and south ways in addition to the previous directions.

In the 3D CMPs, the thermal challenge has been exacerbated for two reasons. Firstly, one bringing more active components closer together, increasing the power density and hotspot generation. Secondly, adding multi-layers increase the primary thermal paths to the heatsink and to the circuit board. As the memory miss, reliability, and lifetime all are the function of temperature, thermal management has become the most crucial aspect in the 3D CMPs.

2.7. Trends on Power Aware CMPs

CMPs create the issues previously mentioned, such as power and memory walls, transistors variation, and many manufacturing issues. Power consumption is the most important part as it has direct relation to the reliability and performance of the CMPs. This section lists several design methodologies on power and thermal aware CMPs.

2.7.1. Dynamic Voltage Frequency Scaling

Dark silicon sets a limited power budget that cannot be exceeded by CMPs. If the operating system requires more resources to schedule parallel, power consumption must be scaled down to bring the required resources from the dark. Dynamic power consumption of a CMPs part or component can be calculated using Equation 2.2.

$$P_{dyn} = \propto . C . {V_{dd}}^{2} . f \tag{2.2}$$

Whereas (α) is the activity factor of inputs, (C) is the capacitance related to the area, (V_{dd}) the voltage of power supplier, and (f) the frequency.

These parameters are convenient parameters to reduce dynamic power consumption. An emerging challenge in the nano-scale region is the leakage in power consumption. It is getting worse with reducing technology size and again it is related to the area and voltage (V). Voltage has quadratic effect on the power consumption interrelated with frequency. With scaling the voltage, transistors will work slower and the frequency must be adjusted to meet the timing constraint.

The main objective of Dynamic Voltage Frequency Scaling (DVFS), most commonly used as power management technique, is to adjust the power consumption in multi-core processors. Power gating or switching off unused CMPs components can be regarded as extra case of DVFS. DVFS needs an accurate monitor of the system performance that can scale accordingly without performance degradation. If the chip temperature exceeds a threshold limit, DVFS will force the chip to scale its power consumption to prevent a non-reversible chip damage regardless on the impact on the performance. DVFS and its requirement is elaborated in the following chapter.

2.7.2. Task Migration

3D CMPs increase the thermal paths from the active cores or memory bank to the heatsink and circuit board where most of the cooling takes place. Variable operating voltages/ frequencies and workloads can have an impact on local temperature differences. This difference in temperature is called thermal hotspots and it can reach up to 80℃. Excessive thermal hotspots lead to performance degradation, increased cooling cost, and reduced reliability for 3D CMPs [44].

Task migration [46] is a preventive technique to redistribute tasks over the CMPs cores aiming more at normal thermal distribution, as shown in Figure 2.10 [46]. For instant, the task migration technique can be conducted randomly in predefined time intervals or is being actuated when a preset threshold temperature is reached by one of the cores. It can be carried out by swapping the hottest core with a coldest one or more sophisticated method can be used. The implication of task migration includes several factors such as the overhead calculation, latency caused due to the changes in the wire distances. Task migration is more cost-efficient than the global DVFS or other techniques when any chip temperature sensor detects the maximum allowable temperature.

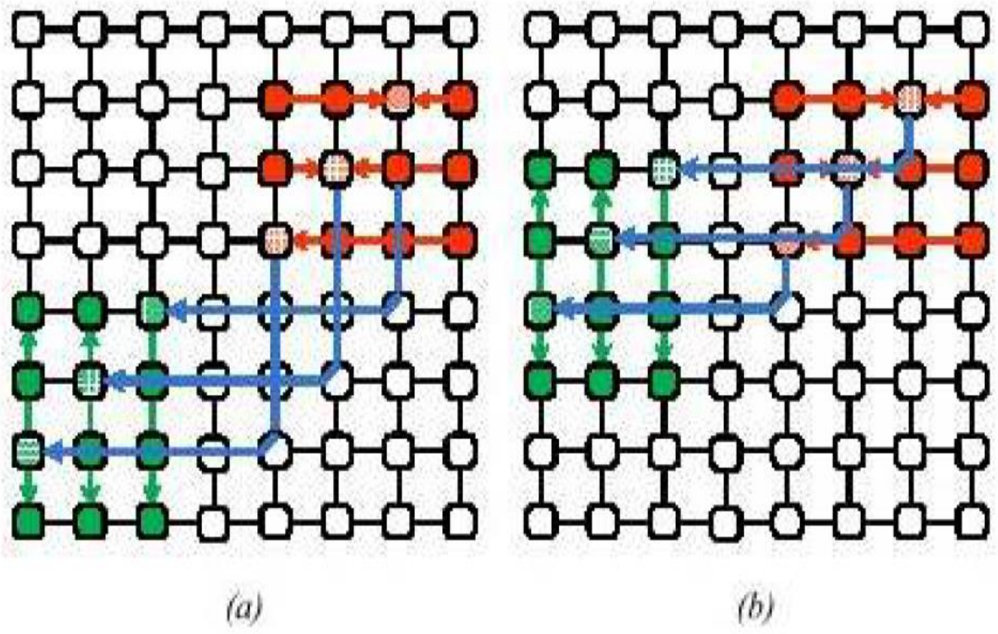


Figure 2.10: An overall view of a task migration technique, the green application in part (a) goes to up in part (b) based on a migration algorithm.

2.8. Chapter Summary

In this chapter, we presented an overview of on chip multi-processor architecture and useful power management techniques utilized in those platforms. A complete description of these methods is provided in the following chapter. Uncore components such as memory systems and on-chip interconnection as well as their challenges were discussed in this chapter.

Chapter 3: Literature Review

3.1. Introduction

A brief review about the RTM techniques which are used to keep the CMPs working state below the maximum temperature and power constraints will be brought up in this chapter. Maximizing performance under thermal and power constraints is the main objective of the proposed RTM techniques in the CMPs. Nowadays, off-line methods are not effective anymore when hundreds of cores are integrated on advanced CMPs. Thereby, on-chip RTM techniques are required to face thermal issues in modern processors. Moreover, with the growing number of cores in CMPs, a rapid increase in uncore resources, which consequently makes the uncore power, is also a critical part of the CMPs power managements.

Researchers have implemented different multicore thermal management techniques such as: DVFS [14-24], clock gating [8,10], task scheduling [1, 2], on-chip interconnection network [11,13], and task migration techniques [3-5, 9, 25, 26].

3.2. Dynamic Voltage Frequency Scaling

The Dynamic Voltage Frequency Scaling (DVFS) is a hardware facility that adjusts the clock frequency according to the operational voltage of a processor in real time [14]. DVFS is an effective technique for increasing the energy efficiency of a multicore processor by varying the voltage and frequency based on execution condition. The key idea in this technique is to satisfy the system performance and power consumption based on constraints of the system to process the assigned workload. DVFS is implemented by a transition of the processor to a low-power state from a high-power state (or vice versa) when the workload's condition changes [14,15]. DVFS can be applied to each core [14,15] or to separate domains of Voltage Frequency Island (VFI) in CMPs [16].

Authors in [17] applied Dynamic Voltage Scaling (DVS) and regulates the voltage of individual NoCs links independently to save power during periods of link under-utilization. The experimental results show that the proposed technique achieved 4.3X power saving (3.2X in average) with 27.4% latency increase and 2,5% throughput reduction.

A history-based DVS policy is introduced in [18], which judiciously adjusts link frequencies and voltages regarding to the previous usage. In [18] power saving was maximized when the scaling included the CPUs and links together, but this work was for specific applications. Gathering information from the all cores, network status, routers, and other resources was the major challenge when applying DVFS in the new generation of processors. The simulation results illustrate that the proposed method brought 13% and

17% energy savings over the pure CPU and pure communication links voltage scaling schemes, respectively.

Combining CPUs-Link DVFS with task scheduling of embedded systems is discussed in [19]. A rule based DVFS has been proposed to control each V/F island of the CMPs according to the queue occupancy in [20]. One of the latest works is comparing and extending a CMPs with adding the link utilization to the router queue occupancy in [21], this achieves 36 % savings in Energy-Delay Product (EDP) as opposed to the DVS policy based on link utilization only.

There are few previous works addressing DVFS for memories, one of the earliest works was in [16]. Memories were partitioned as VFIs similar to the other components (cores or other components) in the previous work and DVFS was applied according to the selected policy. The experimental results achieved 27% NoC and LLC reduction of energy with 7.3% degradation on the Average Memory Access Time (AMAT).

According to the work performed, connecting different operating voltage frequency domains (islands or individuals) necessitates special interfacing arrangements. The interface overhead accumulates with the other overheads resulted from the gathering required information from these domains and computation the V/F according the selected policy. Thus, power management might consume a comparable amount to what is meant to achieve in terms of energy saving and performance degradation. To mitigate the heterogeneity of the uncore voltage frequency, this area is assigned in a single V/F domain separate from the cores ones [16, 22].

The later work was developed in two aspects in [23] for both of the used rules and techniques for the DVFS scaling. AMAT monitors DVFS impact on the overall system performance, in contrast to the injection rate which is a good indicator for the network congestion. The second aspect of the [23] work is that it controls the DVFS using a Proportional and Integral (PI) controller to avoid any ensure smooth impact on the system performance. Another enhancement of [23] is improving the critical latency concept for the product of LLC throughput demand, the latency of the LLC and NoC, as an expression of uncore benefit. This formulation guarantees significant energy savings, more than any prior work to-date.

In [20], authors analyzed trade-off of the power-delay in a NoC under three DVFS policies. Authors applied a PI technique on a CMPs to propose coordinated, distributed DVFS in contrast with local ones. Furthermore, this work achieved EDP improvement of 85.5% from an off–line algorithm [20].

3.3. Clock Gating

Clock gating is a technique used to manage the temperature in CMPs. The hotspot cores are gated – hence the term clock gating – when the thermal threshold has been reached to prevent the occurrence of hotspots and, thus, system failure. It is worth mentioning that when a CMPs system cores run at the highest default frequency and voltage setting, the system will always reach the thermal threshold. It shows that clock gating is an effecting technique in CMPs energy efficiency.

Authors in [8], proposed a hybrid-method that coordinates clock gating and software thermal management techniques such as temperature aware priority management. Experiment results show that the proposed technique has the ability to manage the overall temperature with an average execution time overhead of only 9.9% while it is 24.4 % without any RTM.

Researchers in [10] also proposed a clock gating technique. The process goes as follows: each core runs at the highest frequency and voltage setting while the core has not reached the thermal threshold. Once the core has reached the thermal threshold and has become a hotspot, it stops running and its clock is gated to reduce power consumption. Afterwards, the process is resumed in the next sampling interval once the core temperature has gone below the threshold. This technique achieves 2.5 times throughput improvement and yields an average of 2.6 times speedup improvement compared to the Baseline across all workloads.

3.4. Task Scheduling

In this section, another methodology of performing thermal-aware on the 3D CMPs which is thermal aware task scheduling techniques is discussed. There has recently been a growing interest in Operating Systems (OS) which assisted task scheduling techniques in the CMPs to alleviate the thermal condition [1,2]. It is worth to mention that when the system implements OS-assisted task scheduling techniques, the hardware modifications are not required.

Based on an OS-assisted technique, to keep chip’s temperature low, authors in [1] ensured maximum thermal variations within different tasks, and then, they reschedule tasks accordingly. In this proposed technique, authors also consider the high thermal correlations among layers in one core stack and schedules tasks in bundles.

Subsequently, heavy-loaded tasks are allocated to the closest layers to the heatsink within every stack of cores for the purpose of dissipating heat problems. Therefore, based on a thermal emergency, power scaling is engaged to the core that generates the largest power in this stack or to the core whose temperature exceeds the threshold. This procedure aims at quickly cooling down the core stack, reducing the performance degradation resulted by high temperature. According to this method, vertically adjacent dies have strong thermal correlations, the proposed scheduler considers them jointly in this work [1]. This proposed technique can improve performance by 7.2% over the base processor.

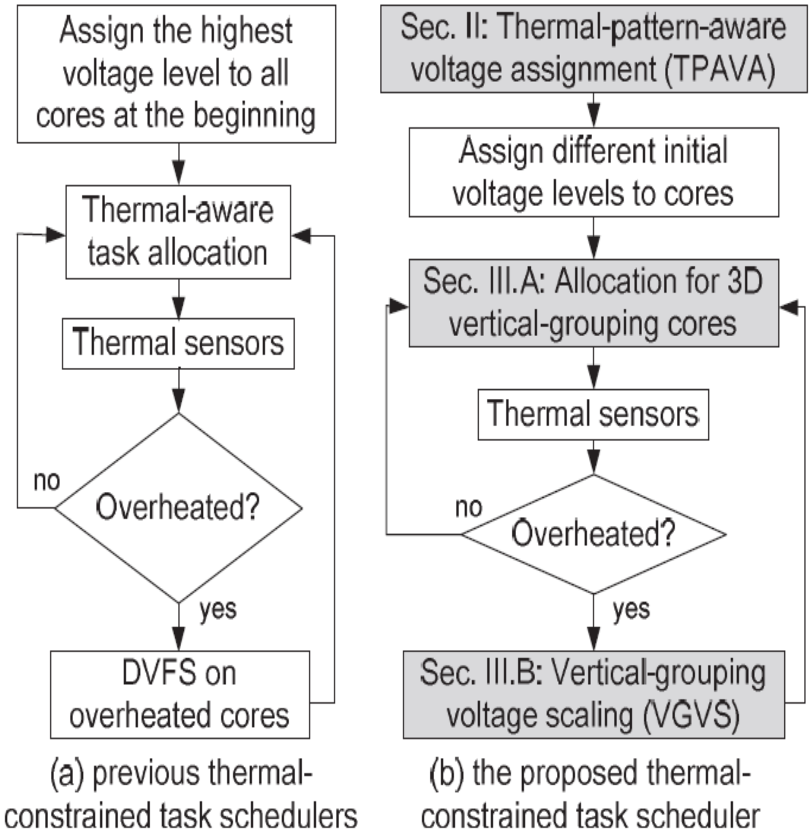

Assign the highest voltage level to all cores at the beginning
Thermal-aware task allocation
Thermal sensors
no
Overheated?
yes
DVFS on overheated cores
(a) previous thermal-constrained task schedulers
Sec. II: Thermal-pattern-aware voltage assignment (TPAVA)
Assign different initial voltage levels to cores
Sec. III.A: Allocation for 3D vertical-grouping cores
Thermal sensors
no
Overheated?
yes
Sec. III.B: Vertical-grouping voltage scaling (VGVS)
(b) the proposed thermal-constrained task scheduler

Figure 3.1: An overview of the proposed allocation strategy in [2].

In [2], a new thermal-constrained task scheduler based on Thermal-Pattern-Aware Voltage Assignment (TPAVA) has been proposed. By analyzing different voltage assignments of different temperature profiles, TPAVA assign different operating-voltage levels to cores as a preventative precaution to reduce the temperature increase in 3D processors. Moreover, the proposed task scheduler incorporates a Vertical-Grouping Voltage Scaling (VGVS) strategy that takes into consideration a thermal correlation in 3D processors as shown in Figure 3.1 [2]. In this work, results proved that the proposed scheduler technique can reduce the occurrence of hotspot by about 47.13% and improve throughput by about 6.5% respectively.

3.5. Interconnection Thermal Management

On-chip interconnection has a big contribution on total power consumption in the CMPs. Some papers proposed power and thermal aware on-chip interconnection architectures to distribute temperature uniformly on the chip [11,13].

In [11], authors proposed a hybrid buffer for on-chip routers to reduce power and hotspots in many core systems. This proposed buffer is made based on SRAM and Spin-Transfer Torque RAM (STT-RAM) technologies. Based on recent studies, traditional SRAM technologies are high speed and high-power consumer. In comparison, NVM technologies, which STT-RAMs are a popular representative of them, are slower and consume less power [14,15].

Based on the on-chip traffic pattern, a low overhead controller has been designed in [11] that selects between two options, SRAM state and STT-RAM state. When traffic is in the highest state, the designed controller selects the high-speed SRAM part of the buffer and when the traffic is in the lowest state, the designed controller selects the low power STT-RAM part of the buffer. The proposed technique achieves up to 30.9% network energy saving, 20.3% energy-delay product (EDP) reduction, and 7.6% router area decrease compared with the Baseline SRAM-based NoC design.

In [13], authors proposed a sprinting technique to bypass the unused interconnection links and routers by mapped applications during their execution time turning off the unused links and routers and, thereby, saving power consumption.

3.6. Task Migration

To achieve balanced thermal profile, task migration-based frameworks have been proposed. This balance is achieved by proactively migrating tasks among cores on the chip to distribute thermal management.

Once the system decided to apply the migration technique between the hottest core and the coldest core, it will exchange the thread on the selected coldest core with the thread on the hottest core. The task migration mechanism assumes that the whole code and data of the tasks will be exchanged from the hottest core to the selected coldest core. After that, the system should stop the running tasks and proceed to the task migration.

Authors in [3,4], proposed a thermal management technique based on the centralized controller for many core systems. Each core in this proposed method has an agent responsible for monitoring the core temperature, communicating, and negotiating with neighboring agents, so tasks can be migrated and distributed among cores accordingly uniformly across the system. The migration policy used in [3,4] distributes different tasks in a neighborhood based on their heat dissipation ability. This migration policy ensures a balanced temperature control on processors in a neighborhood. In that regards, a neural network-based anticipation model was proposed to predict future temperature, and for agents to evaluate the rewards of proposed migration offers. Finally, they showed that their proposed method reduces 29.8% hotspots and 80.68% migration overhead with only 0.98% performance overhead compared to a Baseline thermal management.

In [5], to efficiently guide local migration among cores, researchers have proposed a distributed stack tracking technique so they could proactively estimate the average temperature of cores on the chip which, in turns, determines a thorough view of the temperature of the chip as a whole. The experimental results in [5] show that in a 36-core chip multiprocessor, the proposed method works more efficiently reducing the number of thermal hotspots (30% more thermal hotspots reduction compared to the existing distributed thermal management methods).

In [25], authors suggest a new distributed thermal management method to reduce on-chip temperature variance, and to prevent the occurrences of hotspots for many-core processors. This scheme uses task migrations based on a novel temperature metric known as effective initial temperature.

As shown in Figure 3.2 [25], the task migration policy will come into play once the temperature of the current core reaches a certain threshold temperature T_{th}. The current core communicates with its adjacent cores to check on whether or not migration criteria (1) and (2) are met. The flow starts with *core a* (the upper left adjacent core). If the criteria are met, *core a* is assigned as the target core for task migration. The thermal and load parameters of the current core (*$T_{eff, cur}$*, *P_{cur}*) are updated by the thermal and load parameters of *core a* (*$T_{eff, a}$*, *P_a*). Then, the flow continues to check with *core b*, if the migration criteria are met. It indicates that *core b* is a better migration option with even lower (*$T_{eff, b}$*, *P_b*) than the previous migration selections of the destined core, and thus will be selected to replace the previously chosen one as the destined core for task migration. The same process will be repeated from *core c* to *core h*. The flow continues to update the selection and will finally choose the core with the lowest value of effective initial temperature (*T_{eff}*) and load (*P*) for task migration among all eight adjacent cores. On a 100-core microprocessor, simulation results of this work show that the proposed technique reduced the number of thermal hotspots (illustrate up 21% reduction on hotspots compared with the alternative approach, and 44% reduction on hotspots compared with the existing distributed thermal management methods).

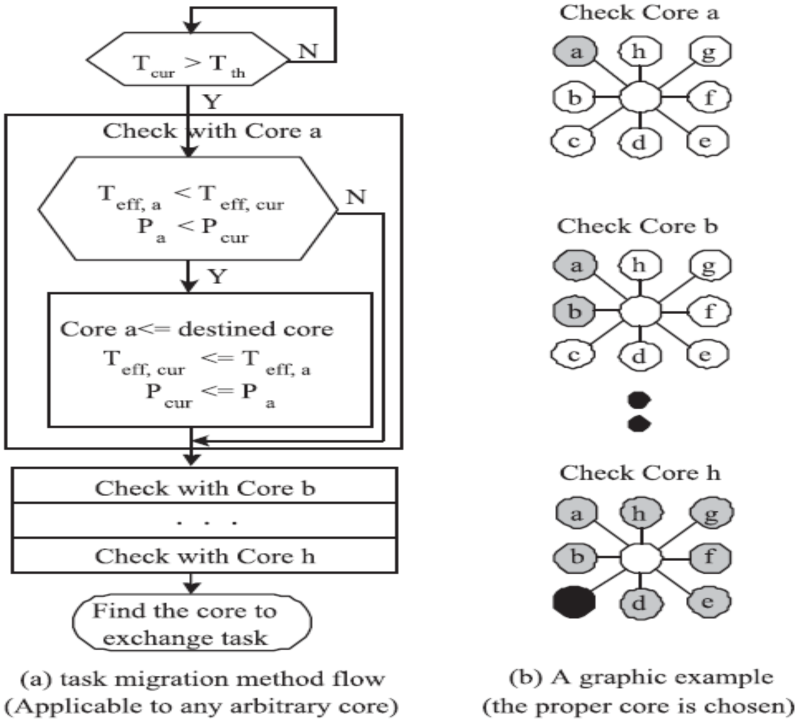


Figure 3.2: The proposed task migration technique in [25].

In [26], researchers proposed a run-time distributed migration algorithm based on game theory to control arising heat issues among PEs in a NoC-based 3D CMPs. Minimizing the peak temperature of the 3D NoC, as well as the overhead imposed on chip performance during migration is the main objective of this algorithm. Due to high thermal interconnection between the adjacent PEs in the same stack in 3D NoC-based CMPs, they model a problem with multi objectives as a cooperative game. Simulation results of this work illustrate up to 23% and 27% reduction in peak temperature, as far as the benchmarks with the highest communication rate and the largest number of tasks are concerned.

3.7. Chapter Summary

In this chapter, we have explored different multicore thermal management techniques in some of the lately researches in RTM techniques such as DVFS, clock gating, task scheduling, on-chip interconnection networks, and task migration techniques in detail.

Chapter 4: The First Proposed Method: A Migration Technique to Balance Thermal Distribution for 3D Chip Multiprocessors

4.1. Introduction

Due to the constant shrinking in the size of the transistor, embedded systems industry continued to increase the number of transistors that are integrated in a single chip. Moreover, CMPs trend moves from 2D CMPs to 3D CMPs architecture as mentioned in the previous chapters. Even though the 3D CMPs architecture is designed to overcome the problems related to the length of the interconnects and power consumption, a new challenge arises, namely thermal hotspot. To overcome these challenges, applying RTM has become crucial.

In this chapter, the proposed technique aims to balance hotspots and temperature variations on the 3D CMPs. In fact, migrating a hotspot to a low temperature core in the core layer to control the temperature variances without considering hotspots in the stacked memory layer has the potential to cause the emergence of new hotspots. Therefore, the proposed technique considers hotspots effects in both core and memory layers simultaneously for equilibrium hotspots on the 3D CMPs.

The proposed technique provides two algorithms that are working in parallel. Firstly, algorithm I aims to distribute hotspot cores that are placed on the central part of the core layer to cores on the surrounding part by applying run-time task migration technique. Secondly, algorithm II is responsible for migrating the most accessed DRAM banks to the central part of the memory layer in order to be close to all cores in the core layer and to make a balanced 3D CMPs temperature.

The proposed method aims at maximizing the throughput on the 3D CMPs while satisfying the peak temperature constraint. Furthermore, the proposed migration technique gathers the temperature of cores and DRAM banks by using performance counters and proposed equations instead of using thermal sensors.

The rest of this chapter is organized as follows: Section 4.2 provides an explanation of the target system architecture. In section 4.3, the proposed migration technique is explained. In section 4.4, experiments are performed, and results are evaluated. Finally, section 4.5 summarizes of the proposed run-time task migration technique.

4.2. The Target System Architecture

The target system architecture provides a 3D CMPs, which contains two layers, the core layer, and the memory layer as shown in Figure 4.1. The core layer contains 64 cores in which each core includes a core, a private L1 memory bank, and a shared L2 memory bank. Each core in the core layer has a performance counter to compute the Instruction Per Cycle (IPC) for measuring the power consumption. In the target 3D CMPs architecture, the core layer is divided into five regions. Those are: the central region ($C_{central}$), surrounding region 1 (S_1), surrounding region 2 (S_2), surrounding region 3 (S_3), and surrounding region 4 (S_4) as illustrated in Figure 4.2.

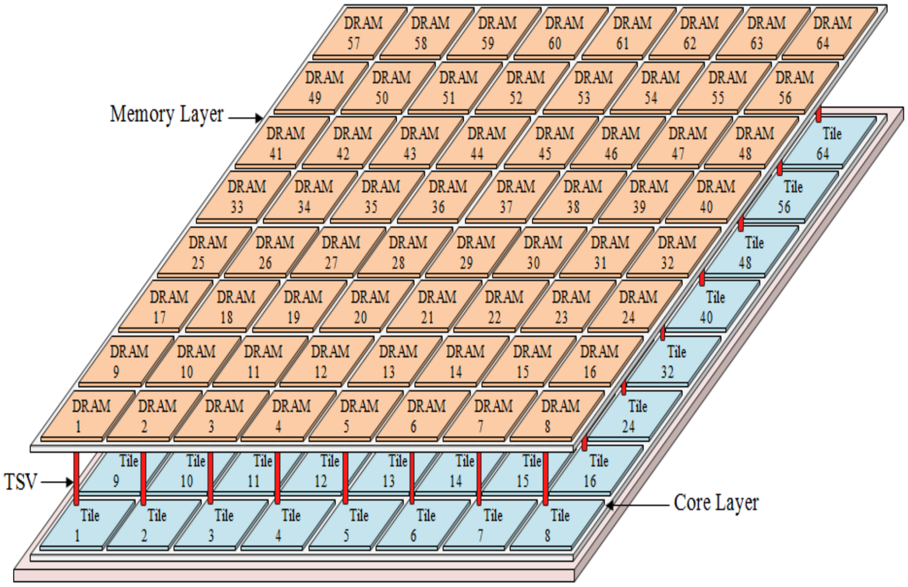
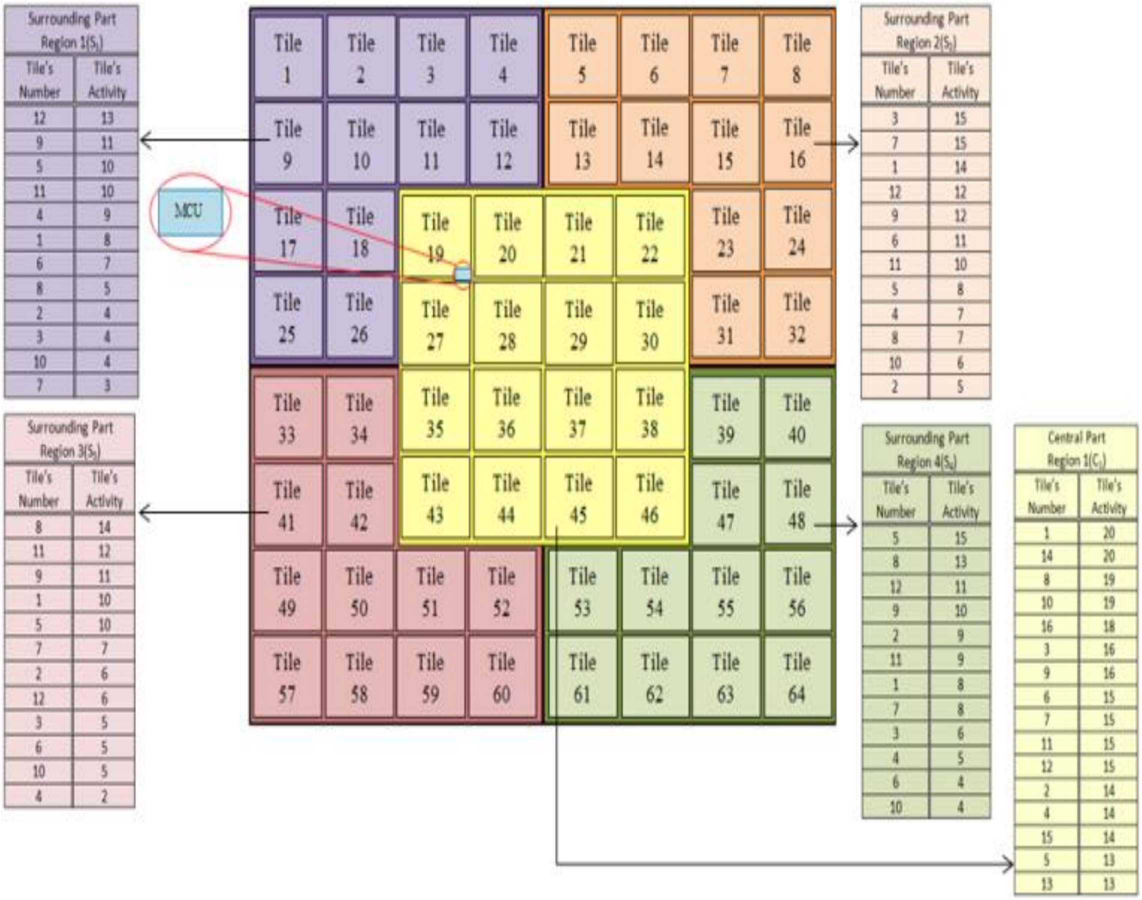


Figure 4.1: 3D CMPs architecture contains the core layer, and the memory layer.

On the other hand, the memory layer contains 64 DRAM banks where each DRAM bank has a performance counter to measure the access percentage level. Same as the down layer, the memory layer is divided into five regions; the central region

(*CD*$_{central}$), surrounding region 1 (*SD*$_1$), surrounding region 2 (*SD*$_2$), surrounding region 3 (*SD*$_3$), and surrounding region 4 (*SD*$_4$).

In fact, the five regions in each layer are presented to fulfill the achievement of the proposed task migration technique, which will be mentioned in detail later. Moreover, as can be seen in Figure 4.2, each region in the core layer has a table that contains the performance counter information of each core in the region and its location. Furthermore, in the memory layer, each region has a table that includes the accesses distribution (performance counter's information) of each DRAM bank and its location.



Surrounding Part Region 1(S_1)	
Tile's Number	Tile's Activity
12	13
9	11
5	10
11	10
4	9
1	8
6	7
8	5
2	4
3	4
10	4
7	3

Surrounding Part Region 2(S_2)	
Tile's Number	Tile's Activity
3	15
7	15
1	14
12	12
9	12
6	11
11	10
5	8
4	7
8	7
10	6
2	5

Surrounding Part Region 3(S_3)	
Tile's Number	Tile's Activity
8	14
11	12
9	11
1	10
5	10
7	7
2	6
12	6
3	5
6	5
10	5
4	2

Surrounding Part Region 4(S_4)	
Tile's Number	Tile's Activity
5	15
8	13
12	11
9	10
2	9
11	9
1	8
7	8
3	6
4	5
6	4
10	4

Central Part Region 1(C_1)	
Tile's Number	Tile's Activity
1	20
14	20
8	19
10	19
16	18
3	16
9	16
6	15
7	15
11	15
12	15
2	14
4	14
15	14
5	13
13	13

Figure 4.2: Core layer with 5 regions, the MCU connection with a core, and a sample of MCU (*C*$_{central,}$ *S*$_{1,}$ *S*$_{2,}$ *S*$_{3,}$ *S*$_4$) tables of each region.

In this context, performance counters play an important role to calculate the power consumption of each core and DRAM bank. For instance, in the core layer, it considers the IPC of each core for calculating power consumption. In this trend, in the memory layer, for calculating the power consumption of each DRAM bank, the access percentage level to each DRAM bank is considered. The information in the tables of each layer is updated at the end of each specified time interval which is fixed to 100ms in this work.

The dynamic power consumption of each core is calculated based on Equation 4.1 and for each DRAM bank based on Equation 4.2.

$$P_{Core} = IPC \times f \times C_L \times V_{DD}^2 \tag{4.1}$$

$$P_{Memory} = PC_{read} \times E_{read} + PC_{write} \times E_{write} \tag{4.2}$$

Where (P_{Core}) is the core's power consumption, (*IPC*) is the instruction per cycle which is the core activity, (f) is the core frequency, (C_L) is the average capacitance, and (V_{DD}) is supply voltage. Also, (P_{Memory}) is the DRAM's power consumption, (PC_{read}) is the performance counter of reading from a DRAM, (E_{read}) is the energy consumed to read, (PC_{write}) is the performance counter of writing on a DRAM, and (E_{write}) is the energy consumed to write. Since the frequency of each core/DRAM bank in the target 3D CMPs is constant and the DVFS technique in today's nano-scale designs is expensive and high-overhead, dynamically change in the frequencies are not assumed in the system.

As can be seen in Equation 4.1, the *IPC* has a key role in calculating the power consumption of each core in the 3D CMPs. In addition, *PC* is important to measure the power consumption of each DRAM bank in the 3D CMPs as shown in Equation 4.2. In this context, for measuring the *IPC* and *PC*, performance counters are embedded near each core and DRAM bank. For instance, Figure 5.2 shows an example of performance counters in the memory layer.

In fact, since heavy-loaded tasks consume larger power consumption, compared to the light-loaded tasks, the increased power consumption of a core or DRAM bank generates hotspots, which is the major concern in our proposed migration technique. The objective of this proposal is to minimize them as much as possible and achieve a balanced 3D CMPs temperature.

In this work, the proposed architecture provides a new decision-making unit which is a centralized hardware in the core layer named Migration Control Unit (MCU). MCU is the system decision-making unit that is responsible for RTM in the target 3D CMPs. MCU aims to make the optimal decision for applying the proposed run-time task migration technique. The MCU is placed near to all of the cores in the core layer as shown in Figure 4.2.

As mentioned before, each layer is divided into five regions and each region has a table. Therefore, it should be noted that the information on those tables in the MCU are updated at the end of each specified time interval. In this context, at the end of each time interval, the information gathered from performance counters, in the core layer and the memory layer, are sent to the MCU in order to make the final decision in the proposed algorithms. The presented tables are arranged according to the activity of each core/DRAM bank, which will be mentioned in detail later. The used hardware overhead in the target 3D CMPs architecture is shown in Table I.

TABLE I. HARDWARE OVERHEAD OF THE TARGET 3D CMPS ARCHITECTURE

The Hardware	**The Overhead**
Performance counters	128 × 32 bits
MCU	160 × 46 bits

4.3. The Proposed Method

The proposed migration technique in this section provides a run-time task migration technique between cores in the core layer and DRAM banks in the memory layer separately. The run-time task migration technique contains two algorithms that work in parallel.

Firstly, algorithm I in the core layer, holds the migration responsibility based on each core's activity in ($C_{central}, S_1, S_2, S_3, S_4$) tables. MCU applies the task migration technique in order to migrate the heavy-loaded tasks to cores in the surrounding regions, and the light-loaded tasks to cores in the central region.

Secondly, in the memory layer, according to the DRAM banks' accesses percentage level in ($CD_{central}, SD_1, SD_2, SD_3, SD_4$) tables, algorithm II applies the task migration technique for migrating the most accessed DRAM bank (the hottest DRAM banks) to the central region so it can be near to all cores that it communicates with and to achieve a balanced 3D CMPs temperature.

In the memory layer, in contrast to the core layer, the central part is the hot region and surrounding parts are cold regions. Therefore, the central hot region in the memory layer is stacked on the central cold region of the core layer, and the surrounding cold regions in the memory layer are stacked on the hot surrounding regions of the core layer.

Once the MCU has decided to proceed the migration, it will exchange the thread on the selected optimal coldest core with the thread on the hottest core. The proposed task migration mechanism assumes that the whole code and data of the tasks will be exchanged from the hottest core to the selected optimal coldest core. After that, the system should stop the running tasks and proceed to the task migration.

Based on this methodology, the temperature will be distributed uniformly on the upper layer of the proposed 3D CMPs. Furthermore, the proposed run-time task migration technique, including algorithms I and II are provided to achieve the balance thermal distribution at run-time for the 3D CMPs. The procedure of the proposed run-time task migration technique, including the algorithms I and II are explained in detail in sections 4.3.1 and 4.3.2 respectively.

4.3.1. The Task Migration Technique in the Core Layer

When MCU gathers the information of ($C_{central,}$ $S_{1,}$ $S_{2,}$ $S_{3,}$ S_4) tables at the end of each time interval, the MCU applies the proposed task migration technique in order to satisfy the balanced chip temperature as it shown in Algorithm I. Algorithm I provides the following steps:

- First, the MCU sorts the ($C_{central,}$ $S_{1,}$ $S_{2,}$ $S_{3,}$ S_4) tables in descending activity order (from the hottest to the coldest cores, assuming that the core with the highest activity is the hottest one).
- Second, MCU selects the hottest core ($HC_{central}$) in table ($C_{central}$) and selects the hottest core (HC_i) and the coldest core (DC_i) in each table among the tables ($S_{1,}$ $S_{2,}$ $S_{3,}$ S_4).
- Third, the MCU checks the (HC_i) in those tables; if any of the (HC_i) cores has reached the TH_S (the specified threshold which is 80), MCU exchanges the (HC_i) core with the coldest core (DC_i) in that region and removes the (DC_i) from the list of the coldest cores in surrounding regions. Otherwise, it provides the fourth step.
- Fourth, MCU selects the optimal coldest core (OPC) among the remaining coldest cores in (DC_i) list. In this procedure, if the (DC_i) list is empty, MCU selects the best second coldest core among coldest cores in (DC_i) list.
- Finally, it migrates the ($HC_{central}$) with the (OPC) core.

Algorithm I: The Applied Algorithm in the Core Layer.

```
Loop
Sort cores in central region ($C_{central}$) in a descending activity order
Select the hottest core ($HC_{central}$) in the ($C_{central}$) table
for ($1 \leq i \leq 4$)
    Sort cores in the ($S_i$) tables in a descending order
    Select the hottest cores ($HC_i$) in ($Si$) tables
    Select the coldest cores ($CD_i$) in ($Si$) tables
    If ($HC_i > TH_S$) then
        Exchange the $HC_i$ with the coldest core ($CD_i$) in that region
        Remove the ($CD_i$) from the list in the related surrounding region
    end if
end for
If ($CD_i$) list is empty then
    Select the second coldest $CD_i$ as an optimal coldest core ($OPC$)
else if
    Select the optimal coldest core ($OPC$) among the list of coldest cores.
end if
```

```
Migrate the ($HC_{central}$) with the ($OPC$)
Go to Loop
```

In other words, after sorting the cores from the highest temperature to the lowest temperature in both central region and surrounding regions by the MCU, the proposed Algorithm I migrates the hottest core in the central region ($HC_{central}$) with the optimal coldest core (OPC) in the surrounding regions ($S_{1,} S_{2,} S_{3,} S_4$).

In this context, the heavy-load tasks are migrated to the edges of the chip and in the central part the light-load tasks are migrated. It is considered that the chip’s edges are a better choice for the placement of the hot tasks than the central part as neighboring cores have a higher impact on the temperature of each core in the central part.

4.3.2. The Task Migration Technique in the Memory Layer

Algorithm II is responsible for applying the proposed migration technique in the memory layer. In fact, each core in the lower layer has a communication rate with DRAM banks in the upper layer. When a DRAM bank has more access percentage level compared to the other DRAM banks, which are not accessed frequently by cores, the most accessed DRAM banks should be placed in the central region of the memory layer in order to be near to all cores.

The higher performance counters of DRAM banks are of higher activity which leads to be a hotspot. Therefore, we assume that the most accessed DRAM banks are hotspots. Thus, each DRAM bank has a performance counter to measure its access percentage level, and then the most accessed DRAM banks are migrated to the center part of the memory layer. In this context, according to the DRAM banks’ access percentage level in ($CD_{central,}$ $SD_{1,}$ $SD_{2,}$ $SD_{3,}$ SD_4) tables, MCU applies algorithm II. Therefore, the main goal of the algorithm II is to migrate the most accessed DRAM banks (the hottest DRAM banks) to the central region of the memory layer in order to be near to all cores that communicate with and also to achieve a balanced 3D CMPs temperature.

The procedure of algorithm II is the same as the procedure in the core layer. However, in the memory layer the hotspots are distributed differently. For example, as mentioned before, in the memory layer, the central part is the hot region and surrounding parts are cold regions. Therefore, the central hot region in the memory layer is stacked on the central cold region of the core layer, and surrounding cold regions in the memory layer are stacked on the hot surrounding regions of the core layer.

When each core in the core layer accesses to each DRAM memory bank in the memory layer and this access is read, the read counter counts up, and it records the core's location (core coordinate in the core layer) in the look up table. Therefore, at the end of each time interval, the information of read counters and multiple use of look up table are read. If the read counter is more than 10 and the look up table has more than 2 entries, this memory bank goes to the Shared-Read-Only (SRO) bank. When MCU gathers the information of ($CD_{central,}$ $SD_{1,}$ $SD_{2,}$ $SD_{3,}$ SD_4) tables at the end of each time interval, the MCU applies Algorithm II which provides the following steps:

- First, sorts ($CD_{central,}$ $SD_{1,}$ $SD_{2,}$ $SD_{3,}$ SD_4) tables in a descending access order level.
- Second, select the coldest bank ($CB_{central}$) in the central region ($CD_{central}$).
- Third, select the hottest banks (HB_i) among surrounding regions ($SD_{1,}$ $SD_{2,}$ $SD_{3,}$ SD_4).

- Finally, exchange the ($CB_{central}$) in the central region with the (HB_i).

This policy leads to move the shared data to the central part of the memory layer, which has the optimal distance based on Manhattan Distance lemma in the memory layer for all cores in the core layer to communicate with. Therefore, the proposed method aims to place all the shared-read banks to the central part of the memory layer while the private-read and write banks are placed in the surrounding part of the memory layer automatically after Shared-Read-Only (SRO) banks are migrated.

Algorithm II: The Applied Algorithm in the Memory Layer.

```
Loop
Sort DRAM banks in ($CD_{central}, SD_1, SD_2, SD_3, SD_4$) in a descending access order level
Select the coldest bank ($CB_{central}$) in the ($CD_{central}$)
for ($i$=1 to $i \leq 4$)
    Select the hottest banks ($HB_i$) in the surrounding regions ($SD_1, SD_2, SD_3, SD_4$)
end for
Exchange the ($CB_{central}$) in the central region with the ($HB_i$) in surrounding region ($i$)
Go to Loop
```

In conclusion, the proposed algorithm, contains sections 4.3.1 and 4.3.2, prepare a balanced thermal distribution at run-time for the 3D CMPs. At the end of each time interval, the proposed algorithm, based on section 4.3.1, migrates the cold cores to the central part and hot cores to the surrounding part in the core layer. Also, based on section 4.3.2, the proposed algorithm migrates the most accessed memory banks to the central part in the memory layer.

Based on this methodology, the central region of the memory layer, which is the hottest region, is placed on top of the central part of the core layer, which is the coldest region in the core layer. Additionally, the surrounding part of the memory layer as a cold region is placed on top of the surrounding part of the core layer which is a hot region.

4.4. The Experimental Evaluation

In this section, 64 cores in the core layer and 64 DRAM banks in the memory layer of a 3D CMPs architecture with multi-threaded workloads were used to perform the proposed run-time task migration technique. First, to evaluate the proposed technique, the experimental setup is described. Second, in order to quantify the benefits of the proposed architecture compared to the traditional architecture, different experiments are performed.

4.4.1. The Platform Setup

We evaluate our CMPs architecture on a simulation platform built upon Gem5 [50] to run our experiments and measure the system performance improvements. GEM5 is a full system simulator to set up the basic system platform. The simulation platform is configured to model a 64-core CMPs. The 64-core architecture which formed 2D-mesh topology was modeled as shown in Figure 4.1. Traces from GEM5 were injected to 3D Noxim [51], a System-C simulator for 3D NoCs modeling the mesh-based on-chip interconnection network between cores and DRAM banks. Table II illustrates the detailed parameters used in the CMPs architecture

TABLE II. BASELINE CMPS CONFIGURATION.

The Component	The Description
Number of Cores	64, 8×8 mesh
Core Configuration	Alpha 21264, 1GHz, in-order, 14-stage pipeline, 45nm.
Private Memory per each Core	SRAM, 4 way, 64 line, size 64KB per core
On-chip Memory	Baseline: 64MB (64 DRAM banks, each of which has 512KB capacity) Proposed: 64MB (64 DRAM banks, each of which has 512KB capacity) + proposed migration policy.

In addition, PARSEC benchmarks [52] for multi-thread workloads were used in this context. The multithreaded applications with large working sets are selected from PARSEC benchmark suite [52]. These selected benchmark suits consist of emerging workloads suitable for the next generation shared-memory programs for the CMPs. For these benchmarks, one billion instructions were executed for the *simlarge* input that was set starting from the Region of Interest (ROI). Moreover, HotSpot [53] version 5.0 was employed as a grid-based thermal modeling tool for 3D temperature estimation. For the experimental evaluation, the maximum temperature limit $\mathrm{T}_{\mathrm{max}}$ was assumed to be 80℃.

4.4.2. Experimental Results

In this section, the 3D CMPs hierarchy were evaluated in two different cases on multithreaded workloads. Firstly, the 3D CMPs with DRAM banks in the memory layer stacked on the top of the core layer without any migration policy (*Baseline*). Secondly, the 3D CMPs with a DRAM memory layer stacked on top of the core layer with the proposed migration policy (*Proposed*).

Figure 4.3, shows the results of normalized energy consumption of each PARSEC application normalized with respect to the *Baseline* and *Proposed*. As shown in this figure, the *Proposed* architecture improves energy consumption by about 60% EDP, on average, compared with the *Baseline*. The best improvement is recorded for *canneal* with only 72% EDP from the *Baseline* energy consumption whereas the least reduction to the energy consumption is achieved by *swaptions* at only 43% EDP.

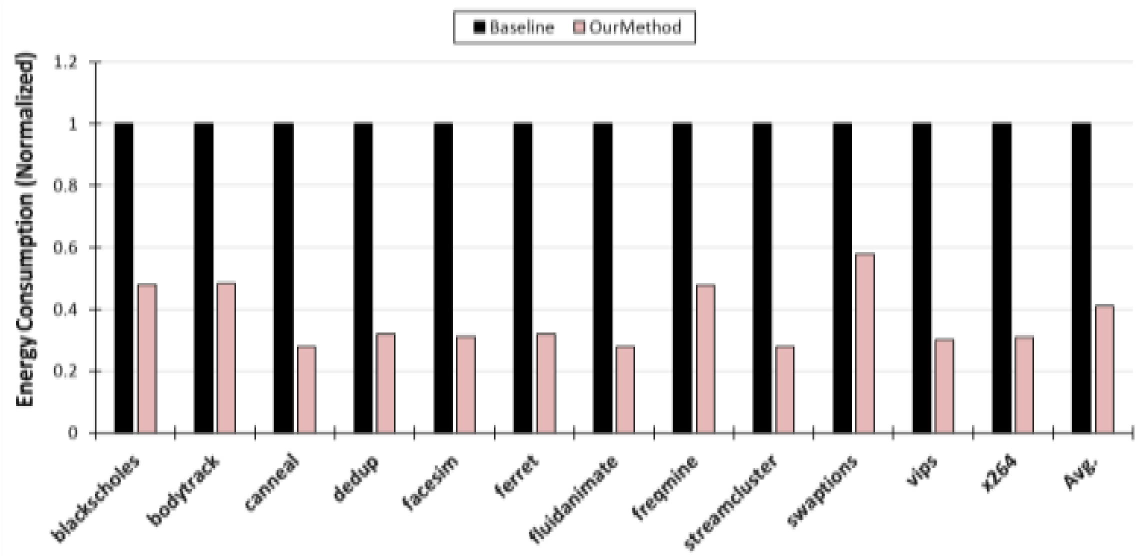


Figure 4.3: Energy comparison between the *Baseline* and *Proposed* architectures.

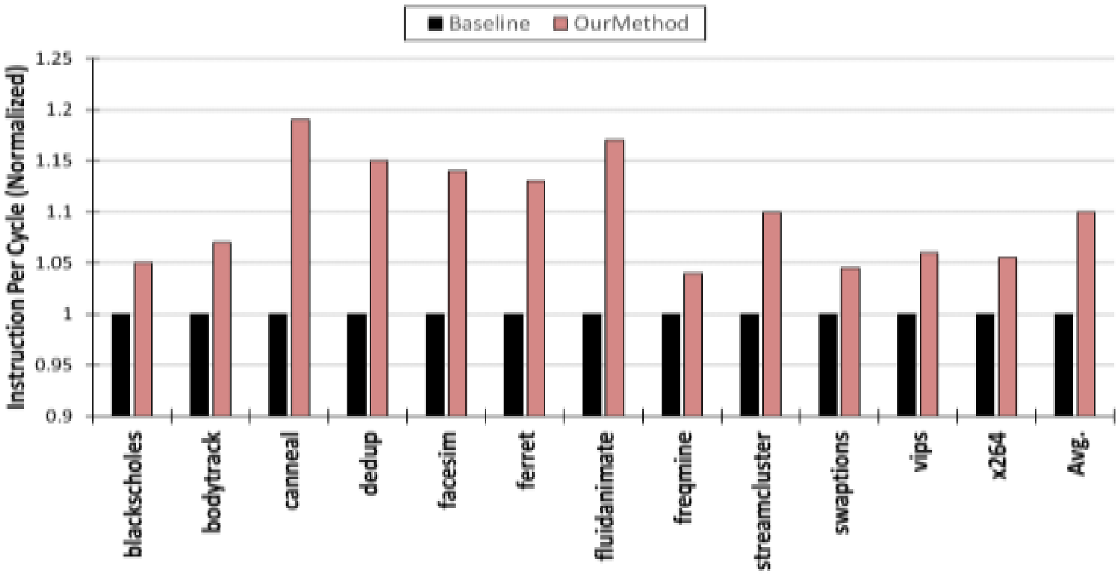


Figure 4.4: IPC comparison between the *Baseline* and *Proposed* architectures.

The difference between the energy enhancements as expected yields due to the differences between the natures of memory-intensive workloads. To investigate the impact of the proposed method on the performance, normalized IPC for each PARSEC application is illustrated in Figure 4.4. Figure 4.4 shows that the IPC in our methods are more compared to the *Baseline* in most of the used benchmark, 10%, on average and especially on *canneal*, *dedup*, *facesim*, *ferret*, *fluidanimate*, and *streamcluster*; which prove that the proposed method is faster. Interestingly, the proposed architecture improves the IPC for *canneal* and *fluidanimate* applications by 18% and 14%, respectively. Furthermore, as shown in Figure 4.5, the overhead of the proposed method is about 0.4% over the *Baseline* architecture.

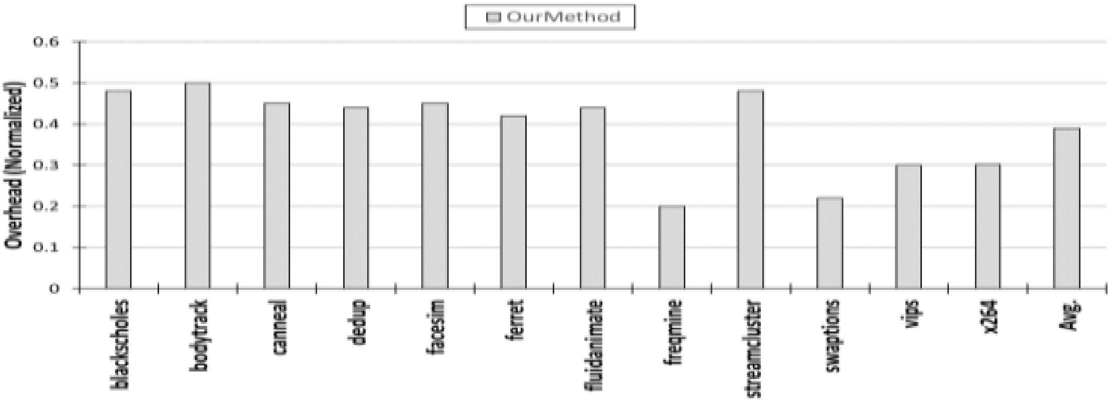


Figure 4.5: Comparison of overhead under the execution of PARSEC benchmarks.

In continue, we consider the execution delay results for the *Proposed* and *Baseline* architecture. As can be seen in Figure 4.6 the performance of the *Proposed* architecture is better than the *Baseline* architecture by 35% improvement on average.

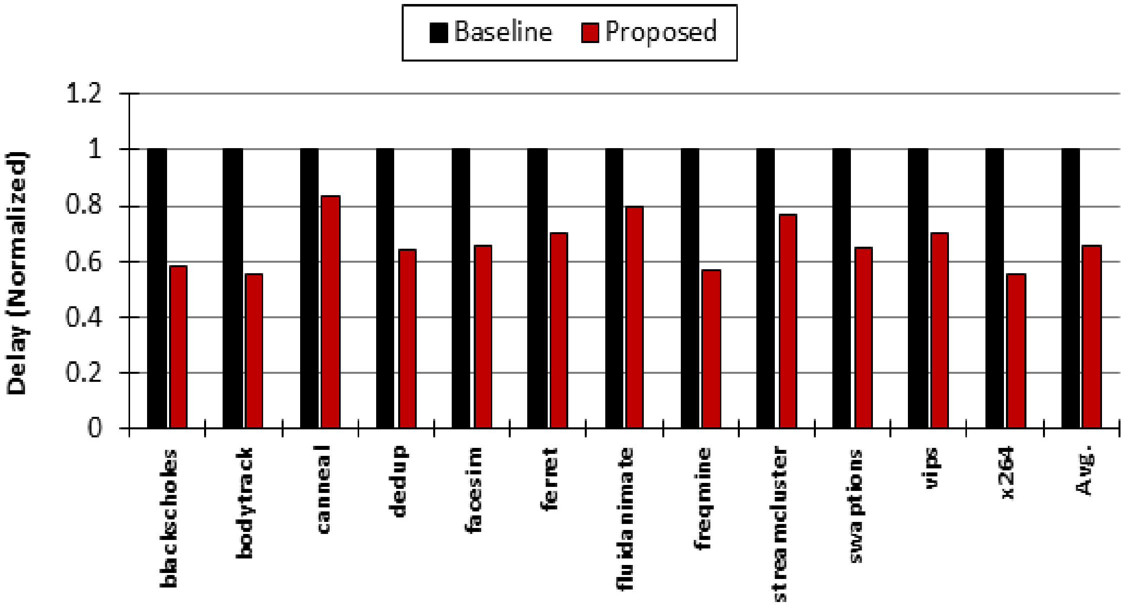
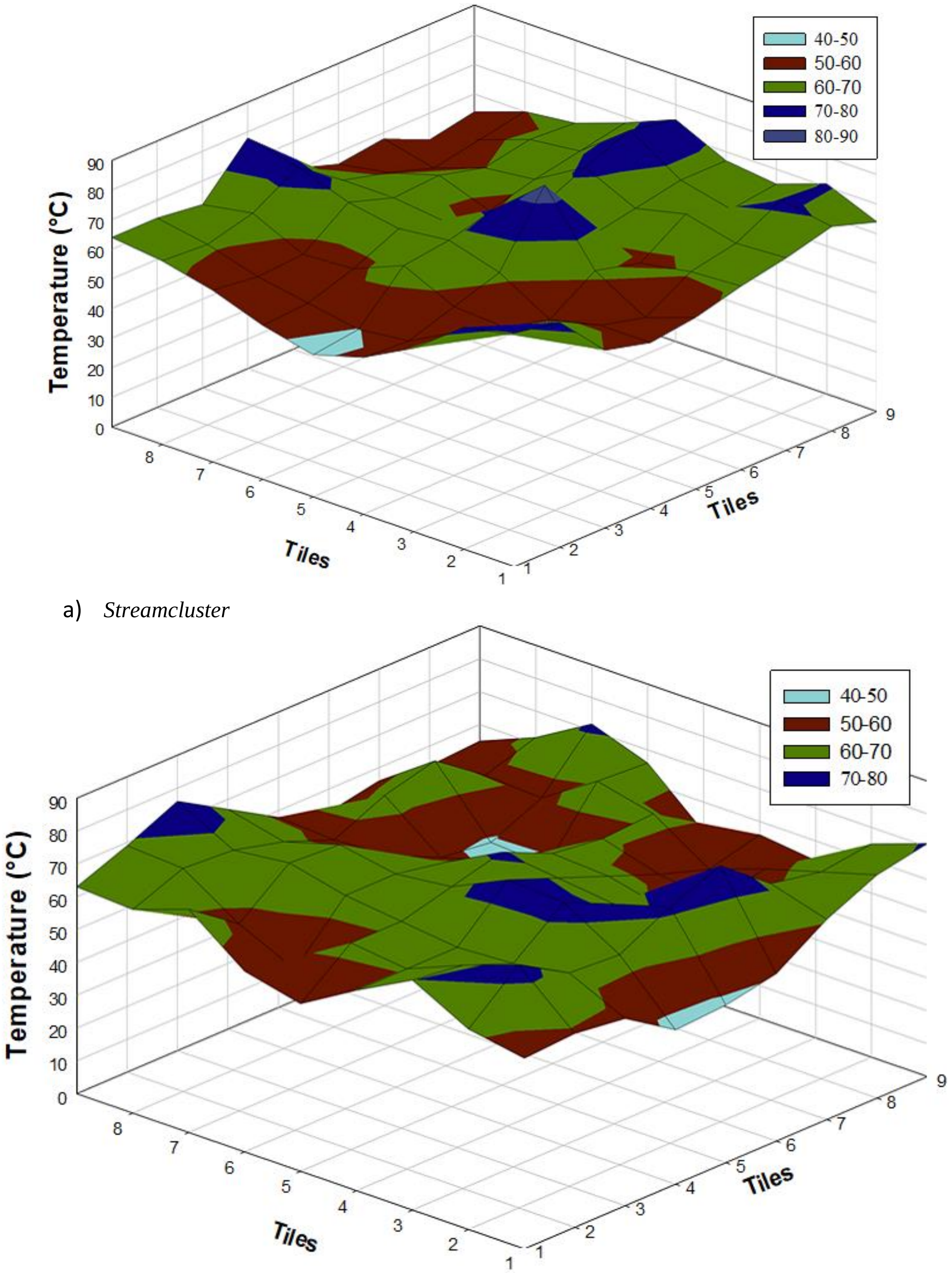


Figure 4.6: The comparison of delay for the *Proposed* and *Baseline* architecture.



a) *Streamcluster*

b) *bodytrack*

Figure 4.7: Thermal maps of the memory layer in a 64-core CMPs under executing a) *streamcluster* and b) *bodytrack* benchmarks.

Figure 4.7, shows the percentage time that the memory layer of the 3D CMPs spent at different temperature points under executing *streamcluster* and *bodytrack* benchmarks in each case. As shown in this figure, the proposed method ensures that the memory layer of the 3D CMPs is below the maximum temperature of 80°C.

4.5. Chapter Summary

In this chapter, a run-time task migration technique was proposed to balance the temperature in a 3D CMPs, including the core layer and the memory layer. In the target architecture based on using the information derived from performance counters of cores and DRAM memory banks, the core layer and the memory layer are divided to five regions with contrast temperature and these regions are stacked on each other to balance thermal distribution. Experimental results on the PARSEC benchmarks show that the proposed architecture yields up to 60% EDP, on average, reduction in overall chip energy consumption while the best improvement is recorded with only 72% EDP from the *Baseline* energy consumption. Moreover, the proposed architecture improves the IPC for *canneal* and *fluidanimate* by 18% and 14%, respectively. Finally, the proposed method ensures that the memory layer of the 3D CMPs is below the maximum temperature of *80°C*.

Chapter 5: The Second Proposed Method: Run-time Thermal Management Based on a New Task Migration Technique in 3D Chip Multiprocessors

5.1. Introduction

As mentioned before, RTM becomes necessary to control hotspots and thereby improving the performance of the 3D CMPs. However, applying the migration technique should consider hotspots both in the core layer and the stacked memory layer simultaneously. This can prevent to cause the emergence of new hotspots. Therefore, a new run-time task migration technique is being proposed in this chapter to control temperature variances both in the core layer and the memory layer simultaneously in order to make the optimum task migration decisions more efficiently.

In this chapter, the proposed technique aims to achieve balanced hotspots and temperature variations on the 3D CMPs. Therefore, it is crucial that the system must select the optimal coldest core to be migrated with the hottest core in the core layer rather than selecting the coldest core. The optimal coldest core refers to a cold core in the core layer that is not located under a hotspot DRAM bank.

Finally, the rest of this chapter is organized as follows. The target system architecture is described in section 5.2. In section 5.3, the proposed algorithm is evaluated and its significance is discussed. Experimental evaluation is presented in section 5.4, and lastly, section 5.5 summarizes the chapter.

5.2. The Target System Architecture

In this chapter, the target 3D CMPs architecture of the second method is similar to the first method which is shown in Figure 4.1. The first part is the down layer identified as the core layer. This layer includes 64 cores. Each core has a thermal sensor in order to measure its temperature as reflected in Figure 5.1. The second part of the target 3D CMPs architecture is the upper layer which is named the memory layer. This layer includes 64 DRAM banks. Each DRAM bank has a performance counter to calculate its accesses percentage level. Figure 5.2, shows each performance counters' connection with DRAM banks.

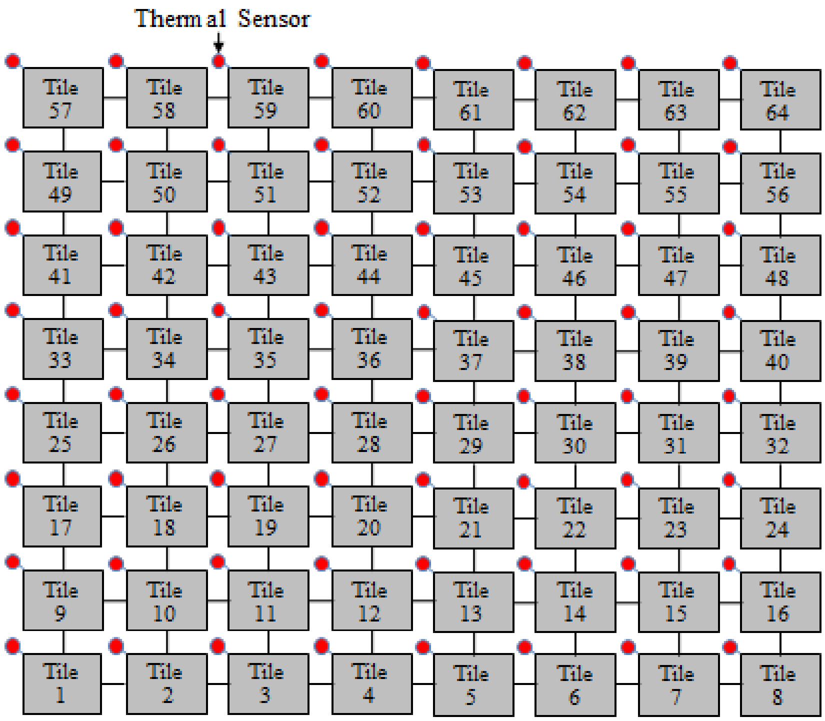

Therm al Sensor
Tile 57 Tile 58 Tile 59 Tile 60 Tile 61 Tile 62 Tile 63 Tile 64
Tile 49 Tile 50 Tile 51 Tile 52 Tile 53 Tile 54 Tile 55 Tile 56
Tile 41 Tile 42 Tile 43 Tile 44 Tile 45 Tile 46 Tile 47 Tile 48
Tile 33 Tile 34 Tile 35 Tile 36 Tile 37 Tile 38 Tile 39 Tile 40
Tile 25 Tile 26 Tile 27 Tile 28 Tile 29 Tile 30 Tile 31 Tile 32
Tile 17 Tile 18 Tile 19 Tile 20 Tile 21 Tile 22 Tile 23 Tile 24
Tile 9 Tile 10 Tile 11 Tile 12 Tile 13 Tile 14 Tile 15 Tile 16
Tile 1 Tile 2 Tile 3 Tile 4 Tile 5 Tile 6 Tile 7 Tile 8

Figure 5.1: Thermal sensors' placement on each core in the core layer.

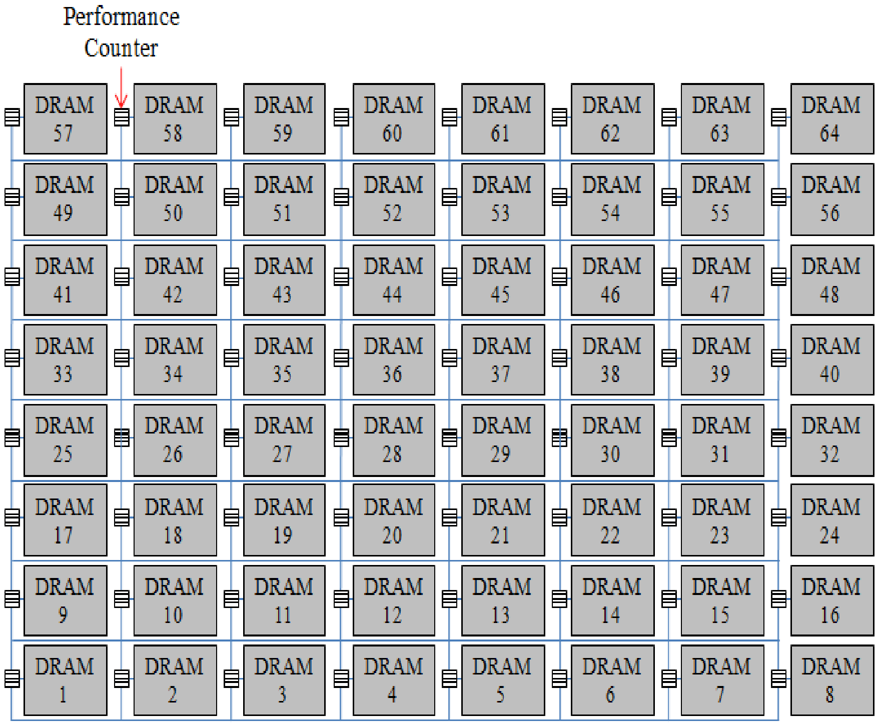
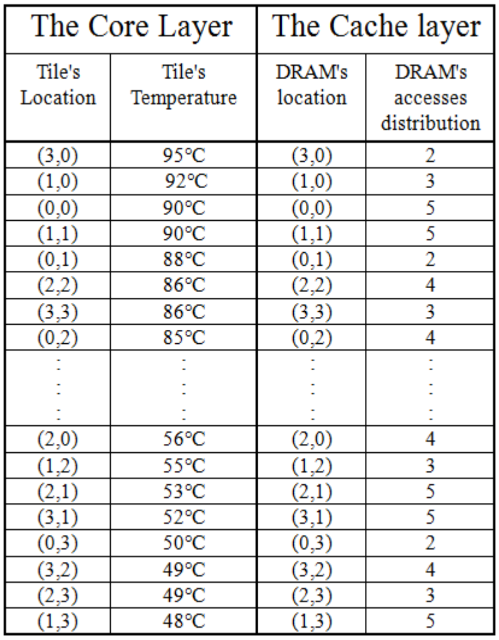

Performance Counter
DRAM 57 DRAM 58 DRAM 59 DRAM 60 DRAM 61 DRAM 62 DRAM 63 DRAM 64
DRAM 49 DRAM 50 DRAM 51 DRAM 52 DRAM 53 DRAM 54 DRAM 55 DRAM 56
DRAM 41 DRAM 42 DRAM 43 DRAM 44 DRAM 45 DRAM 46 DRAM 47 DRAM 48
DRAM 33 DRAM 34 DRAM 35 DRAM 36 DRAM 37 DRAM 38 DRAM 39 DRAM 40
DRAM 25 DRAM 26 DRAM 27 DRAM 28 DRAM 29 DRAM 30 DRAM 31 DRAM 32
DRAM 17 DRAM 18 DRAM 19 DRAM 20 DRAM 21 DRAM 22 DRAM 23 DRAM 24
DRAM 9 DRAM 10 DRAM 11 DRAM 12 DRAM 13 DRAM 14 DRAM 15 DRAM 16
DRAM 1 DRAM 2 DRAM 3 DRAM 4 DRAM 5 DRAM 6 DRAM 7 DRAM 8

Figure 5.2: Performance counters' connection with DRAM banks in the memory layer.

In this chapter, the proposed technique aims to obtain the most efficient task migration decision in order to minimize thermal impacts on the core layer and the memory layer of the 3D CMPs. Therefore, a centralized hardware named the Migration Control Unit (MCU) is introduced, similar to the first method presented in Chapter 4. In this method, MCU is the system decision maker, and it is responsible for the fulfillment of RTM in the 3D CMPs. MCU is assumed to have been placed near all of the cores in the core layer as shown in Figure 4.2. Moreover, in this technique, the MCU has a table named MCU table. The MCU table stores the system information such as lists of each core's temperature in the core layer and its location. It also includes the accesses distribution to each DRAM bank in the memory layer and its locations.



The Core Layer		The Cache layer	
Tile's Location	Tile's Temperature	DRAM's location	DRAM's accesses distribution
(3,0)	95°C	(3,0)	2
(1,0)	92°C	(1,0)	3
(0,0)	90°C	(0,0)	5
(1,1)	90°C	(1,1)	5
(0,1)	88°C	(0,1)	2
(2,2)	86°C	(2,2)	4
(3,3)	86°C	(3,3)	3
(0,2)	85°C	(0,2)	4
⋮	⋮	⋮	⋮
(2,0)	56°C	(2,0)	4
(1,2)	55°C	(1,2)	3
(2,1)	53°C	(2,1)	5
(3,1)	52°C	(3,1)	5
(0,3)	50°C	(0,3)	2
(3,2)	49°C	(3,2)	4
(2,3)	49°C	(2,3)	3
(1,3)	48°C	(1,3)	5

Figure 5.3: A sample of the sorted MCU table.

The MCU table is sorted in some steps which are explained in the next sub-section. An example of the sorted MCU table is illustrated in Figure 5.3. It is noticeable that statistical information is gathered from the entire processor and is sent to the MCU at the end of each time interval, fixed at 100ms in this work. The hardware overhead used in the target 3D CMPs architecture is shown in Table III.

TABLE III. HARDWARE OVERHEAD OF THE TARGET 3D CMPS ARCHITECTURE

The Used Hardware	**The Overhead**
Performance counters	64 × 32 bits
Thermal Sensors	64 sensors
MCU	64 × 59 bits

5.3. The Proposed Technique

In this section, the important issue for starting the proposed migration technique is how to make the most efficient decision for finding the best core to be migrated with a hotspot core in the core layer. Therefore, the most important point is which core amongst cold cores is the best to be selected.

In this context, in order to find the optimal coldest core, the MCU must consider the temperature of each core in the core layer simultaneously with the percentage accesses level of its related DRAM bank in the memory layer. After that, MCU selects the optimal coldest core. The selected optimal coldest core must not have a most accessed related DRAM bank. The related DRAM bank refers to the DRAM bank that located in the top of the intended core in the core layer. If the MCU selects a cold core where its related DRAM bank is a most accessed DRAM bank, then there is a possibility that the selected core becomes a hotspot faster.

In other words, selecting the optimal coldest core in the core layer will prevent the selection of a cold core that may be located under a most accessed DRAM bank and thereby it will prevent the possible appearance of any new hotspots.

Based on this trend, MCU should determine the hottest core to be migrated with the optimal coldest core in order to balance the temperature of the target 3D CMPs. To balance the temperature of the cores in the target 3D CMPs, MCU performs the following steps which are also shown in Algorithm III and Figure 5.4. Algorithm III, presents the proposed task migration technique and Figure 5.4, shows, in detail, the flowchart of selecting the optimal coldest core.

5.3.1. Measuring the Cores' Temperature and the DRAM Banks' Accesses Percentage Level

The measurement of the temperature of the cores and the calculations of the DRAM banks accesses' percentage level are prepared as follows:

- First, in the core layer: all cores read their temperature value from the thermal sensors and then send the information inside control packets to the MCU at the end of each time interval.
- Second, in the memory layer: all DRAM banks read the percentage level of their accesses based on performance counters and then send the information inside control packets to the MCU at the end of each time interval. It is noticeable that any DRAM memory bank with a higher percentage level of the accesses has a higher communication level with the cores. Thus, the higher communication rate results in

the DRAM memory banks getting a higher temperature. Therefore, the most accessed DRAM memory banks are assumed hotspots DRAM memory banks.

5.3.2. Finding Hotspots and Cold Spots in both Layers

To this end, MCU has gathered the statistical information from the core layer and the memory layer. This information includes each core temperature and its locations in the core layer. In addition, the accesses percentage level of each DRAM bank and its locations in the memory layer are also obtained. In this step, the MCU analyzes the received information and performs the following procedure:

1. Sorting the temperature of cores from the hottest to the coldest core in a descending order.
2. Filling each entry of the MCU table based on the accesses percentage of the related DRAM bank that is stacked above each core as shown in Figure 5.3.
3. Dividing the MCU table into four main groups as follows:

$$\left\{\begin{array}{cccc} 1 \le i \le 16 & ; \; j = 1 & : & HOTTEST \\ 17 \le i \le 32 & ; \; j = 2 & : & MEDHOT \\ 33 \le i \le 48 & ; \; j = 3 & : & MEDCOLD \\ 49 \le i \le 64 & ; \; j = 4 & : & COLDEST \end{array}\right\}$$

Where *i* is the core number on the sorted MCU table and *j* is the group number.

4. Finding the hottest core based on the sorted MCU table.

5.3.3. Finding the Optimal Coldest Core

In this context, upon obtaining the knowledge of all the hottest cores and the coldest cores in the core layer, the proposed algorithm finds the optimal coldest core. The

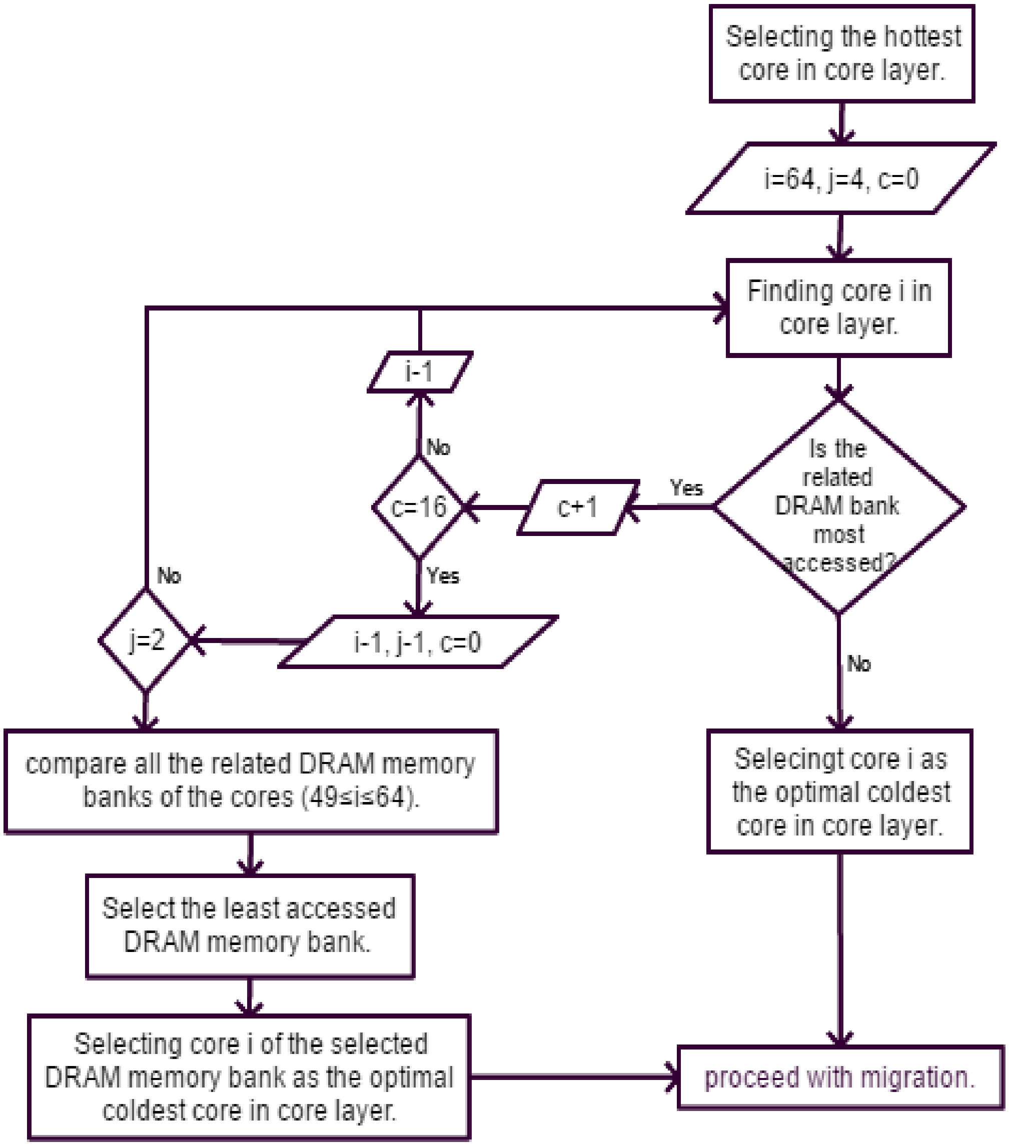
MCU analyzes the information in the sorted MCU table and then follows the procedure that is shown in Figure 5.4. As shown in Figure 5.4, the flowchart of selecting the optimal coldest core is presented where *i* is the core number on the sorted MCU table, *j* is the group number, and *c* is a counter.

The flowchart of selecting the optimal coldest core which is shown in Figure 5.4 is working as follows. The MCU starts with the fourth group (*j=4*), which contains the COLDEST cores ($49 \leq i \leq 64$). MCU then considers the coldest core ($i=64$) with the accesses percentage level of its related DRAM bank. If the related DRAM bank is not the most accessed bank, the MCU will select core *i* as the optimal coldest core. Otherwise, MCU checks the second coldest core ($i=63$) with the accesses percentage level of its related DRAM bank. If the related DRAM bank is not most accessed DRAM memory bank, MCU selects core *i* as an optimal coldest core. Otherwise, MCU will repeat the same procedure with the remaining coldest cores ($49 \leq i \leq 62$) in the fourth group ($j=4$) in order to find the optimal coldest core. If the MCU has not found the optimal coldest core in the fourth group ($j=4$), then it will apply the same procedure within the third group ($j=3$), which contains the MEDCOLD cores ($33 \leq i \leq 48$).

In this context, where MCU does not find the optimal coldest core in the fourth or the third group, MCU should proceed with the following procedure. It will compare all the related DRAM banks of the 16th coldest cores where ($j=4$) and ($49 \leq i \leq 64$) and then it will select the least accessed DRAM bank amongst them. After that, MCU will choose core *i* of the selected DRAM bank as an optimal coldest core which will be migrated with the hottest core.

Algorithm III: The Applied Algorithm of the Proposed Task Migration Technique.

```
Loop:
The temperature and the location of each core is determined.
The accesses' percentage level and the location of each DRAM bank is determined.
The TCU table is filled by sorting cores from the hottest to the coldest.
The accesses' percentage level of each related DRAM bank in the TCU table is tabulated.
The TCU table is divided into four main parts ($1 \leq j \leq 4$).
The hottest core in the core layer is selected.
    Loop:
    The optimal coldest tile is selected based on Figure 5.4.
    End loop;
Proceed with migration.
End loop;
```



Selecting the hottest core in core layer.
i=64, j=4, c=0
Finding core i in core layer.
Is the related DRAM bank most accessed?
Yes
c+1
c=16
No
i-1
Yes
i-1, j-1, c=0
j=2
No
No
compare all the related DRAM memory banks of the cores (49≤i≤64).
Select the least accessed DRAM memory bank.
Selecting core i of the selected DRAM memory bank as the optimal coldest core in core layer.
Selecingt core i as the optimal coldest core in core layer.
proceed with migration.

Figure 5.4: The flowchart of selecting the optimal coldest core.

5.3.4. Proceeding the Migration

Once the MCU has found the optimal coldest core, it will exchange the thread on the selected optimal coldest core with the thread on the hottest core. The proposed task migration mechanism assumes that the whole code and data of the tasks will be exchanged from the hottest core to the selected optimal coldest core. After that, the system should stop the running tasks and proceed to the task migration.

In conclusion, due to the high temperature dependency that exists among vertical adjacent cores and upper adjacent DRAM banks, performing a migration between the hottest and the selected optimal coldest cores can be very efficient in run-time the thermal management of the 3D CMPs. The proposed run-time migration technique has the ability to balance the hotspots both in the core layer and the memory layer simultaneously. Therefore, this technique is performed to balance hotspots among different cores on different layers in order to prevent any system failure, achieve higher overall performance, and fulfill a balanced chip temperature in the 3D CMPs.

5.4. The Experimental Evaluation

In this section, 64 cores in the core layer and 64 DRAM banks in the memory layer of a 3D CMPs architecture with multi-threaded workloads were used to perform the proposed run-time task migration technique. First, to evaluate the proposed technique, the experimental setup is described. Second, in order to quantify the benefits of the proposed architecture compared to the traditional architecture, different experiments are performed.

5.4.1. The Platform Setup

We evaluate our CMPs architecture of the proposed technique same as the previous chapter using a simulation platform built upon Gem5 [50] to run our experiments and measure system performance improvements. GEM5 is a full system simulator to set up the basic system platform. A 64-core architecture which formed 2D-mesh topology was modeled as shown in Figure 4.1. Traces from GEM5 were injected to 3D Noxim [51], a System-C simulator for 3D NoCs modeling the mesh-based on-chip interconnection network between cores and DRAM banks. Table IV illustrates the detailed parameters used in the CMPs architecture.

TABLE IV. BASELINE CMPS CONFIGURATION.

The Component	**The Description**
Number of Cores	64, 8×8 mesh
Core Configuration	Alpha21264, 1GHz, in-order, 14-stage pipeline, 45nm.
Private Memory per each Core	SRAM, 4 way, 64 line, size 64KB per core
On-chip Memory	Baseline: 64MB (64 DRAM banks, each of which has 512KB capacity) Proposed: 64MB (64 DRAM banks, each of which has 512KB capacity) + proposed migration policy

In addition, PARSEC benchmarks [52] for multi-thread workloads were used in this context. The multithreaded applications with large working sets are selected from PARSEC benchmark suite [52]. These selected benchmark suits consist of emerging workloads suitable for the next generation shared-memory programs for the CMPs. For these benchmarks, one billion instructions were executed for the *simlarge* input that was set starting from the Region of Interest (ROI). Moreover, HotSpot [53] version 5.0 was employed as a grid-based thermal modeling tool for 3D temperature estimation. For the experimental evaluation, the maximum temperature limit T_{max} was assumed to be 80℃.

5.4.2. The Experimental Results

In this section, the 3D CMPs hierarchy were evaluated in two different cases on multithreaded workloads. Firstly, the 3D CMPs with DRAM banks in the memory layer stacked on the top of the core layer without any migration policy (*Baseline*). Secondly, the 3D CMPs with a DRAM memory layer stacked on top of the core layer with the proposed migration policy (*Proposed*).

Figure 5.5, shows the results of the normalized throughput for PARSEC workloads, where throughput is the number of executed Instructions per Second (IPS). As shown in Figure 5.5, *Proposed* yields up to 3% throughput improvement when compared with the *Baseline*.

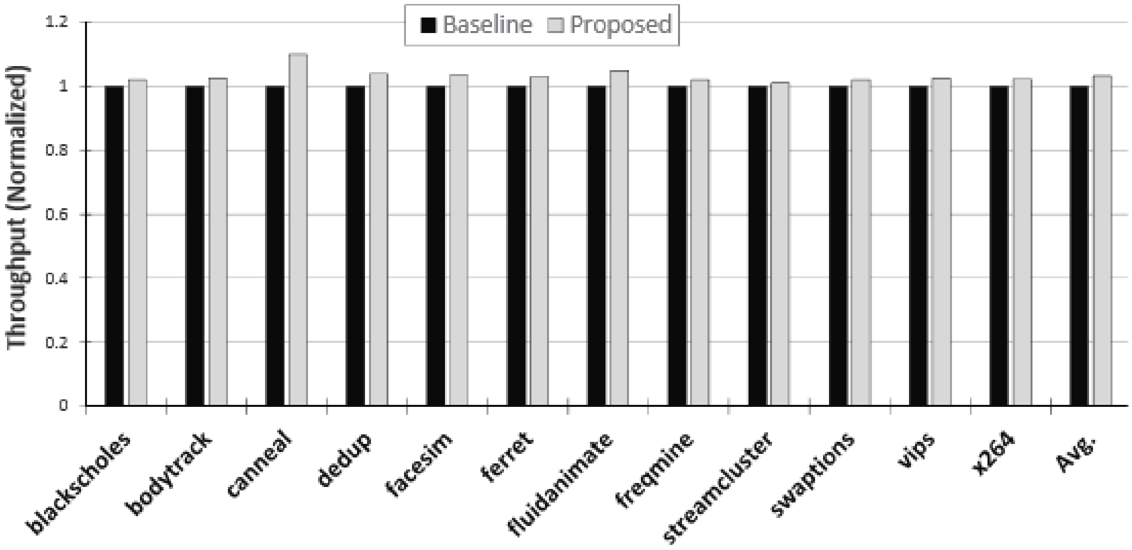


Figure 5.5: Comparison of throughput results of the PARSEC workloads normalized to the *Baseline*.

Figure 5.6, shows the time percentage that the upper layer of the 3D CMPs in each case spent, on average, at different temperature points while executing the *canneal* application in order to get a good representation of the memory intensive workloads in PARSEC. As shown in Figure 5.6, the *Proposed* method ensures that the memory layer of the 3D CMPs is below the maximum temperature of 80℃, while the *Baseline* architecture spends up to 18% of time above the maximum temperature.

As shown in Figure 5.7, when the *blackscholes* is executed as one of the more computation intensive suite in PARSEC benchmarks, the *Baseline* spent up to 28% the time above the maximum temperature.

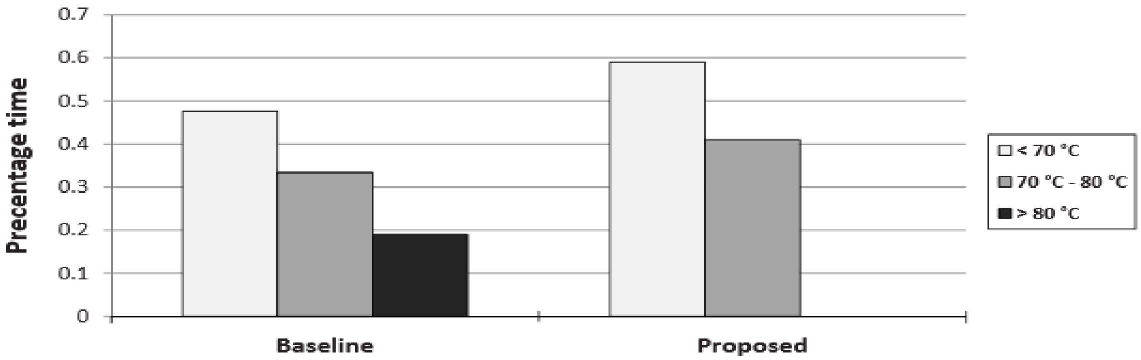


Figure 5.6: Comparison of the percentage time spent on average by the DRAM layer of the target 3D CMPs at different temperature points while executing *canneal*.

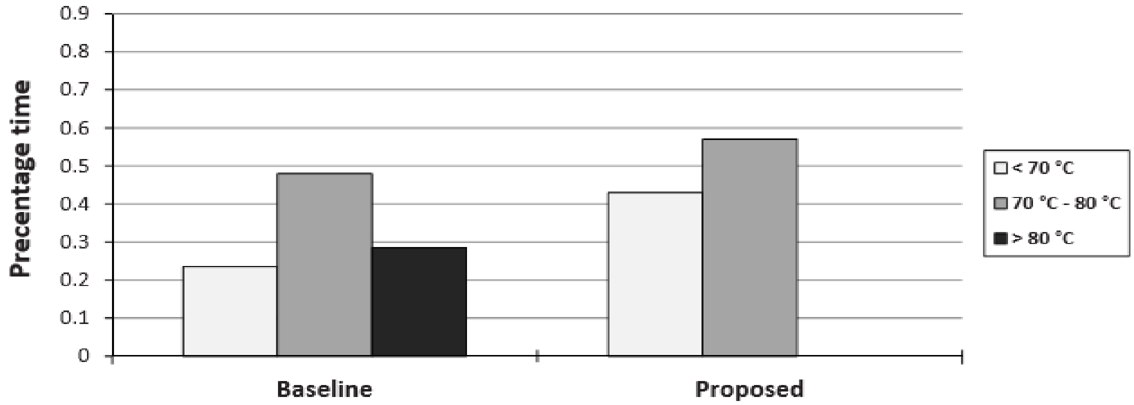


Figure 5.7: Comparison of the percentage time spent on average by the DRAM layer of the target 3D CMPs while executing *blackscholes*.

According to Figure 5.8, the temperature of the upper memory layer in the proposed architecture is lower than the temperature limit under execution of *canneal*, but in the *Baseline* architecture, the temperature for memory intensive is above the limit. As shown in this figure, there is a difference of 13℃ on average between the *Proposed* method and the *Baseline* temperature.

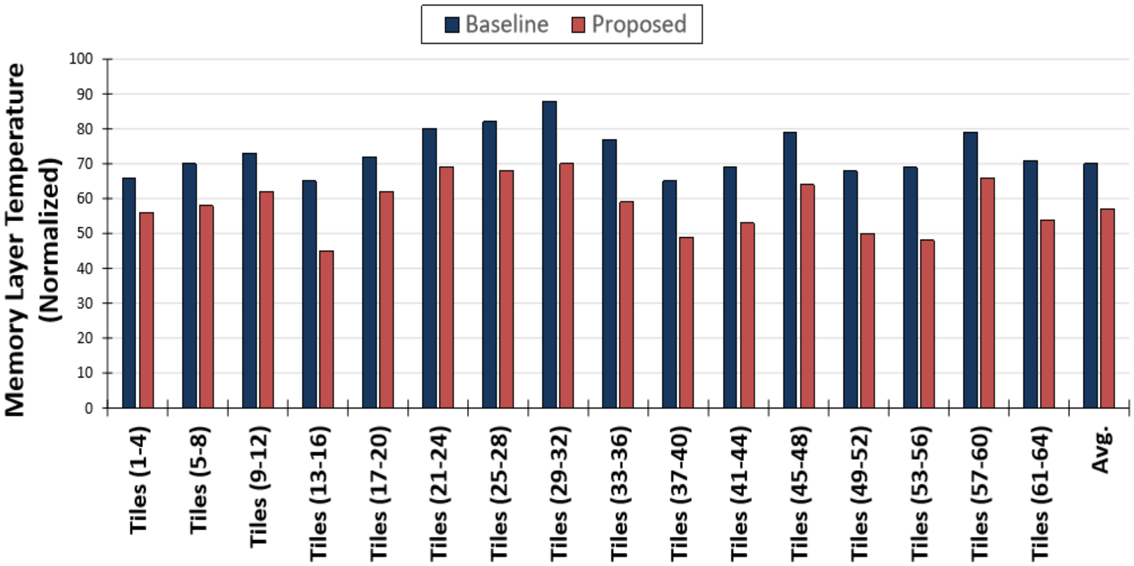


Figure 5.8: The temperature of the upper layer in *canneal*.

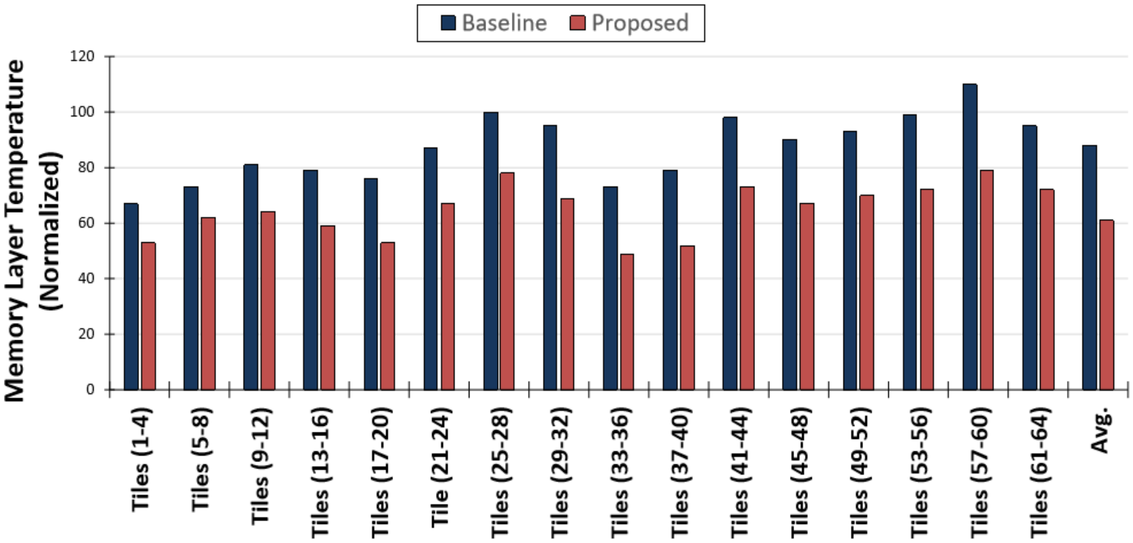


Figure 5.9: The temperature of the upper layer in *blackscholes.*

Moreover, Figure 5.9, demonstrates that the temperature of the memory layer in the proposed architecture is lower than the temperature limit while the execution of *blackscholes* that is a computation intensive workload, but in the *Baseline* architecture, the temperature is higher than the limit. As shown in this figure, on average there is a 24℃ difference between the *Proposed* method and the *Baseline* temperature.

Figure 5.10, shows the thermal distribution of the *Baseline* architecture while executing the memory intensive application *canneal*. As can be seen in this Figure, there are four hotspots in the *Baseline* architecture with a maximum temperature of *110℃* that violates the *80℃* constraints. In addition, as shown in Figure 5.11, the distribution of temperature is more uniform than the *Baseline* architecture and there are not any hotspots in the *Proposed* method.

It can be seen that the difference between the maximum temperature degree in Figure 5.10, and Figure 5.11, is *33℃*. Moreover, the *Proposed* technique did not violate the maximum temperature constraint. Thus, the obtained results demonstrated in Figure 5.10, and Figure 5.11, show that the proposed task migration technique is efficient in minimizing thermal variance in the target 3D CMPs architecture.

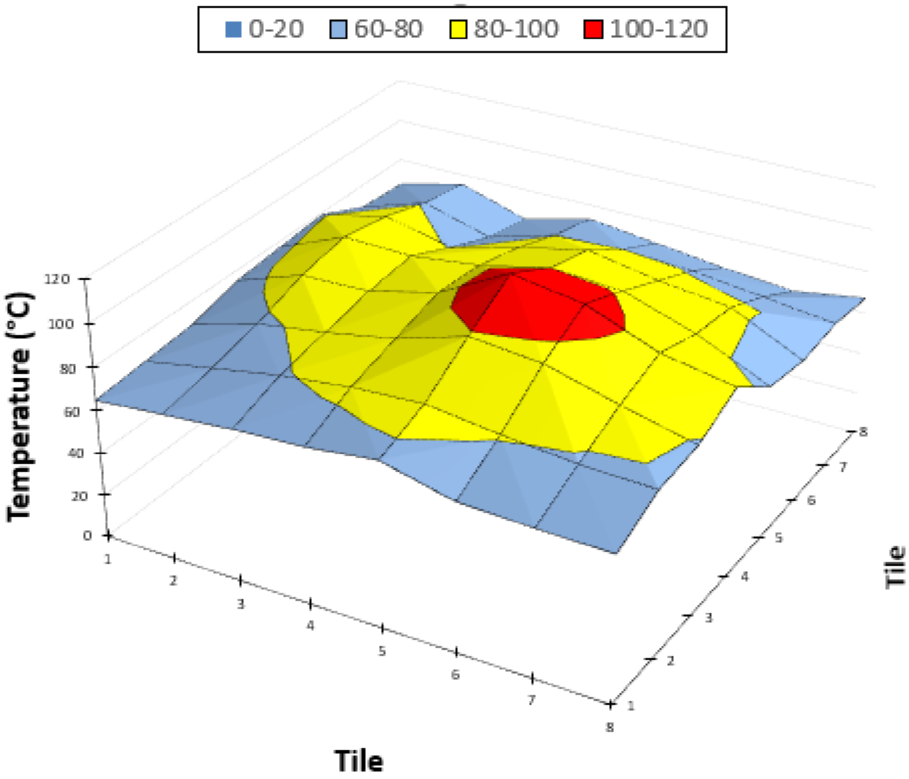


Figure 5.10: The thermal distribution of the *Baseline* architecture under execution of *canneal*.

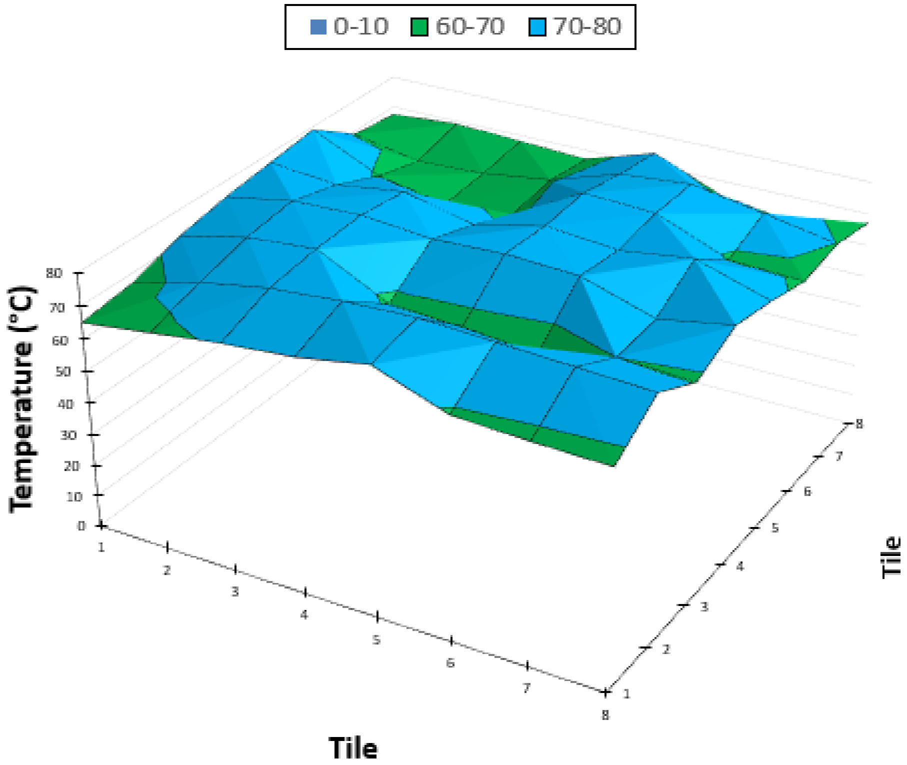


Figure 5.11: The thermal distribution of the *Proposed* method architecture under execution of *canneal*.

5.5. Chapter Summary

In this chapter, a new run-time task migration technique is presented for 3D CMPs in order to balance thermal hotspots. The proposed technique introduces a new decision maker unit named MCU which aims to find the optimal coldest core to be migrated with the hottest core in the core layer. The proposed algorithm continuously analyzes and detects the hottest and the optimal coldest cores in the core layer considering the most accessed DRAM banks in the memory layer simultaneously. Since the proposed technique considers hotspots and cold spots in different layers, the obtained results show a significant reduction of hotspots in the whole 3D CMPs. The obtained simulation results indicate up to a 33℃ (on average 24℃) reduction in the temperature value of the 3D CMPs. Moreover, the *Proposed* technique yields up to 3% throughput improvement when compared with the *Baseline*. Finally, the simulation results clarified that the *Proposed* technique efficiency was the maximum temperature of cores in the core and memory layers are both less than maximum temperature limit 80℃; however, there are four hotspots in the *Baseline* with a maximum temperature of *113℃* that violates the *80℃* constraints.

Chapter 6: Thesis Conclusions

6.1. Thesis Summary

Nowadays, with continued improvement on the Chip Multiprocessors (CMPs) architecture for the purpose of achieving higher performance and more reliable systems, CMPs architecture moves from multi-core to many-core architecture, and also moves from 2D CMPs to 3D CMPs. These features allow the 3D CMPs to execute heavy-loaded tasks, and thus, improve the system performance by increasing system power consumption. However, a new master challenge has emerged recently, which is the appearance of on-chip thermal hotspots. Since thermal hotspots cause performance degradation, and reducing reliability, applying Run-time Thermal Management (RTM) has become inevitable to control the thermal hotspots without any performance degradation.

In this thesis, two run-time task migration techniques are proposed to control hotspots in the target 3D CMPs. These techniques aimed to achieve a balanced 3D CMPs temperature by considering hotspots both in the core layer and the memory layer simultaneously. In the two proposed techniques, a centralized hardware named MCU is presented which is the system decision maker unit.

The first proposed technique aimed at balancing hotspots by providing two algorithms in the core layer and the memory layer that are working in parallel. This proposed migration technique gathers the temperature of cores and DRAM banks by using performance-counters instead of using thermal sensors. The proposed technique aimed to achieve balanced thermal distribution in the 3D CMPs. According to sections 4.3.1 and 4.3.2, the proposed algorithms migrates the cold cores to the central part and hot cores to the surrounding part in the core layer and also migrates the most accessed memory banks to the central part in the memory layer. Based on this methodology, the central region of the memory layer, which is the hottest region, is placed on top of the central part of the core layer, which is the coldest region in the core layer. Additionally, the surrounding part of the memory layer as a cold region is placed on top of the surrounding part of the core layer which is a hot region.

The second proposed technique aimed to control hotspots by migrating the hottest core in the core layer with the optimal coldest core rather than the coldest core. The optimal coldest core is selected by considering hotspots both in the core layer and the memory layer simultaneously. Therefore, performing the proposed migration technique between the hottest and the selected optimal coldest cores is very efficient in the run-time thermal management on the 3D CMPs due to the high temperature dependency that exists among vertical adjacent cores and upper adjacent DRAM banks.

6.2. Thesis Conclusions

In fact, applying task migration algorithm in the 3D CMPs to migrate a hotspot to a low temperature core in the core layer only to address emerging thermal hotspot issues without considering thermal hotspots on the stacked memory layer has the potential to cause the emergence of new hotspots. Therefore, the main objective of the two proposed methods is to consider hotspots impact both in core and memory layers simultaneously for the purpose of making the optimal decision of when and where to procced run-time task migration technique on the 3D CMP.

With regards to the first approach, experimental results on the PARSEC benchmarks show that the proposed architecture yields up to 60%, on average, reduction in overall chip energy consumption while the best improvement is recorded with only 72% from the *Baseline* energy consumption. Moreover, the proposed architecture improves the IPC for *canneal* and *fluidanimate* by 18% and 14%, respectively. Finally, the proposed method ensures that the memory layer of the 3D CMPs is below the maximum temperature of *80°C*.

The experimental results of the second approach yield up to 3% throughput improvement on the proposed technique when compared with the *Baseline*. Moreover, the proposed technique indicates up to 33℃ (on average 24℃) reduction in the cores' temperature of the target 3D CMPs. Finally, the simulation results clarified that the proposed technique efficiency was below the maximum temperature limit 80℃; however, there are four hotspots in the *Baseline* with a maximum temperature of *113℃* that violates the *80℃* constraints.

6.3. Future Work

It is suggested that researchers on the field of the RTM techniques on the 3D CMPs architecture consider the following factors on the future research works:

1. Applying machine learning techniques in predicting the temperature of cores and memory banks in the stacked layers of the proposed architecture instead of using expensive sensors.
2. Applying distributed MCUs instead of one to extend the multicore platforms to more than 1000 cores.
3. Proposing heterogenous memory and core layers based on the emerging low power technologies.

References

[1] Zhou, X., Xu, Y., Du, Y., Zhang, Y., & Yang, J. (2008, September). Thermal management for 3D processors via task scheduling. In *Parallel Processing, 2008. ICPP'08. 37th International Conference on* (pp. 115-122). IEEE.

[2] Liao, C. H., Wen, C. H. P., & Chakrabarty, K. (2015, March). An online thermal-constrained task scheduler for 3D multi-core processors. In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition* (pp. 351-356). EDA Consortium.

[3] Ge, Y., Qiu, Q., & Wu, Q. (2012). A multi-agent framework for thermal aware task migration in many-core systems. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *20*(10), 1758-1771.

[4] Ge, Y., Malani, P., & Qiu, Q. (2010, June). Distributed task migration for thermal management in many-core systems. In *Proceedings of the 47th Design Automation Conference* (pp. 579-584). ACM.

[5] Liu, Z., Huang, X., Tan, S. X. D., Wang, H., & Tang, H. (2013, October). Distributed task migration for thermal hot spot reduction in many-core microprocessors. In *ASIC (ASICON), 2013 IEEE 10th International Conference on* (pp. 1-4). IEEE.

[6] Skadron, K., Stan, M. R., Huang, W., Velusamy, S., Sankaranarayanan, K., & Tarjan, D. (2003, June). Temperature-aware microarchitecture. In *Computer Architecture, 2003. Proceedings. 30th Annual International Symposium on* (pp. 2-13). IEEE.

[7] Heo, S., Barr, K., & Asanovic, K. (2003, August). Reducing power density through activity migration. In *Low Power Electronics and Design, 2003. ISLPED'03. Proceedings of the 2003 International Symposium on* (pp. 217-222). IEEE.

[8] Kumar, A., Shang, L., Peh, L. S., & Jha, N. K. (2006, July). HybDTM: a coordinated hardware-software approach for dynamic thermal management.

In *Proceedings of the 43rd annual Design Automation Conference* (pp. 548-553). ACM.

[9] Sun, C., Shang, L., & Dick, R. P. (2007, September). Three-dimensional multiprocessor system-on-chip thermal optimization. In *Proceedings of the 5th IEEE/ACM international conference on Hardware/software codesign and system synthesis* (pp. 117-122). ACM.

[10] Donald, J., & Martonosi, M. (2006, June). Techniques for multicore thermal management: Classification and new exploration. In *ACM SIGARCH Computer Architecture News* (Vol. 34, No. 2, pp. 78-88). IEEE Computer Society.

[11] Zhan, J., Ouyang, J., Ge, F., Zhao, J., & Xie, Y. (2015, June). DimNoC: A dim silicon approach towards power-efficient on-chip network. In *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE* (pp. 1-6). IEEE.

[12] Bokhari, H., Javaid, H., Shafique, M., Henkel, J., & Parameswaran, S. (2014, June). darknoc: Designing energy-efficient network-on-chip with multi-vt cells for dark silicon. In *Proceedings of the 51st Annual Design Automation Conference* (pp. 1-6). ACM.

[13] Zhan, J., Xie, Y., & Sun, G. (2014, June). NoC-sprinting: Interconnect for fine-grained sprinting in the dark silicon era. In *Proceedings of the 51st Annual Design Automation Conference* (pp. 1-6). ACM.

[14] Asad, A., Ozturk, O., Fathy, M., & Jahed-Motlagh, M. R. (2017). Optimization-based power and thermal management for dark silicon aware 3D chip multiprocessors using heterogeneous cache hierarchy. *Microprocessors and Microsystems*, *51*, 76-98.

[15] Asad, A., Fathy, M., Jahed-Motlagh, M. R. (2017). Power Modeling and Run-time Performance Optimization of Power Limited Many-Core Systems Based on a Dynamic Adaptive Approach. *Journal of Low Power Electronics*, *13*(2), 166-195.

[16] Chen, X., Xu, Z., Kim, H., Gratz, P., Hu, J., Kishinevsky, M., & Ogras, U. (2013). In-network monitoring and control policy for DVFS of CMP networks-on-chip and last level caches. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, *18*(4), 47.

[17] Shang, L., Peh, L. S., & Jha, N. K. (2002). Power-efficient interconnection networks: Dynamic voltage scaling with links. *IEEE Computer Architecture Letters*, *1*(1), 6-6.

[18] Son, S. W., Malkowski, K., Chen, G., Kandemir, M., & Raghavan, P. (2006, April). Integrated link/cpu voltage scaling for reducing energy consumption of parallel sparse matrix applications. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International* (pp. 8-pp). IEEE.

[19] Luo, J., & Jha, N. K. (2007). Power-efficient scheduling for heterogeneous distributed real-time embedded systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 26*(6), 1161-1170.

[20] Guang, L., Nigussie, E., Koskinen, L., & Tenhunen, H. (2009, March). Autonomous DVFS on supply islands for energy-constrained NoC communication. In *International Conference on Architecture of Computing Systems* (pp. 183-194). Springer, Berlin, Heidelberg.

[21] Dorostkar, A., Asad, A., Fathy, M., & Mohammadi, F. (2017, August). Optimal Placement of Heterogeneous Uncore Component in 3D Chip-Multiprocessors. In *Digital System Design (DSD), 2017 Euromicro Conference on* (pp. 547-551). IEEE.

[22] Chen, X., Xu, Z., Kim, H., Gratz, P. V., Hu, J., Kishinevsky, M., ... & Ayoub, R. (2013, May). Dynamic voltage and frequency scaling for shared resources in multicore processor designs. In *Proceedings of the 50th Annual Design Automation Conference* (p. 114). ACM.

[23] Arezoomand, F., Asad, A., Fazeli, M., Fathy, M., & Mohammadi, F. (2017, July). Energy aware and reliable STT-RAM based cache design for 3D embedded chip-multiprocessors. In *Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2017 12th International Symposium on* (pp. 1-8). IEEE.

[24] Flautner, K., Kim, N. S., Martin, S., Blaauw, D., & Mudge, T. (2002). Drowsy caches: simple techniques for reducing leakage power. In *Computer Architecture, 2002. Proceedings. 29th Annual International Symposium on* (pp. 148-157). IEEE.

[25] Liu, Z., Tan, S. X. D., Huang, X., & Wang, H. (2015). Task migrations for distributed thermal management considering transient effects. *IEEE Transactions on Very Large-Scale Integration (VLSI) Systems*, *23*(2), 397-401.

[26] Hassanpour, N., Khadem, P., & Hessabi, S. (2013). A task migration technique for temperature control in 3D NoCs. In *27th IEEE International Conference on Advanced Information Networking and Applications (AINA). Manuscript submitted for publication*.

[27] Aljeddani, S., & Mohammadi, F, (2017). Runtime thermal management based on a novel task migration technique in 3D Chip Multiprocessors, *IJIRR Vol. 04, Issue, 08, pp.4415-4423*.

[28] Aljeddani, S., & Mohammadi, F, "A Novel Migration Technique to Balance Thermal Distribution for Future Heterogeneous 3D Chip Multiprocessors", it is accepted to be published in (*the 8th ICIST 2018*) will be held in Cordoba, Granada, and Seville, in Spain, during June 30-July 6, 2018, *IEEE*.

[29] Safayenikoo, P., Asad, A., Fathy, M., & Mohammadi, F. (2017, March). An energy efficient non-uniform Last Level Cache Architecture in 3D chip-multiprocessors. In *Quality Electronic Design (ISQED), 2017 18th International Symposium on* (pp. 373-378). IEEE.

[30] Esmaeilzadeh, H., Blem, E., Amant, R. S., Sankaralingam, K., & Burger, D. (2012). Dark silicon and the end of multicore scaling. *IEEE Micro*, *32*(3), 122-134.

[31] Mohammad, B. (2015). Embedded memory interface logic and interconnect testing. *IEEE Transactions on Very Large-Scale Integration (VLSI) Systems*, *23*(9), 1946-1950.

[32] Mollick, E. (2006). Establishing Moore's law. *IEEE Annals of the History of Computing*, *28*(3), 62-75.

[33] Zhang, Y., Li, L., Lu, Z., Jantsch, A., Gao, M., Pan, H., & Han, F. (2014). A survey of memory architecture for 3D chip multi-processors. *Microprocessors and Microsystems*, *38*(5), 415-430.

[34] Hennessy, J. L., & Patterson, D. A. (2011). *Computer architecture: a quantitative approach*. Elsevier.

[35] Asad, A., Fathy, M., Jahed-Motlagh, M. R. (2017). Power Modeling and Run-time Performance Optimization of Power Limited Many-Core Systems Based on a Dynamic Adaptive Approach. *Journal of Low Power Electronics*, *13*(2), 166-195.

[36] Neisser, M., & Wurm, S. (2015). ITRS lithography roadmap: 2015 challenges. *Advanced Optical Technologies*, *4*(4), 235-240.

[37] Agarwal, A., Iskander, C., & Shankar, R. (2009). Survey of network on chip (noc) architectures & contributions. *Journal of engineering, Computing and Architecture*, *3*(1), 21-27.

[38] Zonouz, A. E., Seyrafi, M., Asad, A., Soryani, M., Fathy, M., & Berangi, R. (2009, August). A fault tolerant NoC architecture for reliability improvement and latency reduction. In *Digital System Design, Architectures, Methods and Tools, 2009. DSD'09. 12th Euromicro Conference on* (pp. 473-480). IEEE.

[39] Asad, A., Seyrafi, M., Zonouz, A. E., Soryani, M., & Fathy, M. (2009, November). A Predominant Routing for on-chip networks. In *Design and Test Workshop (IDT), 2009 4th International* (pp. 1-6). IEEE.

[40] Marculescu, R., Ogras, U. Y., Peh, L. S., Jerger, N. E., & Hoskote, Y. (2009). Outstanding research problems in NoC design: system, microarchitecture, and circuit perspectives. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *28*(1), 3-21.

[41] P. Safaie, A. Asad, F. Fathy, and F. Mohammadi, "*An Energy Efficient Uncore Architecture in 3D Chip Multiprocessors*" 30th Canadian Conference on Electrical and Computer Engineering, Windsor, ON, April30- May 3, 2017.

[42] Yun, W., Kang, K., & Kyung, C. M. (2011, May). Thermal-aware energy minimization of 3D-stacked L3 cache with error rate limitation. In *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on* (pp. 1672-1675). IEEE.

[43] Li, F., Nicopoulos, C., Richardson, T., Xie, Y., Narayanan, V., & Kandemir, M. (2006, June). Design and management of 3D chip multiprocessors using network-in-memory. In *ACM SIGARCH Computer Architecture News* (Vol. 34, No. 2, pp. 130-141). IEEE Computer Society.

[44] Arezoomand, F., Asad, A., Fazeli, M., Fathy, M., & Mohammadi, F. (2017, September). Reliability and Power optimization in 3D-stacked cache using a run-

time reconfiguration procedure. In *Embedded Multicore/Many-core Systems-on-Chip (MCSoC), 2017 IEEE 11th International Symposium on* (pp. 75-82). IEEE.

[45] Zhao, J., Xu, C., & Xie, Y. (2011, November). Bandwidth-aware reconfigurable cache design with hybrid memory technologies. In *Proceedings of the International Conference on Computer-Aided Design* (pp. 48-55). IEEE Press.

[46] Prasad, E. L., Sivasankaran, V., & Nagarajan, V. (2012, February). Multiple task migration in mesh network on chips over virtual point-to-point connections. In *Computing, Communication and Applications (ICCCA), 2012 International Conference on* (pp. 1-4). IEEE.

[47] Safayenikoo, P., Asad, A., Fathy, M., & Mohammadi, F. (2017, April). Exploiting non-uniformity of write accesses for designing a high-endurance hybrid Last Level Cache in 3D CMPs. In *Electrical and Computer Engineering (CCECE), 2017 IEEE 30th Canadian Conference on* (pp. 1-5). IEEE.

[48] Shafique, M., Garg, S., Mitra, T., Parameswaran, S., & Henkel, J. (2014, October). Dark silicon as a challenge for hardware/software co-design. In *Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2014 International Conference on* (pp. 1-10). IEEE.

[49] Esmaeilzadeh, H., Blem, E., Amant, R. S., Sankaralingam, K., & Burger, D. (2011, June). Dark silicon and the end of multicore scaling. In *Computer Architecture (ISCA), 2011 38th Annual International Symposium on* (pp. 365-376). IEEE.

[50] Binkert, N., Beckmann, B., Black, G., Reinhardt, S. K., Saidi, A., Basu, A., ... & Sen, R. (2011). The gem5 simulator. *ACM SIGARCH Computer Architecture News*, *39*(2), 1-7.

[51] Palesi, M., Kumar, S., & Patti, D. (2010). Noxim: Network-on-chip simulator.

[52] Gebhart, M., Hestness, J., Fatehi, E., Gratz, P., & Keckler, S. W. (2009). Running PARSEC 2.1 on M5. *The University of Texas at Austin, Department of Computer Science, Tech. Rep*.

[53] Huang, W., Ghosh, S., Velusamy, S., Sankaranarayanan, K., Skadron, K., & Stan, M. R. (2006). HotSpot: A compact thermal modeling methodology for early-stage VLSI design. *IEEE Transactions on Very Large-Scale Integration (VLSI) Systems*, *14*(5), 501-513.

[54] Magnusson, P. S., Christensson, M., Eskilson, J., Forsgren, D., Hallberg, G., Hogberg, J., & Werner, B. (2002). Simics: A full system simulation platform. *Computer*, *35*(2), 50-58.

[55] P. Safaie, A. Asad, F. Fathy, and F. Mohammadi, "*A Reconfigurable Hybrid NonUniform Last Level Cache in 3D Many Core Processors*", 30th Canadian Conference on Electrical and Computer Engineering, Windsor, ON, April30- May 3, 2017.

[56] Safayenikoo, P., Asad, A., Fathy, M., & Mohammadi, F. (2017, April). A new traffic compression method for end-to-end memory accesses in 3D chip-multiprocessors. In *Electrical and Computer Engineering (CCECE), 2017 IEEE 30th Canadian Conference on* (pp. 1-4). IEEE.

Glossary

CMPs	Chip Multiprocessors
RTM	Runtime Thermal Management
2D ICs	Two Dimensional Integrated Circuits
3D ICs	Three Dimensional Integrated Circuits
TSVs	Through Silicon Vias
MCU	Migration Control Unit
DRAM	Dynamic Random-Access Memory
NoC	Network on Chip
SoC	System on Chip
PE	Processing Elements
IP	Intellectual Property
CPU	Central Processing Unit
LLC	Last Level Cashe
NVM	Non-volatile memory
DVFS	Dynamic Voltage Frequency Scaling
RAM	Random-Access Memory
SRAM	Static RAM
STT-RAM	Spin-Transfer Torque RAM
VFI	Voltage Frequency Island
DVS	Dynamic Voltage Scaling
ED	Energy-Delay
AMAT	Average Memory Access Time
PI	Proportional and Integral
MCDs	Multiple Clock Domains
EPI	Energy per Instruction
CPI	Cycles Per Instruction
EDP	Energy–Delay Product
OS	Operating Systems
TPAVA	Thermal-Pattern-Aware Voltage Assignment
VGVS	Vertical-Grouping Voltage Scaling
PDP	Power-Delay Product