Exploring the Scalability and Performance of Networks-on-Chip with Deflection Routing in 3D Many-core Architecture

AWET YEMANE WELDEZION

Doctoral Thesis in Electronic and Computer Systems
Stockholm, Sweden 2016

KTH School of Information and
Communication Technology
SE-164 40 Kista
Sweden

TRITA-ICT 2015-29
ISBN 978-91-7595-803-3

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen 2016-01-20, klockan 13:00 i Sal C, Electrum, Isafjordsgatan 26, Kista.

Tryck: Universitetsservice US AB

Abstract

Three-Dimensional (3D) integration of circuits based on die and wafer stacking using through-silicon-via is a critical technology in enabling "more-than-Moore", i.e. functional integration of devices beyond pure scaling ("more Moore"). In particular, the scaling from multi-core to many-core architecture is an excellent candidate for such integration. Nevertheless, as much as there are opportunities to explore, designing systems using 3D integration technology has many challenges to tackle. It follows a complex design process involving integration of heterogeneous technologies. It is also expensive to prototype because the 3D industrial ecosystem is not yet complete and ready for low-cost mass production.

With trends leading towards 3D many-core architecture, it is also imperative to extend the under-lying Networks-on-Chip (NoC) to efficiently facilitate the communication of such massively integrated cores on a 3D chip. In this thesis scalability and performance issues of NoCs are explored in terms of architecture, organization and functionality of many-core systems. The key contributions of the thesis are made by (1) addressing the challenges in modeling and development of deflection routing NoCs for the use in regular and irregular networks, (2) evaluating new configurations of 3D processor-memory stacking, (3) the use of multi-rate vertical interconnect to optimize network performance, and (4) developing predictive models for performance analysis in many-core architecture.

First, we evaluate on-chip network performance in massively integrated many-core architecture. With each addition of cores, the on-chip network size grows and predicting the performance becomes challenging. We propose link and channel models to analyze the network traffic and hence the performance. We consider the absence of a simulation platform for such scalable many-core architecture as a key challenge. We develop a NoC simulation framework to evaluate the performance of a deflection routing network as the architecture scales up to 1000 cores. In designing processor-memory architecture, we propose models and do comparative analysis of proposed 3D processor-memory configurations in scalable many-core architectures.

Second, we investigate how the deflection routing NoCs can be designed to maximize the benefit of the fast TSVs as vertical interconnects by clocking them at a rate that is an integer multiple of the clock frequency of the horizontal links. We propose multi-rate models for inter-layer communication. We quantify the performance benefit through cycle-accurate simulations for various configurations of 3D architectures.

Finally, the complexity of massively integrated many-core architecture by itself brings a multitude of design challenges such as high-cost of prototyping, increasing complexity of the technology, irregularity of the communication network, and lack of reliable simulation models. In order to reduce the design to market time and lower the cost, we propose average distance models for various traffic patterns to help as analyze such systems. We also formulate a zero-load average distance model that accurately predicts the performance of deflection routing networks in the absence of data flow by capturing the av-

erage distance of a packet with spatial and temporal probability distributions of traffic.

The thesis research goals are to explore the design space of vertical integration for many-core applications, and to provide solutions to 3D technology challenges through architectural innovations. We believe the research findings presented in the thesis work contribute in addressing few of the many challenges to the field of combined research in many-core architectural design and 3D integration technology.

Contents

List of Figures

List of Acronyms

2D Two Dimensional

3D Three Dimensional

BW Bandwidth

CMOS Complementary Metal-Oxide Semiconductor

CPU Central Processing Unit

DLL Delay-Locked-Loop

dTDMA dynamic Time-Division-Multiple-Access

DRAM Dynamic Random Access Memory

FIFO First In, First Out

GB Giga Byte

GHz Gigahertz

GPU Graphic Processing Unit

HS Hotspot

KB Kilo Byte

IC Integrated Circuit

IP Intellectual Property

ISA Instruction Set Architecture

MB Mega Byte

MPSoC Multi-Processor System-on-Chip

NI Network Interface

NoC Network-on-Chip

PE Processing Element

PLL Phase-Locked-Loop

QDR Quad Data Rate

RTL Register Transfer Level

SDR Single Data Rate

SDRAM Synchronous Dynamic Random Access Memory

SiP System-in-Package

SoC Systems-on-Chip

TSV Through-Silicon-Via

VHDL VHSIC Hardware Description Language

List of Publications

Papers included in this thesis:

1. A. Y. Weldezion, M. Grange, A. Jantsch, H. Tenhunen, D. Pamunuwa, "Zero-Load Predictive Model for Performance Analysis in Deflection Routing NoCs", in *Journal of Microprocessors and Microsystems (JMM)*, Volume 39, Issue 8, pp. 634-647, November 2015, ISSN 0141-9331, 2015

2. A. Y. Weldezion, M. Grange, D. Pamunuwa, A. Jantsch, and H. Tenhunen, "A scalable multi-dimensional NoC simulation model for diverse spatio-temporal traffic pattern", in *Proceedings of IEEE International Conference on 3D System Integration (3DIC 2013)*, San Francisco USA, pp. 1-5, October 2013

3. A. Y. Weldezion, R. Weerasekara, H. Tenhunen, "Design Space Exploration of Clock-pumping Techniques to Reduce Through Silicon Via TSV Manufacturing Cost In 3D Integration", in *Proceedings of the 14th Electronics Packaging Technology Conference (EPTC 2012)*, Singapore, 2012.

4. M. Grange, R. Weerasekera, D. Pamunuwa, A. Jantsch, and A. Y. Weldezion, "Optimal network architectures for minimizing average distance in k-ary n-dimensional mesh networks", in *Proceedings of the Networks on Chip Symposium (NoCS 2011)*, Pittsburgh USA, pp. 57-64, May 2011

5. A. Y. Weldezion, Z. Lu, R. Weerasekera, and H. Tenhunen, "3-D Memory Organization and Performance Analysis for Multi-processor Network-On-Chip Architecture", in *Proceedings of IEEE International Conference on 3D System Integration (3DIC 2009)*, San Francisco USA, pp. 1-7, September 2009

6. A. Y. Weldezion, M. Grange, D. Pamunuwa, Z. Lu, A. Jantsch, R.Weerasekera and H. Tenhunen, "Scalability of network-on-chip communication architecture for 3-D meshes", in *Proceedings of the 3rd ACM/IEEE International Symposium on Networks-on-Chip (NOCS 2009)*, San Diego USA, pp. 114-123, May 2009

7. A. Y. Weldezion, R. Weerasekera, D. Pamunuwa , L. Zheng and H.Tenhunen, "Bandwidth Optimization for Through Silicon Via(TSV) bundles in 3D Inte-

grated Circuits", in *3D Integration Workshop, The Design, Automation, and Test in Europe (DATE) conference*,Nice France, April 2009

8. A.Y. Weldezion, M. Ebrahimi, M. Daneshtalab, and H. Tenhunen, "Automated Power and Latency Management in Heterogeneous 3D NoCs", in *Eighth International Workshop on Network on Chip Architectures (NoCArc)*, Waikiki, Hawaii, USA, December 2015.

Related publications not included in this thesis:

9. A. Y. Weldezion, R. Weerasekera, D. Pamunuwa, H. Tenhunen, "Performance of Scalable 3D On-Chip Networks Architecture: A Comparative Analysis", *Future Fab International Issue 34, pp. 26-29,* 2010

10. M. Grange, A. Y. Weldezion, D. Pamunuwa, R. Weerasekera, H. Tenhunen and D. Shippen, "Physical Mapping and Performance Study of a Multi-Clock 3-Dimensional Network-on-Chip Mesh". *In Proceeding of IEEE International Conference on 3D System Integration (3DIC 2009)*, San Francisco USA, September 2009.

11. A-M. Rahmanisane, H. Tenhunen, P. Liljeberg, A. Y. Weldezion, S. Kanduru, J. Plosila, M-H. Haghbayan, and A. Jantsch. "Dynamic power management for many-core platforms in the dark silicon era: A multi-objective control approach". *In Proceedings of the International Conference on Low Power Electronics and Design*, Rome, Italy, July 2015.

12. M-H. Haghbayan, A-M. Rahmani, A. Y. Weldezion, P. Liljeberg, J. Plosila, A. Jantsch, and H. Tenhunen. "Dark silicon aware power management for many-core systems under dynamic workloads". *In Proceedings of the International Conference on Computer Design*, Seoul, South Korea, October 2014.

13. W. Ahmed, L. Zheng, R. Weerasekera, Q. Chen, A. Y. Weldezion and H. Tenhunen. "Power Integrity Optimization of 3D Chips Stacked Through TSVs", *In Proceedings of the 18th IEEE Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS 2009)*, Portland USA October 2009.

Chapter 1

Introduction

Since the dawn of electronic computers in 1940s, architectural innovation and design of computer systems have shown considerable changes with the evolution of different technologies. *Scalability* and *performance* have been largely the driving factors behind these changes. With each new technology such as logic CMOS or DRAM, more robust and miniaturized systems are designed and new devices are produced. However, progress has not been the same with all technologies; the pace at which CMOS advances is faster than that of memory technology. This has led to a performance gap between a microprocessor in a CMOS chip sending data packets in every clock cycle to a DRAM chip placed in the same board. Consequently, major efforts of architectural innovations in the last three decades have been about narrowing the gap. One of the innovations that gained wide industrial adoption within short time is clock-pumping technique. Using double-pumping and quad-pumping techniques, two or four data packets can be sent in a single clock cycle respectively. This has led to the commercialization of high-performance DDR-SDRAM chips and recently QDR-SDRAM chips that are now basic components in the global semiconductor market. Even though far from closing the gap, clock-pumping technique has greatly improved the performance [72] [39] [45].

Furthermore, progress in CMOS technology to a billion-transistor era has resulted in a paradigm shift from off-chip centric to on-chip centric design methodology in which on-chip systems can be designed as an intellectual property (IP) cores. Dozens of cores are embedded in a single chip and are networked with each other through serial bus or parallel networks-on-chip (NoCs) architecture [38] [9]. And, such fabless design companies since late 1990s mushroomed as the suppliers of heterogeneous IP cores to the market heralding the era of many-core architecture embedded into a single chip. The many-core deals with how a stored program is distributed within the memory hierarchy for an efficient delivery of instructions and data to the multiple processors connected with a parallel bus.

Chips inside HDTV processors, mobile phones, high-performance computing processors, and embedded systems are some of the many-core applications that

contain tens of cores in a monolithic die. In particular, Networks-on-Chip (NoC) has been proposed through years of research as a scalable parallel bus solution for large many-core systems laid out on a single-die and is being adopted in an increasing number of cutting edge designs [69].

1.1 Networks-on-Chip (NoC)

NoC provides an infrastructure for better modularity, scalability, fault-tolerant and higher bandwidth as compared to the traditional bus approaches. It enables the integration of a large number of IP cores in a single chip [41] [84] [18]. The driving reasons for emergence of NoC as an architectural solution can be summarized as follows:

Push factor: technology scaling limitations lead into slowing the Moore's law and hence partitioning the cores and modification of the traditional von-Neumann architecture into parallel architectures.

Pull factor: Functional specialization of IP cores designed by different vendors leading into the need for creating a network and communication architecture for integrated functionality of the system.

On the technology side, chip fabrication technology is facing new challenges in the deep sub-micron regime [28]. As a result, the NoC design in 2D chips (2DNOC) has limited floor-planning choices which become a bottleneck when the number of processing elements increases. In addition, as the network spreads over a 2D plane, the transmission delay between two points increases significantly, which results in lower performance and higher power consumption [10]. To overcome these limitations, technology is moving rapidly towards the concept of vertical integration where multiple active silicon layers are stacked forming three dimensional integrated circuits (3DIC).

1.2 3D Integration Technology

Three-Dimensional integration technology is an emerging technology that enables stacking of dies using interconnects called Through-Silicon-Vias (TSVs). The key benefits of 3DIC come from its geometric and physical advantage. First, it leads to the improvement in signaling latency and energy [6] enabled by a reduction in the overall interconnect length. Second, it provides a way to bypass the interconnect bottleneck and meet the semiconductor industry's demands for higher memory-logic and I/O bandwidth [14] through reduced interconnect delay. It allows the continued increase of integration density. Finally, it also provides a means for integration of heterogeneous technologies: logic and memory. The key here is the availability of fast TSV, whose physical characteristics differ fundamentally from the on-chip counterparts, in that the resistance is much lower due to the shorter length and the relatively larger cross-section, and have a much smaller RC time constant than on-chip lines [75].

As shown in Figure 1.1, TSVs connect two or more dies horizontally or vertically. When dies are placed side by side on top of dummy interposer, TSVs connect horizontally. This has advantages because it enables low-cost heterogenous integration while using the traditional 2D process technology. When dies are stacked vertically, wafer to wafer or die-to-wafer bonding is possible. However, 3D stacking of dies with high power density suffers with exponential temperature increase risking thermal heating that damages the device [46]. A logic-to-memory stacking such as processor-memory integration is possible. Normally, compared to memory die, logic die is active and generates more heat. Therefore, it is placed on top of the stack, close to the heat sink and is connected with memory using TSV. But, for off-chip access, additional TSVs that pass through the memory layer are required [1] [54].

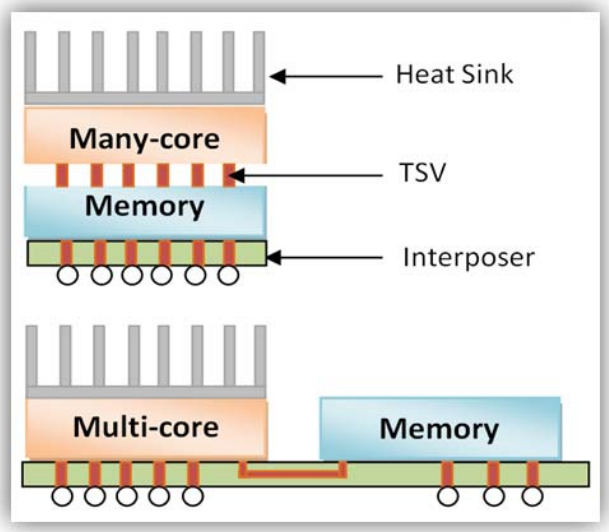


Figure 1.1: TSV as vertical and horizontal interconnects

1.3 Multi-core vs. Many-core

Even though it is implicitly understood, the difference between Many-core and Multi-core architecture is not well defined in literature. In the context the thesis work, we find it imperative to explicitly set the difference between these architectures based on the processing power as the number of cores increase.

It has been according to 'Moore's law' that microprocessor performance improves by increasing the clock speed in newer generation. But, since the beginning of last decade, the semiconductor industry is not able to keep up with the

law because the microprocessor power consumption skyrocketed with increase in performance, also known as power-wall. Designers found out that, with the contemporary technology scaling, two less complex processors (dual-core) in a single chip consume less power and perform better than a single complex processor (uni-core). In the same manner, four simpler processors (quad-core) in a single chip outperforms dual-core. The communication between the cores is done using serial bus network.

The individual processors in such *multi-core architecture* are made powerful enough to exploit instruction level parallelism (ILP) [53] [16] [39]. These processors are generally super-scalar and super-pipelined types and mostly multi-threaded. A degree of parallelism through ILP is achieved only when the performance in terms of cycles per instruction increases. The complexity of the cores is reduced as the number of cores increase. But, when the number of cores is doubled or quadrupled, though there is an increase in performance, the marginal gain starts to fall. Also, the bandwidth of the bus becomes a bottleneck in accommodating the growing number of cores.

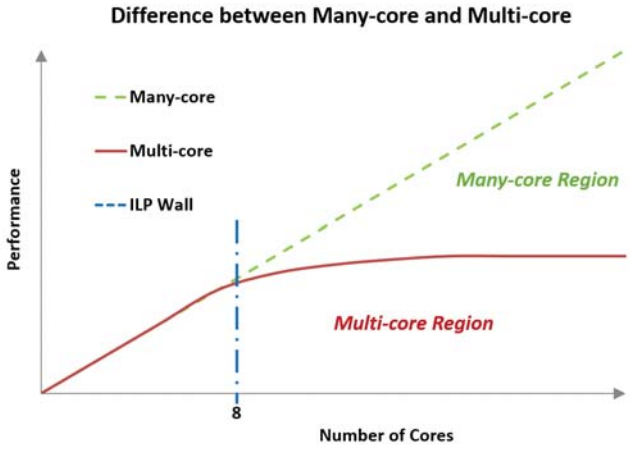


Figure 1.2: The ILP gives diminishing return in Multi-core

This behavior shown in Figure 1.2 is the diminishing return of performance of ILP in multi-core architecture. The breaking point where the ILP performance gain starts to fall with multi-core is the ILP wall. With the contemporary multi-core processors, we assumed the average number of cores for ILP wall is around eight core. Beyond 8-cores in a single-chip, each individual core is reduced to its simplest form as simple-scalar architecture or as a specialized core that efficiently executes specific applications such as video or audio processors. The communication in such *many-cores architecture* is done through parallel network.

The many-core is an alternative to multi-core and uses dozens of cores inter-

connected with parallel networks such as mesh topology in order to generate an aggregate gain in the performance. In many-cores architecture, performance can be gained through increasing the number of simple-scalar cores, through improving the efficiency or specialized cores, or through enhancing the communication among the cores.

1.4 Thesis Motivation

Many-core processors such as Intel Teraflop with 80-cores [69] and Tilera 64-core processor [8] are built as System-On-Chip (SoC). Each core is a simple scalar processor physically connected to routers of a Networks-on-chip (NoC) architecture through which data packet communication among the cores and other resources is facilitated. By considering recent scalability trends, it is believed that in the next ten years many-core processors will have 1000s of cores in a single chip. However, such progress is not without limitations and challenges including high power consumption, form-factor, and high-level issues such as programmability, operating system, and memory consistency issues.

To meet these challenges, 3DIC using TSV is recently gaining momentum as a technology for many-core architecture implementation [73]. The key driver to scalable vertical integration is TSV, which has emerged as state-of-the-art technology. However, its benefits for the use in many-core architectures are not yet fully explored. This is mainly due to the fact that both NoC and 3DIC are relatively recent research frontiers. We believe that there is an immense potential to be tapped by the fusion of these frontiers. In fact, our research group at KTH has been working for a 3D memory related project ELITE funded by EU-FP7 [26]. In the project, we developed TSV models, vertical networks architecture, scalable 3D simulators, and also made several circuit level and architecture level experiments for many-core computer architecture.

A simple conceptual model of 3D many-core computer architecture shown in Figure 1.3 is a relevant example of vertical integration of heterogenous technologies. Here, the heterogeneity is manifested in different ways. There is *architectural heterogeneity* in which the cores are with different physical sizes and varying functional specialization often made by a third party fabless companies. *Heterogeneity in network* may occur when cores are connected with irregular topology. There is *heterogeneity due to process node variation* among the layers and dies to be integrated (For example, one layer processed in 65nm node can be integrated with another layer processed in 22nm node technology). There is also *heterogeneity due to dissimilar technologies*. As an example, the conceptual model shows standard CMOS technology, distributed DRAM layer, and NAND Flash memory technologies stacked in a single chip.

Design and implementation of such massive integration for the use of many-core applications is complex. In fact, it is a major 3D integration challenge by itself [73]. We believe that an integration-centric approach where networks are

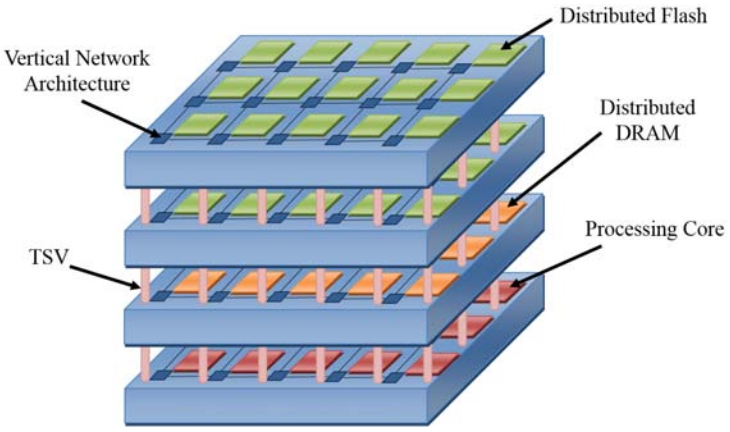


Figure 1.3: A conceptual model of 3D many-core computer architecture

utilized as integration fabric should be followed to design such complex systems. To implement such design approach, currently there is a lack of integrated design tools and standards due to many constraints.

Firstly, the design complexity keeps increasing with the degree of integration of the heterogenous technologies.

Secondly, the high-cost of 3D chip fabrication makes design prototyping limited to running simulators and analyzing simulation results.

Finally, experiences in 2D mesh networks with hundreds of cores show that extracting results by running software and RTL simulators by itself is tedious and time-consuming process. The problem worsens when the number of variables to be extracted increases, and when the complexity level of integration goes from 2D mesh to 3D cube network.

This motivates us to investigate architectural and performance limits of NoCs integrated in 3D, NoCs in irregular network of hundreds of cores, or NoCs in designing a many-core architecture.

1.5 Thesis Objectives

In the context of the thesis work, *scalability* is defined as the ability of a system to adapt to qualitative and quantitative changes with a predictable and enhanced outcome. In general, systems can scale up vertically or horizontally. In our case, in order to explore the integration challenges of heterogenous technologies and systems through experiment, it is imperative to first develop a scalable 3D NoCs platform for many-core architecture.

The NoCs is used as a backbone for the cores to communicate with each other as well as with other off-chip resources. By combining the approach of System-on-

Chip design with the new features of 3D integration technology, we use the NoCs platform to serve for the following thesis objectives.

1 To model and develop an integrated simulation environment for deflection routing networks in the context of designing heterogenous many-core architectures.

2 To investigate the limits and potentials of architectural scalability of deflection routing networks through performance evaluation in massively integrated many-core architecture.

3 To evaluate the NoCs performance when used as a shared communication resource replacing the memory system bus in various configurations of processor-memory models.

4 To show ways of exploiting the benefits of fast TSVs to enhance the overall performance of scalable many-core architecture.

5 To develop analytical models for performance evaluation of heterogeneous many-core architectures during design space exploration.

1.6 Thesis Organization and Author's Contribution

The thesis is organized into six chapters summarizing with logical build up to meet the thesis objectives. Each chapter describes the main contributions while results are mostly reported in the published papers. Chapter 1 introduces the background to the research topic. Chapter 2 discusses a NoC simulation platform developed to carry out experiments to validate models and processes proposed in the thesis. Chapter 3 summarizes the performance evaluation of NoC architectures when scaled in size to support 1000 core architecture, and when topologically configured in 3D processor-memory architecture. In Chapter 4, we look at new design schemes that enhance the performance of inter-layer communication in the third dimension by exploiting the unique features of through-silicon-vias. Chapter 5 summarizes a performance prediction model proposed for integrated design of many-core architecture. Finally, Chapter 6 concludes the thesis with discussion on future prospects of the research studies.

The author's contribution in each chapter is reflected through publications of peer reviewed papers and can be summarized as follows.

Chapter 2

On-chip-networks of realistic applications are irregular networks with heterogeneous traffic. The resource sizes are not the same. Also, with the emerging 3D many-core architecture, the irregularity and heterogeneity of NoCs increases. However, the standard modelling and simulation approaches are made based on regular, homogeneous 2D networks. In this chapter, a scalable and multi-dimensional simulator

that captures the irregular and heterogeneous network properties is proposed. 2D and 3D router micro-architecture using TSV are developed. Network monitors are also added to check runtime activities. By combining the geometric size and the traffic contribution of cores, power and performance simulations are performed. Link utilization, buffer size monitoring, fault monitoring information can be extracted and analyzed. Moreover, the model can be easily configured to simulate specific applications such as many-core processor memory stacking.

Part of the work is published in:

- A. Y. Weldezion, M. Grange, D. Pamunuwa, A. Jantsch, and H. Tenhunen, "A scalable multi-dimensional NoC simulation model for diverse spatio-temporal traffic pattern", in *Proceedings of IEEE International Conference on 3D System Integration (3DIC 2013)*, San Francisco USA, pp. 1-5, October 2013.

  The author's contribution: The author proposed a multi-dimensional simulator that captures the heterogeneous network properties. The author developed a 1000-core simulator, router micro-architecture, and traffic pattern models for regular and irregular deflection routing networks and was responsible for the overall implementation of the simulation framework and writing of the manuscript.

- A.Y. Weldezion, M. Ebrahimi, M. Daneshtalab, and H. Tenhunen, "Automated Power and Latency Management in Heterogeneous 3D NoCs", in *Eihth International Workshop on Network on Chip Architectures (NoCArc)*, Waikiki, Hawaii, USA, December 2015.

  The author's contribution: The author proposed and implemented Q-routing algorithm for 3D NoCs, look-up-table based routing for irregular networks, and network monitors to extract runtime power estimations within the simulation framework. The author also contributed in the writing of the manuscript.

Chapter 3

Scalability of NoCs in designing massively integrated many-core architecture is a key issue to ensure ideal on-chip communication as the systems grows. This chapter summarizes the investigations done to examine the performance of deflection routing NoCs, when scaled up by extending from 2D to 3D architecture using TSVs for inter-die connectivity. The scalability of 3D cube networks is evaluated under various traffic scenarios and compared to 2D mesh networks in order to measure the performance limits when designing 3D many-core architectures. We also look at specific applications by proposing the use of deflection routing NoCs as a medium of communication for various 3D processor-memory configurations. Cycle accurate simulations are done to evaluate the performance of each proposed model in uniform and local traffic patterns.

Part of the work was published in:

- A. Y. Weldezion, M. Grange, D. Pamunuwa, Zhonghai Lu, A. Jantsch, R. Weerasekera and H. Tenhunen, "Scalability of network-on-chip communication architecture for 3-D meshes", in *Proceedings of the 3rd ACM/IEEE International Symposium on Networks-on-Chip (NOCS 2009)*, San Diego USA, pp. 114-123, May 2009.

  The author's contribution: The author contributed in the modelling of links and metrics that would be used to analyze NoCs scalability and its effect to the performance. The author was responsible in the implementation of the models, in extending 2D-to-3D router using TSV, developing network structure and traffic pattern models, and was responsible for the simulation, data gathering, analysis, and writing of the manuscript.

- A. Y. Weldezion, Z. Lu, R. Weerasekera, and H. Tenhunen, "3-D Memory Organization and Performance Analysis for Multi-processor Network-On-Chip Architecture", in *Proceedings of IEEE International Conference on 3D System Integration (3DIC 2009)*, San Francisco USA, pp. 1-7, September 2009.

  The author's contribution: The author proposed various 3D processor-memory model configurations. The author was responsible for the implementation of the models, and cycle accurate simulations to evaluate the performance of each model in uniform and local traffic patterns simulation framework, and writing of the manuscript.

Chapter 4

The communication architecture within a massively integrated many-core systems has to be designed to make optimal use of the underlying physical properties of the horizontal and vertical interconnects, in order to maximize the potential performance benefits afforded by 3-D integration. In particular, the inter-layer communication using vertical interconnects can be greatly enhanced by exploiting the unique features of TSVs. As a short and fast interconnects, TSVs can be clock-pumped to generate multi-rate data communication. In this chapter we propose double-data-rate (DDR) TSV models to enhance the NoCs performance. The proposed models are validated through cycle accurate simulation of symmetric and asymmetric networks.

Part of the work was published in:

- A. Y. Weldezion, R. Weerasekara, H. Tenhunen, "Design Space Exploration of Clock-pumping Techniques to Reduce Through Silicon Via TSV Manufacturing Cost In 3D Integration", in *Proceedings of the 14th Electronics Packaging Technology Conference (EPTC 2012)*, Singapore, 2012.

  The author's contribution: The author proposed double-data-rate (DDR) TSV models to enhance the NoCs performance. The author implemented router micro-architecture to support the proposed model, validated through several experiments, and prepared of the manuscript.

- A. Y. Weldezion, R. Weerasekera, D. Pamunuwa, Li-Rong Zheng and H. Tenhunen, "Bandwidth Optimization for Through Silicon Via(TSV) bundles in 3D Integrated Circuits", in *3D Integration Workshop, The Design, Automation, and Test in Europe (DATE) conference*,Nice France, April 2009.

  The author's contribution: The author proposed analytical methods to optimize the total bandwidth of TSV bundles placed in a structure with a fixed area and length. The author was also responsible for the writing of the manuscript.

- M. Grange, A. Y. Weldezion, D. Pamunuwa, R. Weerasekera, H. Tenhunen and D. Shippen, "Physical Mapping and Performance Study of a Multi-Clock 3-Dimensional Network-on-Chip Mesh". In Proc. IEEE International Conference on 3D System Integration (3DIC 2009), P.P 1-7, San Francisco USA, September 2009.

  The author's contribution: The author contributed in the conceptual development of Multi-clock model, and was responsible for the implementation of the simulation framework.

Chapter 5

The massive integration of many-core architecture gives space for unlimited design exploration in terms of defining network topologies, heterogeneity, and traffic scenarios. However, determining performance of any NoC design in many-core architecture is a challenging task due to the limited power of existing analytical performance models and in the absence of comprehensive simulation platforms. In this chapter, we propose an average distance model for deflection routing NoCs that accurately captures the average distance of a packet in any given topology considering the spatial and temporal probability distributions of traffic. By using the average distance model, we predict the relative performance of deflection routing networks. The validity of the model is verified through cycle-accurate simulations under various traffic patterns such as uniform random, localized, hot-spot, bursty, and others.

Part of the work was published in:

- A. Y. Weldezion, M. Grange, A. Jantsch, H. Tenhunen, D. Pamunuwa, "Zero-Load Predictive Model for Performance Analysis in Deflection Routing NoCs", in *Journal of Microprocessors and Microsystems (JMM)*, Volume 39, Issue 8, pp. 634-647, November 2015, ISSN 0141-9331, 2015.

  The author's contribution: The author extended the use of average distance as a zero-load performance predictive model for irregular networks in many-core architecture. The author was also responsible for the implementation of the model in different scenarios including regular and irregular networks and traffics, and for writing of the manuscript.

- M. Grange, R. Weerasekera, D. Pamunuwa, A. Jantsch, and A. Y. Weldezion, "Optimal network architectures for minimizing average distance in k-ary n-dimensional mesh networks," in *Proceedings of the Networks on Chip Symposium (NoCS 2011)*, Pittsburgh USA, pp. 57-64, May 2011.

  The author's contribution: The author contributed in the conceptual development average distance models for performance analysis. The author was also responsible for the implementation of router micro-architecture, network structure, and traffic pattern models used to validate the proposed model.

1.7 Thesis Navigation

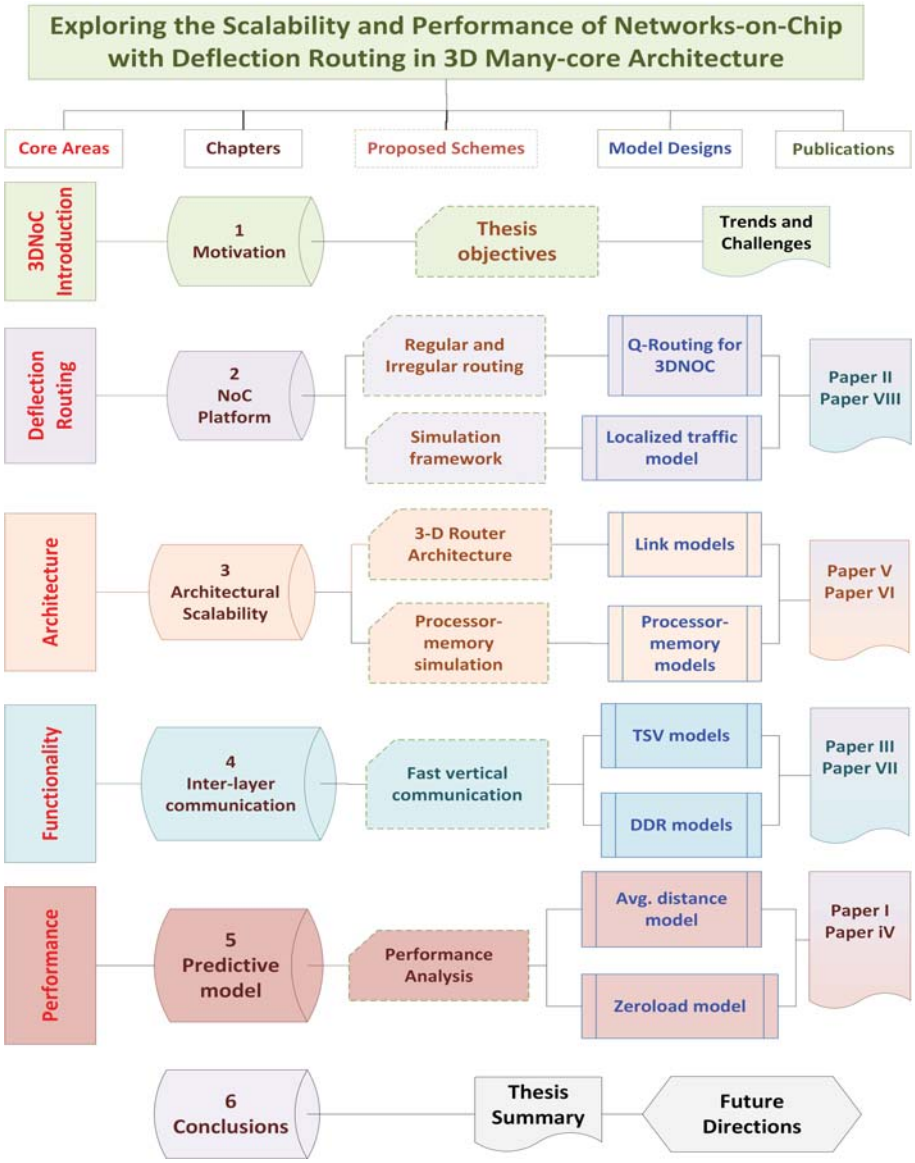


Figure 1.4: Navigation of the dissertation

Chapter 2

Scalable NoC Simulation Platform

This chapter discusses research and development of a deflection routing NoC platform, specifically the extension of 2DNoC into 3DNoC architecture in the context at which we carried out all experiments described in the thesis. We explore challenge in designing router micro-architectures and routing algorithms that are scalable enough to accommodate 1000s of cores in heterogenous many-core architecture.

First let's consider a system consists of processor (P) and memory (M) cores connected with a traditional bus network as shown in Figure 2.1. The cores communicate with each other by sending data packets through the bus network. Since there is only a single bus to serve all the cores, only one packet can occupy the bus at a time. Thus, each core is served serially through time slicing implemented as part of bus arbitration mechanism. However, with each additional core, the waiting time to get the service increases leading to bandwidth limitation and performance degradation.

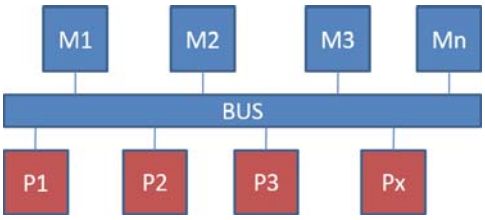


Figure 2.1: Traditional bus network

As an alternative to the traditional bus with inherent bottleneck, designs are driven to use more advanced and scalable networks with *intelligent routers* which implement routing algorithms to move packets inside the network through channels called *links*. In our context, such network is provided as an on-chip solution - *NoC*. As shown in Figure 2.2, the way NoC is configured in parallel network allows a packet from a source core to be routed to a destination core concurrent to other

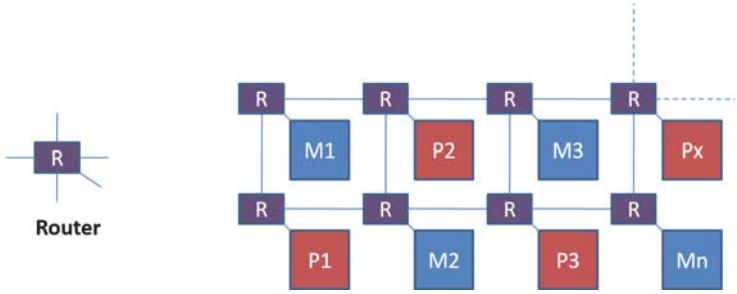


Figure 2.2: Parallel network with routers

packets. The basic NoC configuration in many-cores is therefore based on satisfying efficient delivery of data in parallel. The performance is measured in different metrics including in terms of *latency* it takes for a packet to travel the source-to-destination core distance and the overall *throughput* of the network.

The destination core can be the same source core (*selfcast*), any one of the other cores (*unicast*), a few selected cores (*multicast*), all cores in the network (*broadcast*) [24].

To identify the intended destination cores from others, a unique address is allocated for each core. When a packet is routed to a destination specified with an address, the routing of the packet along the source-to-destination path is controlled through different methods depending on the place where decisions are made. One method is to manage all packets in the network from one *centralized* controller. A second method is to divide the network into *hierarchy* of regions where a router can have a control in its own region but not in other regions. A packet is then routed based on the regional hierarchy of source and destination cores. A third method is to make *distributed* decisions by localizing control mechanism at routers level. Finally, a *hybrid* of the above control mechanisms can be used in different ways.

A NoC can grow to accommodate a network traffic generated by hundreds of cores. Designing and prototyping such networks requires a multi-disciplinary design team with design tools that significantly increase research cost and overall time-to-market. Thus, heavy use of NoC platforms and simulation based analysis becomes imperative. Simulation environments reported in [40] [85] [86] [32] proposed NoC simulation models for specific applications or configurations. However, the reported design approaches are not comprehensive and scalable to accommodate the emerging integration of heterogenous systems. Even, extracting results by running simulators for a 2DNOC with hundreds of cores by itself is tedious and time-consuming process. The problem worsens when the level of integration complexity increases, for example from 2DNOC to 3DNOC.

In this chapter, we show a scalable cycle-accurate simulation model proposed to addresses the design and simulation challenges of networks characterized by a high level of complexity and heterogeneity. Despite the necessity of heterogeneous

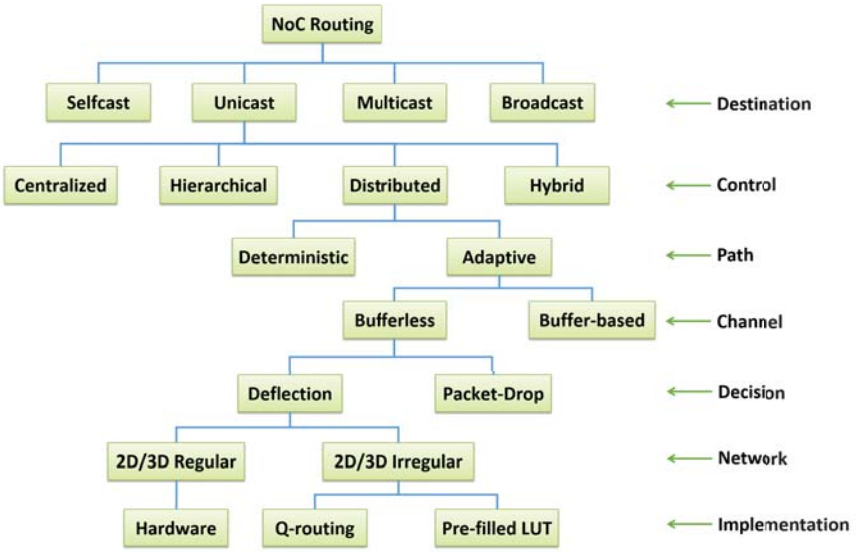


Figure 2.3: Classification of the implemented routing algorithms

network designs in many-core architecture, the main issue is that the network performance cannot be easily optimized. Specifically, the design of an efficient and deadlock-free routing algorithm for 3D networks is challenging. This is due to the fact that cycles can be formed between and within layers. Designing an efficient routing algorithm is nearly non-practical in irregular networks. This implies that heuristic approaches should be applied for each topology configuration not only to provide a deadlock-free communication between the cores but also to reach a satisfactory level of performance and low power consumption. Proposing a heuristic approach for different configuration of heterogeneous 3D NoCs imposes huge costs and engineering efforts.

To overcome the complexity of designing high-performance routing algorithm in 3D NoCs, a practical solution is turning toward deflection routing which is well compatible with irregular networks and provides a full adaptivity in routing packets. To guarantee a high performance and low power consumption under any topology configuration, an intelligent Q-routing algorithm is proposed in this thesis. The combination of Q-routing algorithm in buffer-less network offers a general high-performance approach for heterogenous 3D designs.

Depending on the way how packets are routed, Figure 2.3 shows the deflection routing NoCs employed in the proposed simulation model. The model can be used early in the exploration phase of the design. It supports up to 1000 core architecture in both regular and irregular networks. Part of the results are already published in Paper II [80] and in Paper VIII [82].

2.1 2DNOC

A two-dimensional Networks-on-Chip design, shown in Figure 2.4, consists of a grid of 2D routers connected to each other through *links*. A 2D router has five ports; four of them are *directional ports* connected to the *north*, *south*, *east*, and *west* neighboring routers in addition to one *local port* connected to a local core (a core can be a general-purpose processor, a DSP, or a memory subsystem). A *Network Interface (NI)* is used as an interface connecting a core with its local router. The fundamental function of a network interface is to provide communication service between cores and the network infrastructure. That is, the network interface translates the language between the cores and routers based on custom made or standard communication protocols such as AXI [4], OCP [60], and DTL [66]. A generic plug and play network interface architecture allows any NoC enabled core to be attached to the NoC [47]. The network interface design is generally similar for 2D and 3D NoC designs [25] [20].

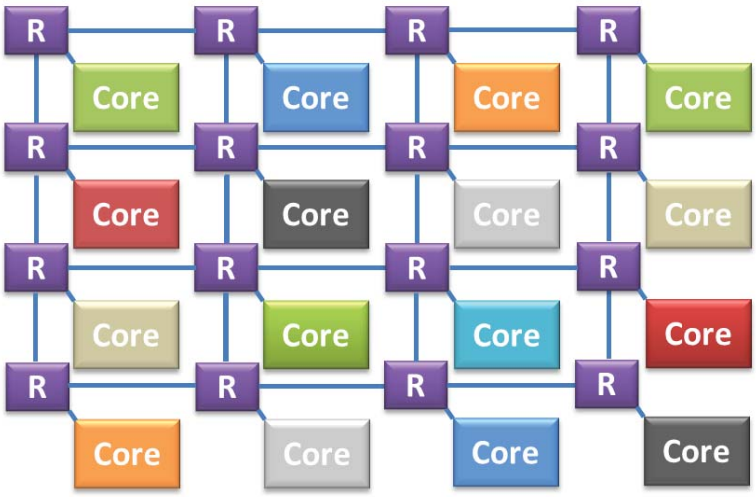


Figure 2.4: Two-dimensional Networks-on-Chip model

2.2 3DNOC

A 2DNoC can be extended to 3D systems in a straightforward way and is shown to be an efficient platform for network sizes with hundreds of cores [81]. As shown in Figure 2.5, the third dimension enables the vertical stacking of cores as layers.

Generally, the combination of NoC and 3DICs (i.e., 3DNOC) provides the major advantages of considerable reduction in the average wire length and wire delay, resulting in lower power consumption and higher performance [28] [55] [65] [62].

A 3DNOC uses 3D routers for inter-layer communication. A 3D router extended from a 2D router has two additional, *up* and *down* ports giving a total of six directional ports in addition to the local port.

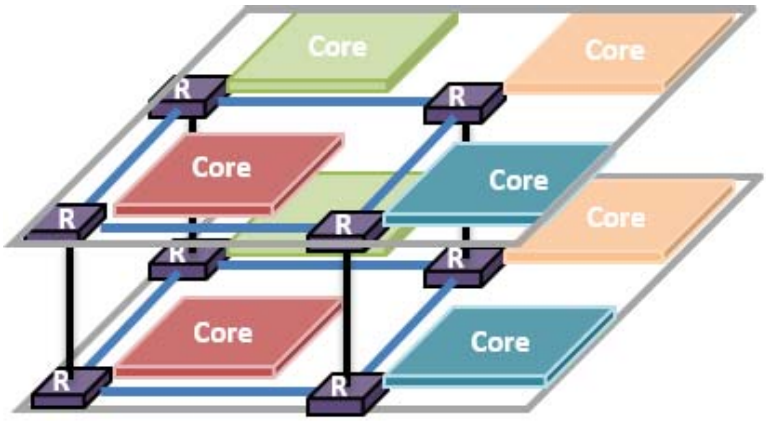


Figure 2.5: Three-dimensional Networks-on-Chip model

2.3 Routing Algorithms

When defining the path from the source to the destination core, different routing algorithms can be used as shown in the classifications tree in Figure 2.3. Once the destination is known, a packet can be routed through a pre-defined fixed path or *deterministic routing* [2]. On the other hand, there is also *adaptive routing* in which a packet can be routed through any path selected during run-time at each router, an important feature specially considering scalable networks [23]. However, to reduce the complexity of the routers, deterministic routing is favored over adaptive routing in buffered networks [51] [19] [6] [18].

A *buffered* network means a network contains *buffers* in its channels to temporarily store any packet unable to find a port on its path towards the destination. When the port is available, the packet takes over and continues to flow to the next port. On the other hand, *bufferless* networks do not allow any packet to stop its flow through the channels.

In bufferless networks, a packet unable to find a port that leads to the destination is either misrouted away from the path (*deflection routing*) or *dropped* altogether. Generally, in bufferless networks, deflection routing is advocated [68] because it is possible to design fast and small routers with simplified control circuitry. Also, packet-dropping leads to a lossy transmission which adds more complexity to packet management [37] [31].

Deflection routing (also known as hot-potato routing) is a well-studied topic in the on-chip network field. Originally the term was used in [7], where the deflection

routing is utilized in distributed communication systems to decrease the contention by sending packets through redundant links. Moscibroda et al. in [57] has shown the power/performance aspect of buffer-less routers using deflection routing as compared to buffered ones. This paper, similar to other works, makes the common assumption that the relative position between source and destination routers can be computed a-priori in a regular network hence it employs *regular routing* algorithm implemented as a *hardware*. Deflection routing employed in our model provides more flexibility in routing packets by supporting both regular and irregular routing and offering an inherent resilience against faults in routers and links. The main properties of deflection routing can be described as:

Deadlock-freedom: Designing deadlock-free routing algorithms has been always a major topic as it is a main factor in achieving efficiency. To prove deadlock freedom in networks, some routing limitations are applied by the means of turn models [21]. These limitations, even small, strongly limit the flexibility of routing packets. For example, a single forbidden turn prevents packets to take some minimal paths and consequently a large number of non-minimal routes. Routing algorithms are specially becoming very challenging in irregular networks or when some routers or links are disabled in regular networks. Based on the observation in [87], a single faulty router drops the performance to half in a 4×4 regular mesh network as the routing algorithm cannot efficiently adapt itself to the new configuration.

Deflection routing is inherently deadlock free. A message is divided into single flits and each packet covers one flit which can be routed through any possible path, either minimal or non-minimal. The priority is always given to the output ports leading to the shortest paths, however if not possible, the packet can be sent out through any other output ports. The priority of output ports is stored in a table which can be filled out at the design time. As packets are never blocked, a deadlock-free network is guaranteed.

Live-lock-freedom: Priority schemes in the arbitration unit avoid the live-lock situation. The common priority schemes are age-based priority and source-based priority. In the age-based priority scheme, if a packet takes a longer time than the threshold value, its priority (i.e. ability to take any preferred output port) increases. In the source-based priority scheme, on the other hand, as a packet moves away further from the source, its priority increases.

Deflection routing is inherently fault tolerant: Irregularity is often indirectly discussed in the fault-tolerant domain where by occurring faults in routers or links the network topology may change from a regular to an irregular one. In general, fault-tolerant methods offer solutions to protect the network in the case of faults but they make less attention to the performance of the irregular network. For example, turn models may cause a router to be unreachable even though the router is still connected to the neighboring routers through some channels. Thanks to the characteristics of deflection routing, as long as a router has at least one remaining physical connection, the router is reachable. The presented approach in [12] suggests solutions to design hardware-efficient routing methods for an irreg-

ular mesh. The focus of this approach is on static shortest-path routing. In our proposed approach, packets are able to choose any possible routes in the network, either minimal or non-minimal with the given priority on minimal paths.

Compatibility with heterogeneous networks: Deflection routing is fully compatible with irregular networks. In heterogeneous systems with unequal core sizes and links length, the type of network topology that can be mapped is irregular. The deflection routing can be directly implemented to such irregular networks without any particular modifications. Under any topology configurations, all packets can be delivered to the destinations. However, to improve performance, the priority of output ports can be adjusted according to the shortest paths.

2.4 Q-routing

A packet in irregular networks can't be routed towards the destination with regular routing algorithm implemented in a hardware because the source-destination path is unknown. In irregular networks, all possible source-destination paths must be recorded in a look-up-table (LUT) defined in each router. Each LUT has an array of rows filled with values of destination addresses. For each destination, the corresponding distance from the current router through each port is filled in the column. In this way, when a packet is routed, its destination is read and the corresponding port value with the shortest distance towards the destination is selected. The LUT value filling can be done in two ways: In *Pre-filled LUT*, the values are filled statically during the network design phase.

In *Q-routing*, a Q-learning algorithm is implemented to fill the values dynamically. When the packet is sent from current router to next router, a new estimation on the distance from the current router to the destination is obtained. The new estimated distance can be calculated at the next router and sent back to the current router. Upon receiving this new estimation by the current router, the corresponding entry in the Q-Table will be updated. The current entry of the router refers to the one associated with the destination as the row and the output port connecting the current router to the next router as the column.

Q-Routing based models have been studied in different domains [13] [43], but they have rarely been investigated in the context of on-chip networks. The algorithm in [52] is proposed to handle communication among modules which are dynamically placed on a reconfigurable NoCs. This algorithm is inspired by the method in [13] for a general case of packet switching networks. FTDR-H [30] utilizes Q-Routing methods to tolerate faults and find a path between each pair of routers as long as a path exists. Moreover, the size of Q-Tables is reduced by taking advantages of the clustering model. The clustering model is also extensively discussed in the C-Routing method [63]. Bi-LCQ [82] applies the Q-Routing method on a cluster-based NoCs. All of the aforementioned works are presented in the realm of 2D network.

Our deflection routing model supports both Pre-filled and Q-Table LUT in 2D and 3D networks. More details are given in Paper VIII [82].

2.5 Switching Policy

A *flit* is a set of data that can be delivered in parallel. A *packet* is a chain of flits ranging from *1 to n*. A collection of packets forms a *message* ready to be delivered from a source core to a destination core. The data can be *control information* such as destination address, or the actual *payload data* intended to be transmitted over the network.

When a packet with n chain of flits is injected to a network, each flit is switched from one router to the next without breaking the chain. In order to keep the integrity of the chain and to facilitate the switching activity of the packet, a flow control mechanism is defined. Normally, the first flit in the chain called *header* contains the control information, followed by a set of *body* flits with payload data. The following mechanisms are commonly used switching mechanisms.

Wormhole (WH) switching allows the packet to flow at the same time. When the header flit hops from current router to the next router, the next flit occupies the current routers. If the header is stalled, all other body flits are also stalled at the same time. *Store-and-forward (SAF)* allows the packet to be first fully stored in a single router and then forwarded to the next router. When the header flit hops to the next router, the whole body flit follows one by one. After making sure the whole packet is stored in the next router, then the header flit continues its journey in the same manner. *Virtual-cut-through (VCT)* is a mix of WH and SAF. In VCT, the header flit flows like WH, but when stalled it acts like SAF - the body flits continue to flow to be stored in the router where the header stopped.

More details of the switching mechanisms can be found here [11]. However, as this study is not intended to explore these switching mechanisms, they are not considered in depth.

In deflection routing, we use a single flit switching in which a message is divided into packet and each packet is a single flit long. It is a hot-potato implementation with router architecture as described in [58]. A packet shown in Figure 2.6 carries both the control information and payload data in an n-bit wide flit.

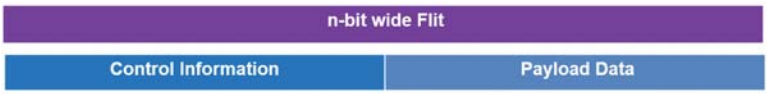


Figure 2.6: An n-bit wide single flit packet model

A Packet can be classified into different types based on its payload.

Data packets are the general purpose packets that carry data from processors to memory or IO.

Instruction packets are these that are read by a router to change its functionalities based on the instruction stored in the packet. For example, an instruction packet can be sent to all routers in a network to change their setting to congestion-aware routing.

Monitor packets can be used to check the run-time activities of the network. For example, congestion monitor, power monitor.

Test packets can be sent for a round-trip source-destination journey in the network. For example, we can test the network latency or if a destination is faulty.

Acknowledgement packets can be used whenever there is a request for acknowledgement as a proof of reception. For example, memory cores must send an acknowledgement packet at the end of writing data packets.

Other types of packets can be defined on need basis. Once injected into the network, a packet is by default sent to an output port that leads to a path with the minimum distance towards the destination. If the output port is not available due to faulty link or contention, the packet is deflected to the next best output option. There are two main drawbacks in single-flit switching.

1. Control information is repeated per flit than per packet compared to wormhole switching and other switching with long chain of flits, and thus imposing redundant control information.

2. Flits should be reordered to reconstruct the final message after arrival at network interfaces incurring additional latency outside the network.

2.6 Router Micro-architecture

Router micro-architecture designs may offer different characteristics for NoCs and affect the performance, area, and power consumption. Depending on the design, a router can be a single cycle or multi-cycle. In a single cycle, a packet passing through a router needs only one clock-cycle to hop to the next router [29]. In multi-cycle architecture, a packet passes through a number of pipeline stages inside the router in order to hop to the next router. In our work, these pipeline stages are implemented in the router logic circuitry and by segmenting long wire links. Pipelining allows the router to run in higher clock speed which leads to an over-all performance increase.

A 3D router shown in Figure 2.5 is basically an extension of 2D router with additional vertical ports for inter-layer communication. The router micro-architecture is buffer-less and enacts a fully adaptive, non-minimal deflection routing algorithm. A packet is only a single flit long and comprises control and payload bits. A hop for a packet is counted when it traverses the link from one router to the next. In the case where two or more packets compete for the same link, the router honours an oldest-first priority scheme. This is done in the sorting block. The output ports for each packet is selected according to the destination address.

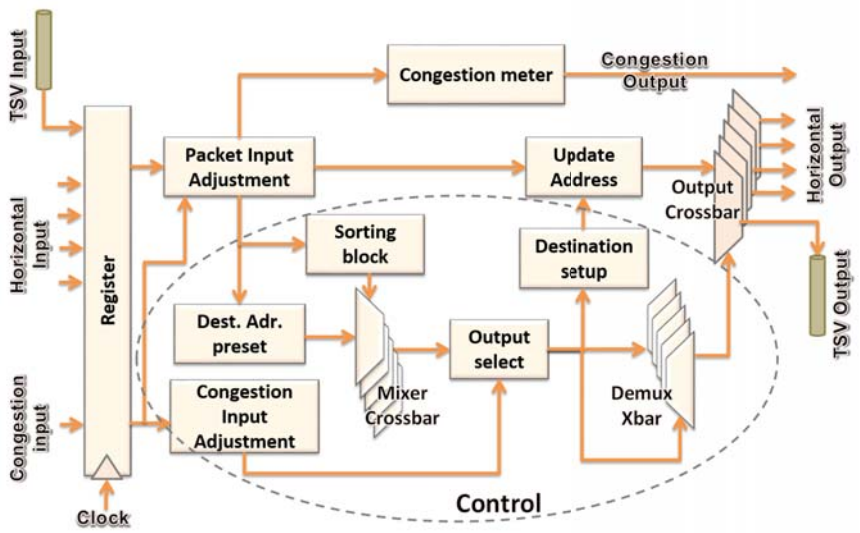


Figure 2.7: A 3D Router micro-architecture with vertical ports to TSV extension for inter-layer communication

No packets are dropped from the network. Instead, when the network is congested, the packets are accumulated in FIFO buffer in the network interface (NI) situated between each router and its local processing element. There is also a congestion meter that monitors and sends the local traffic to the neighboring routers. The router can be pipelined into stages. For regular networks, a relative addressing scheme is implemented which simplifies the duplication of identical routers when network structures of varying sizes are designed. More details of the routing protocol and router micro-architecture are given in Paper II [80]. For irregular networks, look-up-table based addressing scheme is implemented, more details are given in Paper VIII [82].

2.7 Topology

A topology determines the arrangement of routers and links. Different topologies have been proposed so far for 2D NoCs, such as Ring, Mesh, Torus, and Butterfly. A 2D NoC architecture based on mesh topology has been widely used in regular networks [9] [38] [44]. A 3D cube topology can be made by vertical extension of 2D mesh. Various on-chip network topologies have been studied for 3DNOC [28] [48]. Mesh-based structures are popularly used in 3D systems as their grid-based regular architecture is intuitively considered to be a proper approach for the 2D VLSI layout for each stack layer. Nevertheless, if the number of IP cores and memories increases in each layer, more TSVs are necessitated to handle the inter-layer communication.

In as much as each TSV employs a pad for bonding, the area footprint of TSVs in each layer is augmented significantly [28] [61]. In heterogenous system design, however, the topology might be different in various layers, for example, one layer may have a mesh-based NoC while another layer may take a connectivity form of ring-based NoC leading into an irregular network. In irregular systems, due to the difference in the shape and size of cores, length of each link connecting the routers is not the same. In addition, faults in router and links may disturb the regularity and change the topology to an irregular type [22].

2.8 Traffic Generator

Network architecture is optimized primarily to regulate smooth flow of packets. In realistic scenario, these packets are generated by the processing cores and injected to the network. Inside the network, the flow of packets creates a traffic distribution with temporal and spatial properties that can be synthetically modeled as a pattern. In temporal distribution, the timing of packets over period is regulated where as in spatial distribution the variance of destination address within the network is set.

The simulation platform has a number of built-in traffic patterns generator. These traffic patterns can be used to stress-test a design under realistic scenarios so that the network configuration and placement strategies can be defined to produce the maximum performance and efficiency.

2.8.1 Spatial Distribution

Spatial traffic specifies the destination address in the network that is set according to a synthetic traffic pattern in use. In our experiments we utilize the following commonly used deterministic and probabilistic traffic patterns: uniform random (URT), bit reverse (BRT), bit complement (BCT), and local random (LRT) traffic.

For the purpose of defining spatial traffic patterns, cores are assigned unique numbers $S = 0 \cdots N-1$ with N being the number of cores. In a 3D cube topology the x, y and z address components are mapped from these core identifiers as follows:

$$\begin{aligned} x &= S \bmod N_x \\ y &= (S \operatorname{div} N_x) \mod N_y \\ z &= S \operatorname{div} (N_x N_y) \end{aligned} \tag{2.1}$$

where div is integer division and N_x, N_y, N_z denote the size of the network in each dimension. For a 2D mesh the same equations hold except for the third, which becomes irrelevant.

Uniform Random Traffic (URT)

In uniform local traffic the destination addresses are generated randomly as any processing element across the network other than the source. For a given network

size of N cores, URT creates a uniformly distributed spatial pattern, with equal destination probabilities for all source-destination pairs:

$$P_D = \frac{1}{N-1} \tag{2.2}$$

Bit-Reverse Traffic (BRT)

In BRT, the destination address is mirror image of the source address. The destination address is formed by reversing the binary format of the source core identifier as defined in Equation (2.1). For example, source core (001110) will send all its packets to destination core (011100).
Let ς_n denote the bit-reverse of n, (e.g. $\varsigma_{100} = 001$), S the source core identifier, D the destination core identifier, and $S_x, S_y, S_z, D_x, D_y, D_z$ the address components of the source and destination cores respectively [19]. Then Equation (2.1) results in the following dependencies:

$$\begin{aligned} S_x &= S \bmod N_x \\ S_y &= (S \operatorname{div} N_x) \mod N_y \\ S_z &= S \operatorname{div} (N_x N_y) \\ D &= \varsigma_S \bmod N \\ D_x &= D \bmod N_x \\ D_y &= (D \operatorname{div} N_x) \mod N_y \\ D_z &= D \operatorname{div} (N_x N_y). \end{aligned} \tag{2.3}$$

If N is not a power of 2, i.e. $N \neq 2^k$, some bit-reversed values ς_S will be greater than N. Therefore we define $D = \varsigma_s \bmod N$. When $N = 2^k$, the modulo operation has no effect.

Bit-Complement Traffic (BCT)

The destination core identifiers in the bit-complement pattern are derived by bit-wise complementing the source core identifier [19]. If $\neg n$ denotes the bit-wise complement operation on a bit string n (e.g. $\neg 01011 = 10100$), then Equation (2.1) gives:

$$\begin{aligned} S_x &= S \bmod N_x \\ S_y &= (S \operatorname{div} N_x) \mod N_y \\ S_z &= S \operatorname{div} (N_x N_y) \\ D &= \neg_S \bmod N \\ D_x &= D \bmod N_x \\ D_y &= (D \operatorname{div} N_x) \mod N_y \\ D_z &= D \operatorname{div} (N_x N_y). \end{aligned} \tag{2.4}$$

Localized Random Traffic (LRT) - the Alpha Model

In SoC design, components with frequent communication are placed close to each other. This avoids unnecessary interconnection delay in circuit level and congestion in architecture level. Local traffic model addresses such practical design issues. In local traffic more packets are generated with destinations close to the source. As the source-destination distance increases, the frequency of packet generation decreases. The probability of packet generation, therefore, is dependent upon the source-destination distance. More packets are generated with short source-destination distance. Thus, the probability function has inverse relationship with the distance.

The level of localization is another factor in the model. For highly localized design, most of the packets generated are with short distance destinations. For loosely defined design where any source communicates with almost all nodes in the network, the localization is low. This variation in the level of localization can be explicitly specified in the model by the locality coefficient, α. When α=0, localization does not exist, and every core generates packets with equal probability to all cores (always excluding self-traffic), whether near or far; this is identical to URT. As α increases, the localization effect increases and the number of packets generated with nearby destinations increases. As $\alpha \to \infty$, the average packet distance approaches 1 hop.

For a given network size of N cores the probability of sending a packet from S to D is

$$P_D = \frac{1}{K_S} . \frac{1}{|S-D|^{\alpha}} \tag{2.5}$$

for $S \neq D$, where $|S-D|$ is the geometric (Manhattan) distance and K_S is a normalizing factor that limits the sum of all probabilities to 1. Its value is different for each source S and is calculated as follows:

$$K_S = \sum_{D=0}^{N-1} \frac{1}{|S-D|^{\alpha}}. \tag{2.6}$$

The localization effect varies according to the network size and topology. Figure 2.8 shows the localization effect on the average distance for a network of 216 routers arranged as 8×8×8 and 2×8×16 cuboid and a 16×16×1 mesh. When α=0, the average hop-count is the same as with URT, though the values are different for each configuration. As α increases, the localization of traffic increases, and the average distance decreases until all curves converge to a value of one hop-count.

2.8.2 Temporal Distribution

A more comprehensive approach is to use spatio-temporal traffic patterns that exhibit bursty characteristics, which is more representative of how real applications communicate over networks. The temporal distribution defines the timing of release of packets into the network. Several studies have concluded that realistic

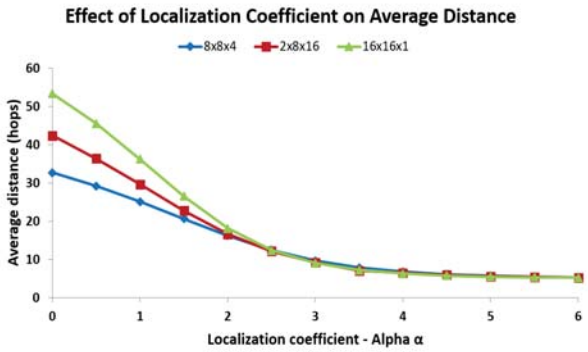


Figure 2.8: Effect of the variation of localization coefficient α on the average distance measured in hop-counts. The hop-count converges to 1 with increasing α

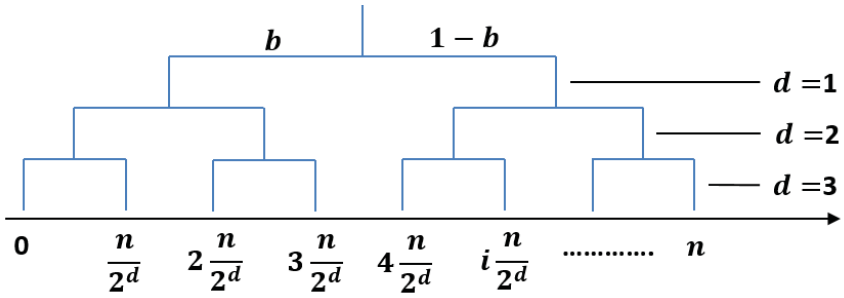


Figure 2.9: Distribution of packets in the B-model

network traffic demonstrate the property of *self-similarity* over a long period of time [5] [71] [70]. As bursty traffic is very prevalent in real applications, we have established a self-similar synthetic pattern as a bursty traffic model which emulates realistic streaming of data.

Discrete self-similar traffic can be modelled by the bursty model (B-Model) as described in [71]. In the B-Model, a bias β ($0 < \beta < 1$) is introduced to the streaming pattern. A bias β=0.5 indicates that packets are streamed at a uniformly distributed rate throughout a time interval comprising, say, n cycles. When the bias is set below or above 0.5, the streaming rate becomes skewed, with the n-cycle time interval being split into two equal portions, and a specified fraction of packets being emitted in the first half, and the rest in the second half.

For example, a bias of β=0.2 implies that 20% of the packets are streamed in the first half and 80% in the second half of the time interval under consideration, or vice versa. This process of halving is continued for each generated half of the original interval, for a number of times that is defined as the depth d, resulting in

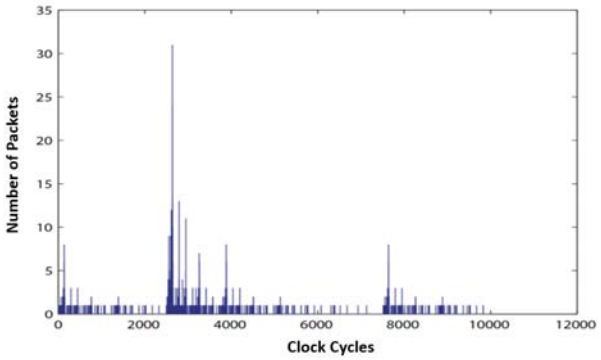


Figure 2.10: Distribution of 1000 packets over 10,000 cycles according to the B-model with β=0.2

some number of discrete time intervals in which the packets are distributed. For an n-cycle time series, shown in Figure 2.9, the number of such discrete intervals in the final sequence is given by $\frac{n}{2^d}$. The maximum value for d is limited by the inequality $\frac{n}{2^d} \geq 1$ or $d \leq \log_2(n)$ (where the simulation cycle duration has been normalized to 1), as a simulation cycle is an indivisible, atomic unit of time. After each division, the choice of which half is assigned 20%, and which 80% (in this example), is made randomly.

The number of packets that a node injects into the network within any period, $x(i\frac{n}{2^d})$, can be expressed as a function of the bias, β, the division depth, d, and the injection rate, γ

$$x(i\frac{n}{2^d}) = (\{\beta, 1-\beta\})^d(\gamma n - \sum_{j=0}^{i-1} x(j\frac{n}{2^d})) \tag{2.7}$$

Equation 2.7 shows that the traffic volume at a given point in the final time sequence is defined as a function of the traffic volume at the coarser time step, and has a straightforward recursive implementation. The derivation of the equation can be found in Paper I [79].

Figure 2.10 shows the distribution of 1,000 packets over 10,000 cycles with a bias of $\beta = 0.2$ and an injection rate of $\gamma = 0.1$. If the total is increased to 2,000 packets ($\gamma = 0.2$), the only change is in the amplitude (y-axis). The temporal distribution (x-axis) is identical.

2.9 Memory Model

A 3D integration technology enables processor-memory stacking in many ways for different applications [77]. Our simulation model includes a complete memory system to address the needs of the different applications. These memory blocks can

be treated as cores that are activated for read or write packet from any processor core. More details with experimental examples can be found in Section 3.4.

2.10 TSV Model

TSVs are short and fast vertical interconnects as compared to the long and thin planar (horizontal) wires. To exploits the unique TSV features, the simulator provides TSV models which can be optimized and reused for different applications allowing high-speed inter-layer communication. For example, double data rate (DDR) configuration of inter-layer communication of the simulation model uses a clock pumping technique to vertically deliver two bits per cycle through a single TSV [83].

TSVs are relatively expensive and complex to prototype. Thus, an accurate TSV model is particularly important for 3D chip design. In each TSV model, a number of TSVs are set as a bundle and the clocking mode is configured. It has sender and receiver components in its internal configuration, physically connected with TSV bundles to the immediate cores in both ends. More details with experimental examples can be found in Section 4.2.

2.11 Network Monitors

Network monitors are used to check runtime activities of a router. The information can be used by the same or other routers to make future routing decision. It can also be extracted and used by designers to analyze the network behaviour of a simulated system and make design decisions. In our model, the network monitoring service can be used to manage network traffic congestion, system faults, thermal variations and other challenges that may arise with the 3D integration.

Basic performance metrics such as throughput, latency and additional information from network hop-count and delay at zero-load can be calculated.

In every cycle, packets arrive at all destinations at a rate based on the injection rate and the congestion level of the network. The throughput per resource per cycle, λ, is defined in Equation (2.8), where P_{total} is the total number of packets received over the simulated range, N is the number of cores in the network and C is the number of cycles in the sampling region.

$$\lambda = \frac{P_{total}}{N \times C} \tag{2.8}$$

The network latency, $T_{network}$, is the time required for a packet to travel the source-destination distance in terms of clock-cycles. The parameter C_{Init} is the initial time when the packet is sent from the source and C_{Final} represent the final time of packet arrival at the destination core. When the network is at zero-load, the raw latency is equivalent to the minimum distance.

$$T_{network} = C_{Final} - C_{Init} \tag{2.9}$$

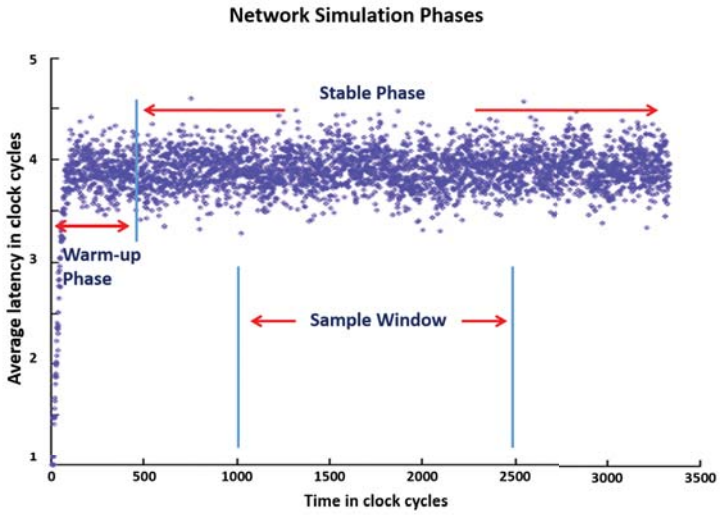


Figure 2.11: Network simulation phases shown for 6x6x6 with 0.9 injection rate under local traffic

The model allows easy integration with basic development tools such as ModelSim and Matlab through which input variables can be set for measurement and results can be evaluated. Metrics of performance and power such as average network latency, processor-memory access latency, network throughput: ejection per core, hot-spot bandwidth, power consumption of individual layers and hot spots, and core energy per bit can be measured. In addition, vertical and horizontal data traffic distribution and local and global cache utilization status can be rapidly extracted and analyzed.

For a range of injection rates within the simulation period, the average latency values are calculated for packets collected from a sample window defined within the stable phase of the network.

2.12 Network Simulation Phases

When the simulation is run, initially the network is empty. A packet injected from a source core to the network is ejected at the destination core only after certain network time lapses. The first packets to be ejected are these with short source-destination distance. Slowly, more long distance packets start to get ejected. Thus, the aggregate latency builds up over-time as shown in the simulation of 6×6×6 network with high injection rate in Figure 2.12. The simulation consists of two phases: a *warm-up phase* followed by a *stable phase*. In the warm-up phase, the

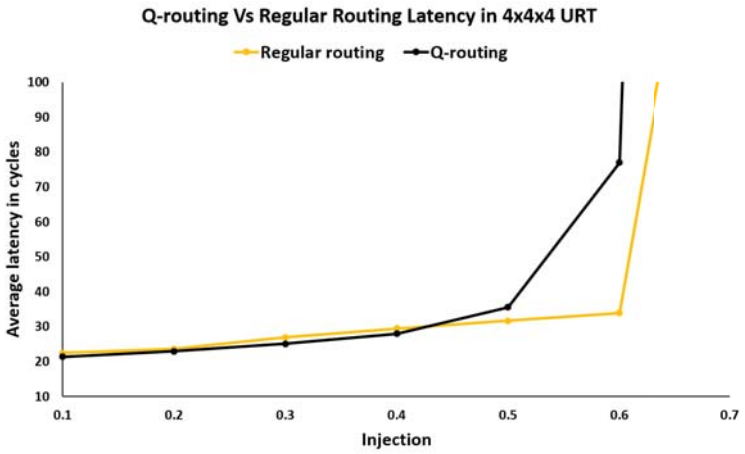


Figure 2.12: Comparison between Q-learning and regular routing in 4x4x4 under URT

average latency initially starts from 1 cycle and slowly grows to 3.5 cycles. After 500 cycles, the simulation enters a stable phase with latency fluctuating between 3.5 - 4.5 cycles giving an aggregate of 4 cycles. In order to extract the performance of the simulated network, a sample window must be defined in the time period of the stable phase. From the sample window, relevant data can be extracted safely while ensuring the accuracy of resultant information.

2.13 Simulation of Irregular Networks

The Q-routing for irregular networks is compared against the regular routing as a baseline by simulating a regular 4×4×4 network under URT. All the simulation steps described in the previous sections are used to extract the average latency as shown in Figure 2.11. The Q-routing exhibited comparable performance when the network work-load is stable under low injection rates. However, when the injection rate is increased, the Q-routing saturates earlier than the regular routing. This is because when the traffic congestion increases, more packets are misrouted. This leads for the Q-learning mechanism to update the look-up-tables with values worse than earlier. The advantage of of Q-routing comes when the network is irregular since the regular routing implementation can not be used in irregular networks. Additional simulation results and analysis can be found in Paper VIII [82].

2.14 Summary

In this chapter we introduced a simulator with several features. A router micro-architecture for deflection routing, various spatio-temporal traffic generators, TSV models, memory models, and network monitors are implemented. We demonstrated the capability of our simulation model to analyze the performance of regular and irregular network topologies under the spatio-temporal traffic patterns. By using the simulation model, selected network topologies can be configured to meet the performance requirements early in the design flow. The test results show the potential of simulator to handle irregular network configuration using the proposed Q-routing algorithm for 3D NoCs.

Chapter 3

Architectural Scalability of NoCs

Architectural scalability refers to the changes that occur at the architecture level. Processor architecture evolved from single core with sequential execution to multi-core and many-core architecture that support parallelism. In scaling many-core architecture, the NoCs infrastructure becomes a key component to integrate the cores. Specially when the many-core architecture is implemented in 3D, the whole design scheme becomes integration-centric. With NoCs at the center, new ways of processor-memory communication can be defined. In fact, experimental and commercial processors such as Godson-T with 64-cores [27], Intel Teraflop with 80-cores [69] and Tilera with 64-core processor [8] show that with scalability, future many-core architecture as a SoC solution will have 1000s of cores.

Networks-on-Chip consists of resources: routers and links. Scaling NoCs in many-core architecture essentially means proportional increase in the number and capacity of routers and links. Here, the following scientific questions can be raised. How does the performance scale when the number of routers grows in a 2D mesh, or in a 3D cube (with router connectivity between layers)? How would the processor-memory communication be optimized with each scaling? What are the key trade-offs with regard the performance metrics in terms of latency and throughput?

In this chapter, we address these and other related key challenges. We explore the scalability of 2D mesh and 3D cube NoCs architecture in an effort to develop design guidelines for future development of many-core architecture. We propose new processor-memory model configurations that use NoCs to communication, and make comparative analysis in order to study the performance under various traffic scenarios.

Most of the models and results presented in this chapter are already published in Paper V [77] and Paper VI [81].

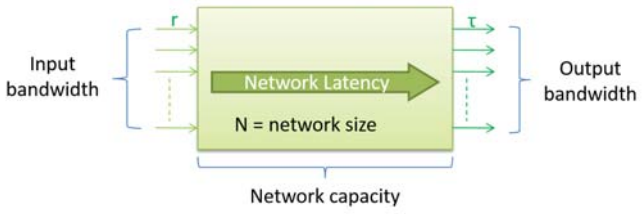


Figure 3.1: Network capacity is dependent on the resources inside the network

3.1 Related Works

In literature, we find several studies related to NoCs in many-core architectures. Park et al [59], explored a multi-layer NoC router architecture for a number of traffic patterns. In the paper, the number of cores is fixed at 36 and the number of layers in 3D cases is kept constant at four. Grot et al in [36], proposed an energy and area efficient network architecture, after concluding that the quality of service in traditional router-centric networks degrades in 1000 core processors. A well known work by Dally et al describes the performance of communication networks of varying dimension for wormhole routing is [17], which generalizes the interconnection network as being a k-ary n-cube torus, with n being the dimension of the cube, and k being the radix, or number of routers in a given dimension. However, in practice torus topologies are rarely used as compared to 2D mesh and 3D cube topologies.

With the advent of 3D technology, memory unit is no more an off-chip component that are accessed through hierarchial form. It can be stacked as a memory-layer and communicate with processor layers through TSVs. The application of 3D technology enables the stacking of memory layer on top of processor layers creating a massive many-core architecture. Early 3D memory packaging techniques and benefits that uses system-in-package (SiP) with stack of memory chip layers are discussed in [67]. The paper [64] shows that die to die Multi-chip-Module stacking is a reliable solution but with cost limitation. Gabriel presented in [50] that by using 3D DRAM on CPU, a significant increase in memory system performance can be achieved. The approach is novel but assumes that, there is a CPU at the bottom layer connected with a bus to the DRAM layers on the top. In multiprocessor environment, the bus performance becomes a bottleneck. Hence, we find it imperative to examine the performance of such system if NoCs is used in order to relieve the bottleneck. We attempt to find out the optimum configuration of processor-memory stacks that implement NoCs as a communication backbone.

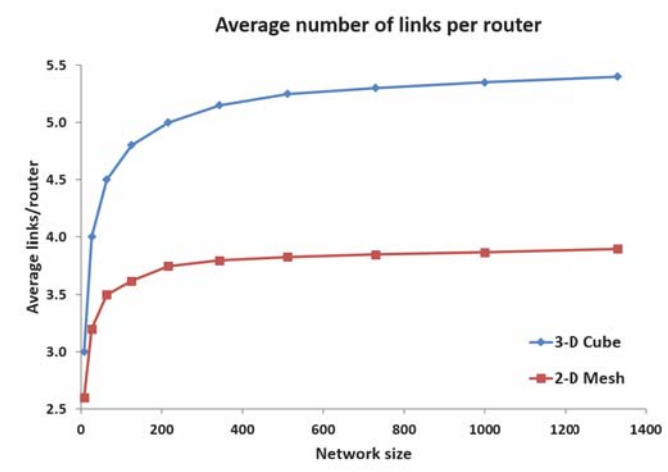


Figure 3.2: Number of links per router for 2D mesh and 3D cube networks

3.2 Modelling of Network Links and Loads

Let's consider a generic NoC with a single-cycle router micro-architecture. Each router has a port connecting to a local core which is used to inject and eject packets. Figure 3.1, shows the model for network latency. Packets are injected from the input side with input bandwidth (injection rate r), and ejected at the output ports with output bandwidth (throughput τ). The input and output ports to the local cores are of the same size, and equal to the number of routers available in the network. Once a packet is injected from a source, it flows inside the network until it is ejected at the destination after certain time delay. This delay inside the network is the *network latency* which is mainly caused by the source-destination physical distance, and also by deflections due to the traffic congestion inside the network.

In the absence of traffic congestion, the network latency is purely the time it takes for the packet to travel the minimal path in the source-destination distance (i.e. Zero-load delay). If there is traffic congestion, the packet competes for output links based on a defined priority scheme, and may get deflected from its minimal path. The solution for this is to increase the network capacity with more links and other resources for the packets to flow. Network capacity is the ability of a network to accommodate a number of packets at any given time. This is in effect the network bandwidth. Networks with higher bandwidth facilitate easy flow of packets as many as injected (input bandwidth), and hence reduce the network delay.

The network capacity or network bandwidth is dependent on the number of links available in the network. Each link is a *channel* to route packets. There are 4 directional links available in each 2D router has, whereas a 3D router has 6 directional links. However, depending on the topology, dimension, and size of the

network, not all of the available links can be used for routing packets, and hence the total number of links varies. For example, in 2D mesh, only half of the links in the corner routers of the network are connected. The other half are unused. As we have shown in Paper VI [81], the expression for the total number of links in 2D mesh and 3D cube networks is given as follows.
For a 2D $k_x \times k_y$ networks:

$$L_{2D} = 4k_xk_y - 2(k_x + k_y) \tag{3.1}$$

For a 3D $k_x \times k_y \times k_z$ networks:

$$L_{3D} = 6k_xk_yk_z - 2k_xk_y - 2k_xk_z - 2k_yk_z \tag{3.2}$$

which quantifies the differences in link bandwidth in different topologies.

In order to simplify our analysis, let's assume a network of equal radices, i.e., $k_x{=}k_y{=}k_z{=}n$.

The plot in Figure 3.2 depicts the number of links, as a function of network size to highlight the differences between a 2D mesh and 3D cube networks as defined in Equations (3.1), and (3.2), respectively.

The increasing number of links per router in both networks allows for higher network throughput as the network size increases due to the increased bandwidth available to route a packet in the network. The 3D cube network has an advantage over the 2D mesh due to the increased number of links per router. This trend is evident in the simulation results shown in the next Section 3.3.

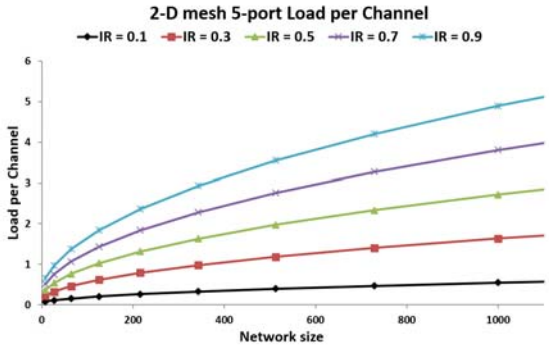
If the *network capacity* is a measure for the number of links available in the network, the *network load* γ is a measure for the number of packets flowing in the network at a given cycle. It is dependent on the network size and on the injection rate. For N size network, packets injected under uniform random traffic (URT) with a rate r per router travel at least the average of the source-destination distance ($\overline{\mathbb{H}}$) before ejection. Hence the number of packets γ available inside the network is given as follows.

$$\gamma = r \times N \times \overline{\mathbb{H}} \tag{3.3}$$

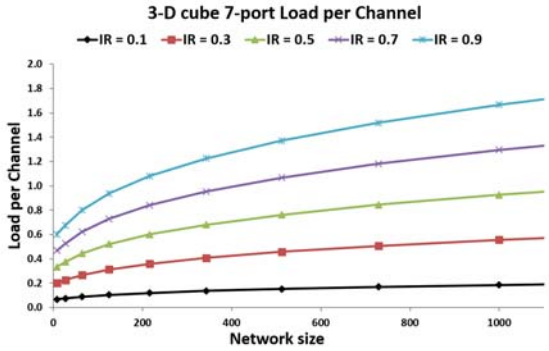
Let's consider Figure 3.1 as a model. As the network grows, the network capacity grows and hence the network can handle more load. But, at the same time the input and output bandwidth also grows.

Now if we take the load under URT for a 3D cube $N{=}n{\times}n{\times}n$ network as an example, the average distance $\overline{\mathbb{H}}_{urt}{=}n$ [17]. We see that the load γ in Equation (3.3) grows faster than the network capacity in Equation (3.2). Consequently, the injection rate r has to decrease as the network grows to keep the network stable.

The load per channel, γ_{Chnl}, is the amount of packets contained in each channel. For the above example, γ_{Chnl} equivalent to all packets distributed over all links defined as:



(a) Channel load in 2D mesh network



(b) Channel load in 3D cube network

Figure 3.3: Load per channel under uniform random traffic in 2D mesh and 3D cube networks

$$\gamma_{Chnl} = \frac{r \times n^2}{6 \times (n-1)} \tag{3.4}$$

The load per channel for 2D mesh and 3D-cube for varying injection rate is shown in Figure 3.3a and Figure 3.3b respectively.

3.3 Scaling in 2D mesh and 3D cube topologies

A channel can hold maximum of single packet at any given cycle. Thus, as a guideline, a value of one in the load per channel γ_{Chnl} is the upper limit for the saturation point. In order to maintain network stability, injection rates r are lowered and the load per channel, γ_{Chnl}, is kept below the saturation point. When stable, the injec-

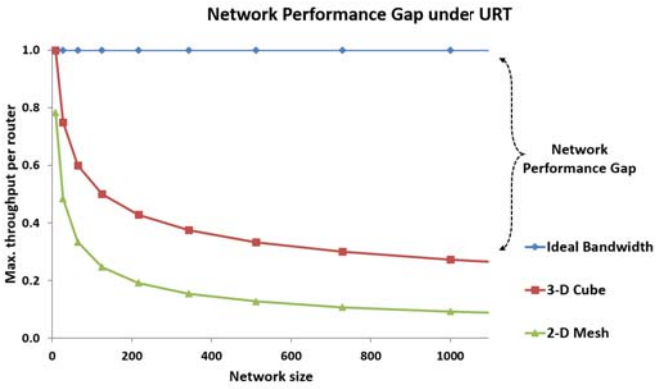


Figure 3.4: The network performance in URT degrades as the number of cores sharing the network increases

tion rate equals the throughput $r{=}\tau$, and in an ideal case, maximum throughput $\tau{=}1$

To show the effect of network load when network size grows, we carried out performance simulation for in 2D mesh and 3D cube networks under URT. The results in scalability show that network performance in many-core degrades as the number of cores increases into hundreds. As shown in Figure 3.4, there is a growing network performance gap due to the unmatched network capacity with that of network load. The simulation results presented for 2D mesh and 3D cube indicate that the 3D cube networks perform better than 2D mesh. This can be attributed to the design of 3D router micro-architecture which has the higher link per router ratio as shown in Figure 3.2 and better channel load as shown in Figure 3.3.

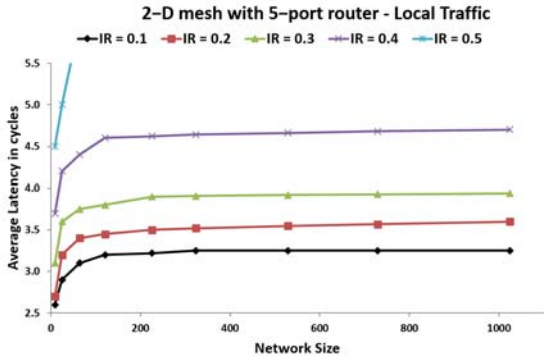
Further experiments are carried out to derive performance figures related to latency and throughput for 2D mesh and 3D cube networks when tested under local traffic pattern.

We see that under local traffic shown for 2D mesh in Figure 3.5 and 3D cube in Figure 3.6 that a 2D NoC with localized architecture can be scaled to a very large dimension with no significant effect on throughput or latency. This is because packets in local traffic rarely travel long distances across the network. Hence the effect of network scaling at global level does not affect the channel loading at local level.

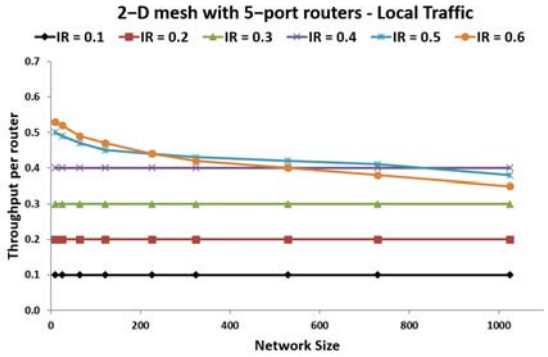
More details and additional experiments can be found in Paper VI [81].

3.4 3D Processor-Memory Models

In traditional processor-memory configurations, the processor is a separate chip accessing off-chip memory units through system bus extended on PCBs. In this



(a) Latency



(b) Throughput

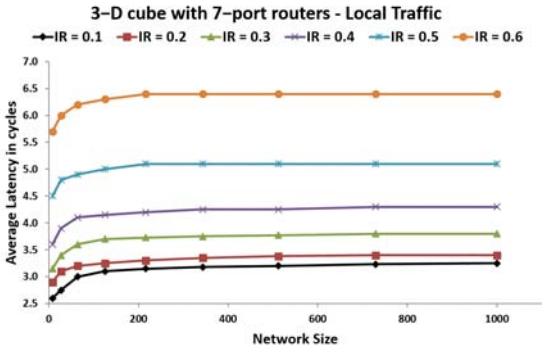
Figure 3.5: 2D Mesh Performance under local traffic

section, we examine 3D processor-memory stacks when NoCs is used as a medium of communication between processors and shared memories. Here, we assume a heterogeneous many-core architecture where both the processor and the memory are treated as local cores, each with their own routers in the network.

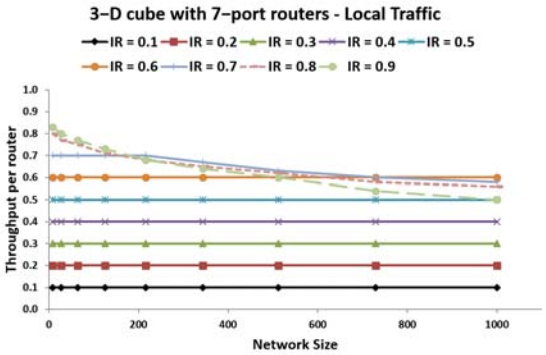
First, we propose new 3D processor-memory configurations as shown in Figure 3.7. Each configuration consists of an array of processor layers and memory layers.

a. *Dance-hall architecture* - The processor layers are placed on one side and all the memory layers are place on the opposite side. This is similar to the multi-chip memory stacking on top of processor die. However, in place of bus, NoCs is used.

b. *Sandwich architecture* - The processors layers are placed between memory layers.



(a) Latency



(b) Throughput

Figure 3.6: 3D Cube Performance under local traffic

c. *Per-layer architecture* - Each layer is a memory layer mixed with processor blocks. We assume the process technology for such mixed processor and memory configuration per layer is available.

d. *Terminal architecture* - The processor layers are on both ends (top and bottom) with memory layers in between.

e. *Mixed architecture* - One processor layer is at the bottom end and another processor layer placed in the middle between memory layers. ...

3.5 Evaluation of Processor-Memory Models

Experiments are conducted to find out the optimum configuration that efficiently utilizes deflection routing NoCs as a communication backbone. Let's consider a

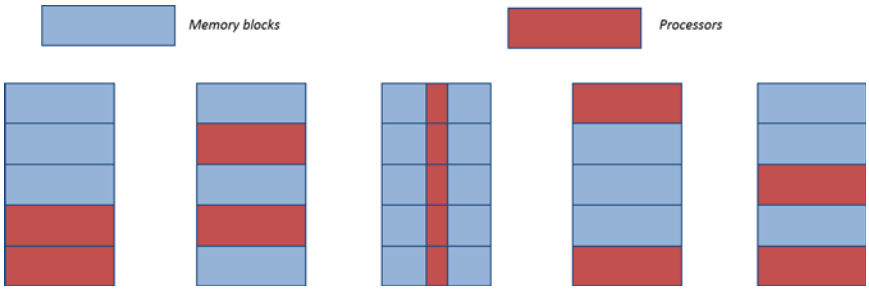


Figure 3.7: 3D processor-memory configurations (a) Dance-hall (b) Sandwich (c) Per-layer (d) Terminal (e) Mixed

massive processor memory integration representing a many-core architecture, we used a 16 layers model. Each layer is an array of 4×4 cores. There are two types of cores; a processor and a memory core, each connected to a NoC router. For example Figure 3.8 shows the Dance-hall architecture modeled for simulation. The bottom two layers are processor layers and the top 14 layers are memory layers. A request packet is sent from a processor core to any memory core and in return the memory core responds with an acknowledgement. Thus the performance is measured based on a a round-trip journey a packet makes. Additional details regarding the experimental setup can be found in Paper V [77].

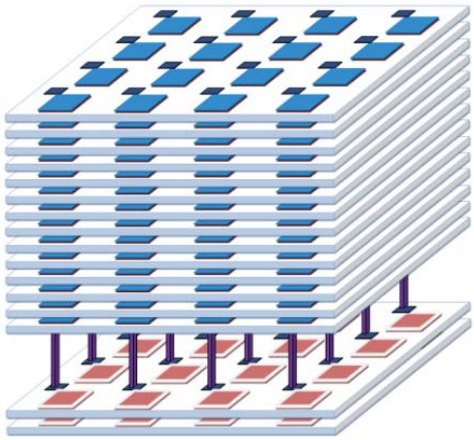
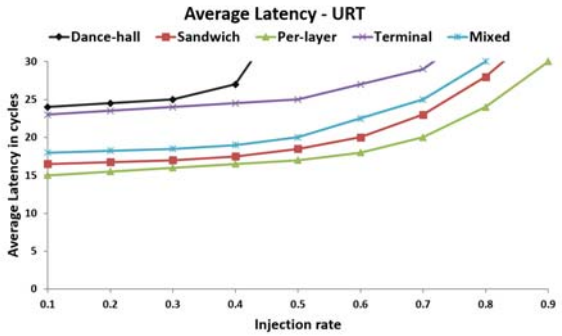


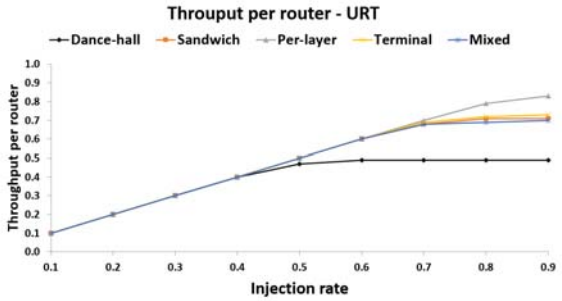
Figure 3.8: Dance-hall architecture

The plots in Figure 3.9 show the performance of each configuration when run under URT. Figure 3.9a is the latency for varying injection rate. Generally, the latency for each configuration increases with increasing injection rate. This trend is expected because of the increasing level of traffic congestion. When injection

rate is above 0.4, the Dance-hall architecture saturates. This shows the network has reached its limit earlier than others in accepting new packets. Because of the relative positioning of the processor layers, it takes longer time for a packet to reach memory layers placed on the opposite end. On top of that, the traffic on the side of the memory starts to grow because more packets are competing for link resources during the round-trip journey. In comparison, the Per-layer architecture has lower latency because on average its processor layers are physically placed in close proximity to any of the memory layers. Packets can make round-trip journey with less congested paths. Thus, the network is stable for all injection rates. The other architectures saturate in between.



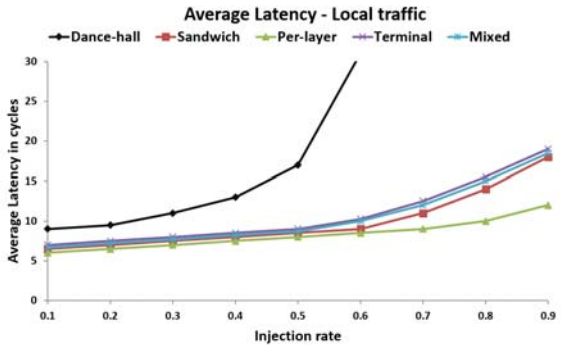
(a) Actual Latency



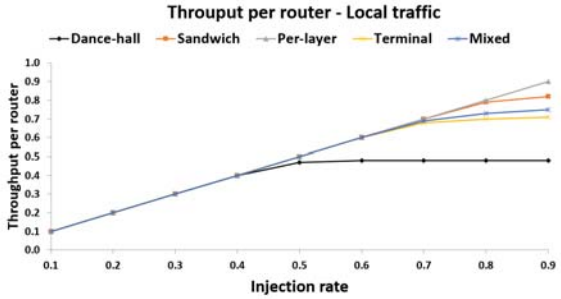
(b) Throughput

Figure 3.9: Performance under Uniform Random Traffic Pattern

Figure 3.9b is the throughput per cycle. Again, the Dance-hall architecture throughput levels-off just above injection rate of 0.4. This shows that the architecture cannot handle any network traffic when the injection rate is more than 0.4. In comparison, the Per-layer architecture has an advantage of balancing the distribution of load among the layers freeing links for more packets to be injected.



(a) Latency



(b) Throughput

Figure 3.10: Performance under Local Traffic Pattern

Figure 3.10 show results under local traffic pattern to see the effect of locality as explained in Section 2.8.1. When the packets are localized, each processor frequently communicates with nearby memory cores. Two distantly placed processor layers such as in the Terminal architecture have fewer memory cores to share compared to closely placed layers with overlapping boundaries as in the case of Dance-hall architecture.

Figure 3.10a is the latency for varying injection rate. Compared to the URT, the latency under local traffic decreases for all architectures. But, the Dance-hall is still the architecture that saturates earlier than others. The one-sided position of its processor layers make the processors compete to locally communicate with the memory layers close to the processor. This leads to the concentration of traffic in the boundary between the processor and memory layers. In comparison, the processor cores in Per-layer architecture are surrounded by memory cores. This gives the architecture a more localized environment leading to lower latency and late saturation.

Figure 3.10b is the throughput per injection rate. In case of Dance-hall architecture, the throughput is equal to the injection rate up to 0.5. However, beyond that the network becomes unstable making the throughput to level off around 0.5. The injected packets are not able to enter the network. In case of Per-layer architecture, the throughput increases almost steadily for the whole range. The injection rate is equivalent to the ejection rate.

The performance evaluation of the proposed processor-memory configurations show that the 3D many-core architecture opens new design schemes in defining memory-hierarchy with NoC as communication backbone.

3.6 Summary

In this chapter, we explored the scalability of the NoC communication architecture for 3D systems under the physical constraints imposed by processing and stacking technology and provided models and guidelines for network design of future many-core architectures.

We have shown that with scalability, the performance of NoCs yields a diminishing return leading to the network performance gap. The gap is one of the major challenges in many-core parallel architecture that is unmatched growth of the network capacity and network load.

We also evaluated NoC performance when used as a medium for processor-memory communication. We proposed several 3D processor-memory configurations and compared each configurations' performance with two traffic flows, uniform random and local. Based on the saturation point for each case, we highlight the optimal configurations that could be used by designers of NoC based massively integrated processor-memory architecture.

Chapter 4

Exploiting Fast Vertical Interconnects

In previous chapters we have seen the benefits and limitations of architectural scalability. We examined the performance of NoCs when networks grow in size, or when networks are topologically reconfigured as 2D mesh or 3D cube. In this chapter, we study how the communication architecture in many-core systems can be designed in such a way to make optimal use of the underlying physical properties of the TSVs as horizontal and vertical interconnects.

TSVs are short and fat interconnects as compared to the long and thin planar (horizontal) wires. This feature allows TSV wires bundled together to be signaled in tens of GHz frequency range with insignificant interconnect delay enabling the possibility of ultra-low-power, high-speed inter-layer communication [33]. In contrast to the off-chip communication between ICs on board, the TSVs indeed bring disruptive features.

The number of TSVs in a bundle varies depending on the application and process technology. But, adding more TSV in a bundle has its limitations; it introduces thermo-mechanical instability, increases TSV defect, consumes large area, and it increases process complexity and cost [74] [76].

To address these limitations and to maximize the potential benefits afforded by fast TSV, we explore the performance of NoCs when the vertical link is clocked at a multiple of the horizontal clock. Having a faster vertical clock maximizes the benefit of the faster TSVs, and provides increased bandwidth for inter-layer data communication, especially beneficial in the presence of heavy vertical traffic as would occur with memory being stacked above processors as discussed in the previous chapter.

Our study provides cycle-accurate simulations for various traffic patterns, and quantifies the performance improvement in 3D NoCs when the data rate across the vertical links is clocked twice the data rate of the horizontal links (DDR NoC) as opposed to a homogeneous data rate over horizontal and vertical links (SDR NoC).

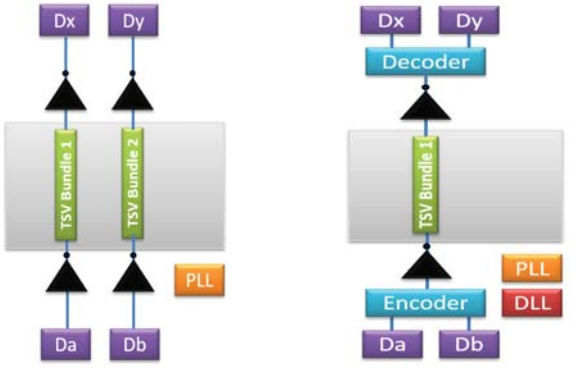
The study considers the scalability of these schemes as the network size grows, and also considers asymmetric networks, where the radices in the different dimensions are unequal, as would happen in a tall die stack with multiple memories.

The main contribution of the work is a novel exploration and modelling of a 3D aware NoC design under realistic physical constraints, utilizing the fast electrical properties of TSVs in 3D many-core architecture. The details of how the use of the proposed models can further reduce TSV manufacturing cost while increasing the over-all performance systems are published in Paper III [83], [35] and Paper VII [78].

4.1 Related Work

TSV, compared to the planar interconnects, has its own unique properties. Whereas the horizontal interconnects are long and thin, TSVs are short and fat wires. Global planar interconnects delay is long and requires several stage repeaters in between to maintain signal integrity with increasing speed. When compared to the horizontal link delay, the higher TSV signalling speed allows enough room for a vertical link to be clocked faster than the horizontal links [73] [33]. Hence 3D ICs differ from 2D ICs in that the vertical links have a higher bandwidth than the horizontal links. In spite of the nascent nature of the field, quite a few works in the literature address issues around the design of the communication architecture for 3D many-core systems. The different NoC-based system architectures for 3D systems have been exhaustively enumerated in [62], depending on whether a die housed in a tile has one layer or several layers, and whether the NoC itself is 2D or 3D.

The authors of [48] proposed a dynamic time-division-multiple-access (dTDMA) with a centralized arbiter for the vertical communication link, which allows single hop latency for packets between any number of vertical layers to take advantage of the electrical behavior of the relatively short and wide TSV which can support a much higher signalling speed. The work in [28] describes a detailed study using various traffic patterns under a realistic protocol for both architectures comprising a 7-port with router connectivity to the horizontal and two adjacent vertical routers and a 6-port router where instead of 2 vertical ports, a single port provides connectivity to a bus interface that allows access to any router in the vertical dimension. They also study other mesh topologies. However the simulations are only carried out for a fixed network size. In [81], we have shown the scalability of these different schemes by quantifying the performance in terms of throughput and latency for various network sizes, and in particular shown that the solution with a bus for the vertical connectivity does not scale as well as the 3D router under uniform random and local traffic patterns, due to the inherently lower bandwidth in the network when the number of vertical links are reduced. Finally [56] describes a working 3 layer 27 core prototype that provides proof of concept of the 3D NoC, but identifies the need for system-level explorations of the kind discussed here.



(a) SDR model using a pair of TSV bundles

(b) DDR model using single TSV bundle

Figure 4.1: Proposed DDR model doubles the inter-layer communication

4.2 Modelling of Multi-rate Through-Silicon-Vias

First we explore the use of double-data-rate (DDR) model to double the clocking of TSVs. The standard model commonly applied in NoCs uses a single-data-rate (SDR) model as shown in Figure 4.1a in which a single clock input is shared by the horizontal and vertical links. Data is sent from D_a to D_x and from D_b to D_y, each through separate TSV bundles forming a single data rate (SDR).

In Figure 4.1b, a DDR clock pumping technique is used to send data. An encoder and decoder blocks comprise the DDR circuit. A common TSV bundle is shared by both senders to propagate data D_a and D_b within one cycle - one at the rising edge and the other at the falling edge.

Phase-locked loop (PLL) and Delay-locked-loop (DLL) blocks are used to generate clock inputs [15]. The PLL is used to synthesize frequencies for main clock inputs, which is common to both SDR and DDR, whereas the DLL is used to delay by phase shifting the main clock input 180° degree in order to capture the second input Db in the falling edge. Each logic is self-timed to match the DDR conversion in the next clock domain. Generic models of digital DLL and PLL are used to make comparative analysis between the SDR and DDR implementation of TSV models. As this study is intended to explore the communication architecture, the DLL and PLL resources are not considered in depth. Further analysis of the DDR circuitry can be found in Paper III [83] and in [35].

4.3 Topological Properties

The DDR TSV models are functional only when connected to a router micro-architecture that supports multi-rate communication. The standard router micro-architecture basically uses one clock input for both horizontal and vertical links and it communicates in single-data-rate (SDR). For the DDR TSV model, a DDR router micro-architecture that communicates in double-data-rate is required.

In order to clearly identify the contrast between a SDR router and a DDR router, relevant topological properties are derived. A 3D cube is a k-ary n-cube with the dimension $n(=3)$ and k being the radix of the router, or the number of routers in a given dimension. We also consider 3D cuboid, where the number of routers in a given dimension are not equal. For a cuboid with the number of routers in the x, y and z dimensions being k_x, k_y and k_z respectively, the number of horizontal and vertical links can be found by splitting the total number of links given in Equation 3.2 into Equations 4.1 and 4.2 respectively. The total link bandwidth is shown in Equation 4.3, where s is the number of pipeline stages (if the router is not single cycle) in the horizontal links and m is the vertical to horizontal clock ratio.

$$L_H = 2[2k_xk_y - (k_x + k_y)]k_z \tag{4.1}$$

$$L_V = 2k_xk_y(k_z - 1) \tag{4.2}$$

$$BW = sL_H + mL_V \tag{4.3}$$

In paper IV [34], we have formulated that the average distance, $\overline{\mathbb{H}}$ between any core to all other cores in an $k_x \times k_y \times k_z$ network can be calculated from

$$\overline{\mathbb{H}} = \frac{1}{3}\left(k_x - \frac{1}{k_x}\right) + \left(k_y - \frac{1}{k_y}\right) + \left(k_z - \frac{1}{k_z}\right) \tag{4.4}$$

which evaluates the average hop-count in any given network for injection rates close to zero-load. In a stable network, the throughput should track the injection rate. Having a higher vertical clock may allow the network to remain stable for higher injection rates than may be possible with equal clocks. Each packet loads the network by occupying one link every cycle. Hence the load that each core puts on the network is a function of the injection rate, as well as the amount of time that the packet spends in the network. If the injection rate is r, and the amount of time a packet spends in the network on average is $\overline{\mathbb{H}}$, the loading of the network by one core γ_{core} is:

$$\gamma_{core} = r \times \overline{\mathbb{H}} \tag{4.5}$$

Equation 4.5 is basically equivalent to when Equation 3.3 is distributed to the number of cores available in the networks $\gamma_{core}=\frac{\gamma}{N}$.

A single hop by a packet is equivalent to the usage of one link. Thus, the average hop-count, H_C, is equal to the number of links used by the packets per cycle. The total Link Utilization (θ_t) is the ratio of the links in use to the total available links in the network.

$$\theta_t = \frac{H_C}{L_t} \tag{4.6}$$

In essence, the link utilization θ_t in Equation 4.6 is equivalent to the channel load γ_{Chnl} described in Equation 3.4.

4.4 Router Micro-Architecture

In this work, two router designs have been investigated to provide the added functionality and capability of faster vertical routing. An SDR router is essentially shown in Figure 2.7 with four horizontal ports connecting to routers on the same layer and two additional vertical ports allowing communication to neighboring layers above and below. All routing decisions in this design are made in a single clock, and each port is treated identically.

In order to exploit the benefits of faster vertical network links, the SDR router is redesigned as DDR router to support faster vertical packet handling as shown in Figure 4.2. Physically, the number of vertical ports and TSV bundles in an SDR router are doubled to make a DDR router. It has four horizontal ports connecting to routers on the same layer and four additional vertical ports allowing communication to neighboring layers above and below. All routing decisions in this design are made on a single clock and each port is treated identically.

In general, beyond DDR, any multi-rate router implementation is a modification of the routing strategy and structure of the SDR router with m additional input and output ports per vertical link, where m is the vertical to horizontal link clock ratio. The control logic allows for up to m packets to be routed to a vertical link per cycle, where previously only one packet per link would be allowed in a SDR router network. The vertical link is clocked m times faster than the horizontal link and thus is capable of routing all packets stored at the vertical output port of a router to the next adjacent layer within one horizontal cycle. The limitations of this implementation become obvious when m becomes larger, as the number of ports and control circuitry increases.

For this study, let's consider a fixed clock ratio at $m = 2$ (DDR). For this case, two packets per vertical port can be routed to an adjacent 3D layer within one cycle of the network.

4.5 Evaluation of Symmetrical Networks

In order to examine the effect of scaling the number of cores in the network on the performance, network sizes of 2×2×2, 4×4×4, and 8×8×8 were simulated with

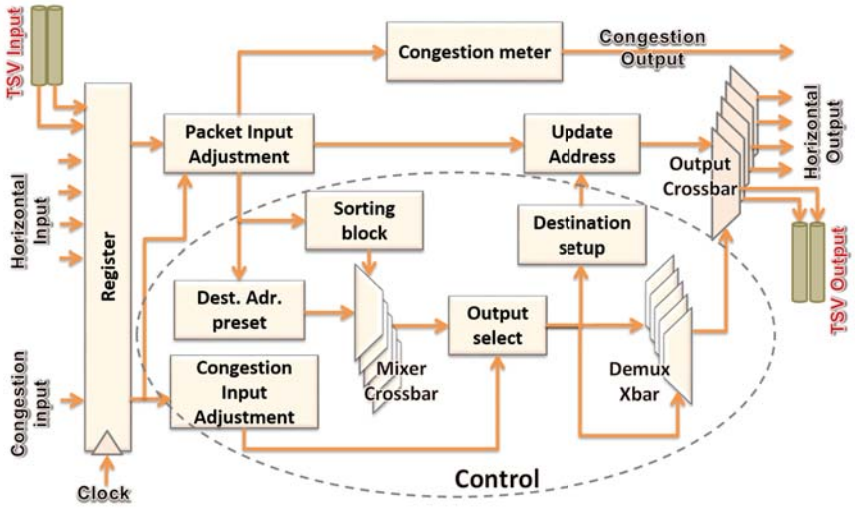
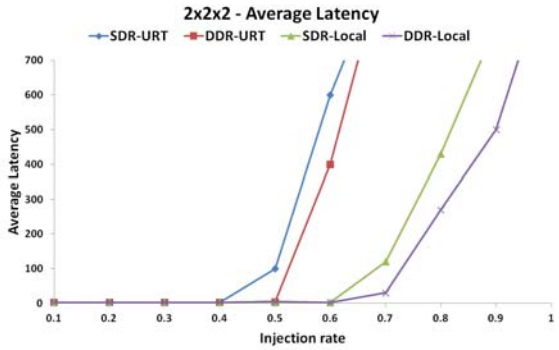


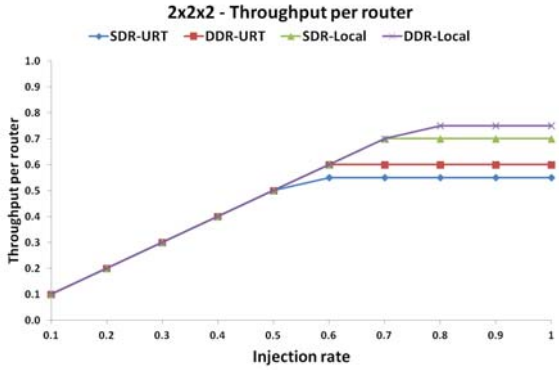
Figure 4.2: DDR Router micro-architecture with double vertical ports to TSV extension for inter-layer communication

injection rates from 0.1 to 0.9 in 0.1 increments. Figures 4.3a, 4.3b and 4.3c show the performance of a 2×2×2, for both SDR and DDR implementations under URT and local traffic patterns. Figure 4.3b shows that the throughput for SDR URT is close to that of DDR URT and that the local traffic has improved the performance for both SDR and DDR networks. The effect of the faster vertical links is not significant in a small network due to the lower number of vertical links. The latency shown in Figure 4.3a increases only in heavy congestion under higher injection rates. This is confirmed in Figure 4.3c by the link utilization for increasing injection rate. It is intuitive that with only two layers the performance benefits of the DDR implementation is likely not worth the overhead of the router and design complexity.

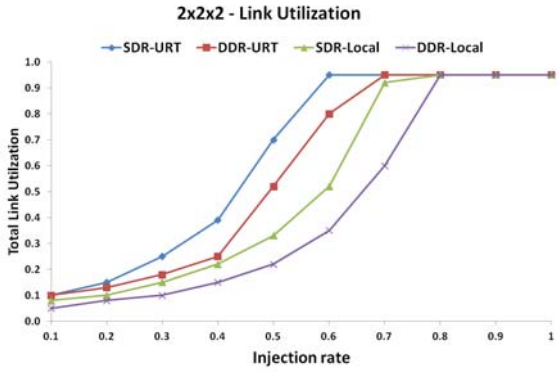
The impact of the DDR scheme becomes apparent in a 4×4×4 network shown in Figures 4.4a, 4.4b and 4.4c. It is clear that the DDR router has improved the throughput capability as shown in Figure 4.4b. The additional bandwidth on the vertical links is more prevalent as the layers increase, which help relieving the congestion of the network under increased traffic. In both the DDR URT and local case, the stability of the network is increased by 0.1 packets per router per cycle ejected from the network as compared to the SDR case. This is also reflected in the performance of the latency shown in Figure 4.4a. In both cases the SDR latencies beyond injection rates of 0.1 begin to increase beyond the DDR network. In Figure 4.4c, the link utilization for URT is higher than the utilization of the local traffic pattern because there are inherently more packets in the network utilizing the available links.



(a) Latency

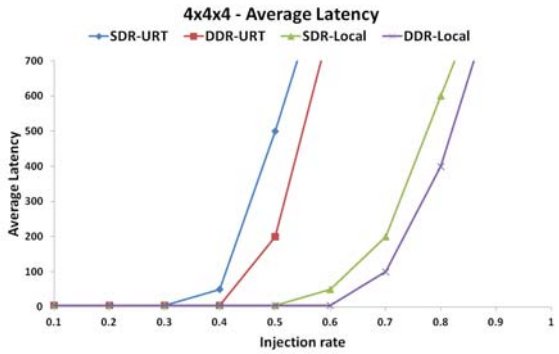


(b) Throughput

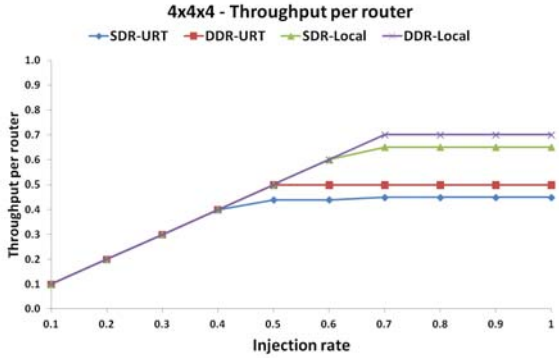


(c) Link utilization

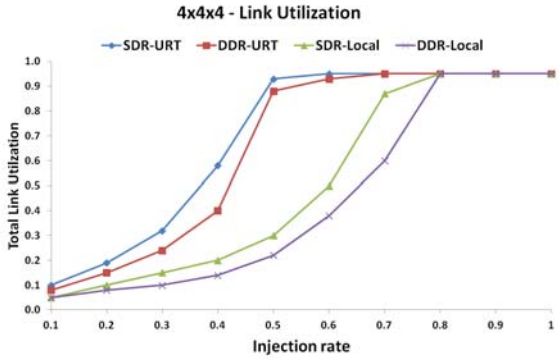
Figure 4.3: Performance of 2x2x2 network



(a) Latency

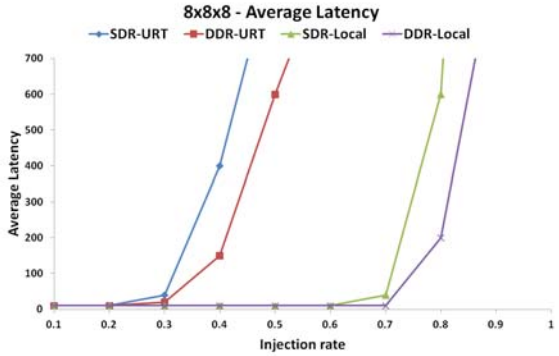


(b) Throughput

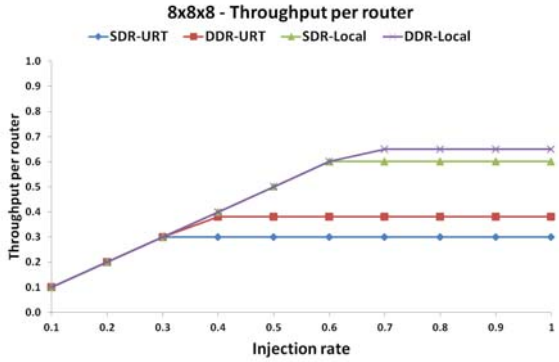


(c) Link utilization

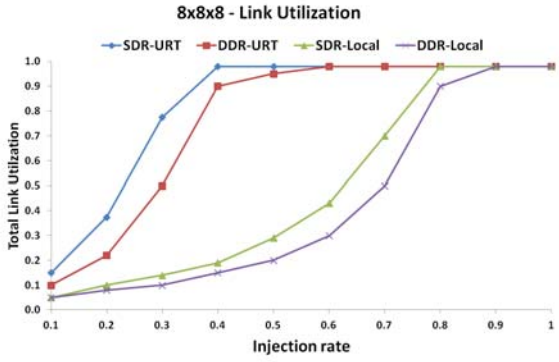
Figure 4.4: Performance of 4x4x4 network



(a) Latency



(b) Throughput



(c) Link utilization

Figure 4.5: Performance of 8x8x8 network

Figures 4.5a, 4.5b and 4.5c plot the performance metrics of an 8×8×8 network for SDR and DDR under URT and local traffic patterns. Here the throughput gain is higher compared to network sizes of 2×2×2, and 4×4×4. The increased number of vertical layers is aided by the higher bandwidth on the faster clocked vertical links. Nevertheless the overall individual throughput is lower and the individual latency is higher for both SDR and DDR modes 8×8×8 as compared to 2×2×2 and 4×4×4 network sizes because of the increasing network load when the number of cores in the network increases.

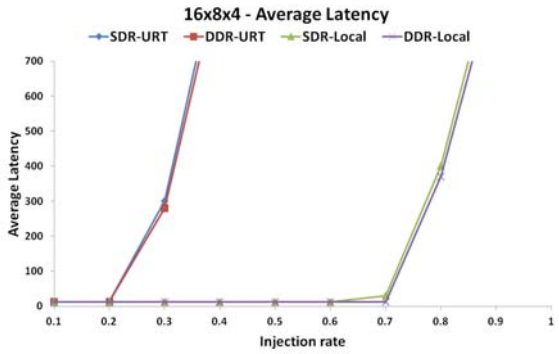
4.6 Evaluation of Asymmetrical Networks

The performance contrasts between SDR and DDR router implementations in asymmetrical networks were also simulated. The number of cores was fixed at 512 to match the 8×8×8 cube but the z and x dimension were altered between two sizes. In this case a bottom heavy 16×8×4 and a top heavy and 4×8×16 are investigated. Figure 4.6a, 4.6b and 4.6c show the performance of 16×8×4, horizontally heavy asymmetric cuboid for SDR and DDR under URT and local traffic patterns. This network only has four layers and the bulk of the cores are laid horizontally. This means there are fewer higher speed vertical links. Figure 4.6b plots the throughput for both SDR and DDR routers, where there is no significant difference in throughput for either case. The URT for SDR and DDR are nearly identical and the locality has improved the performance for both SDR and DDR networks. The latency shown in Figure 4.6a increases only when the network becomes congested with higher injection rates, which is confirmed in Figure 4.6c by the link utilization for increasing injection rates. This network configuration was simulated to demonstrate that in certain network configurations, the added bandwidth of the vertical link does not significantly affect the performance of the network due to the topological properties, in this case fewer vertical links.

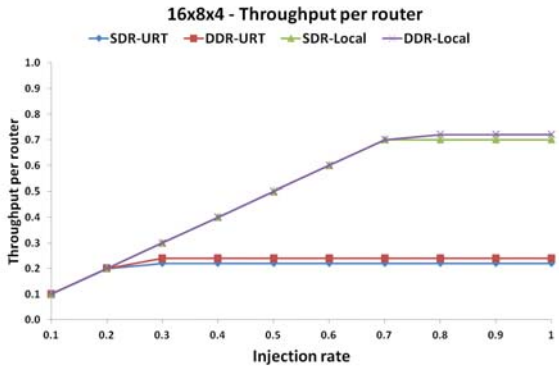
To contrast, Figures 4.7a, 4.7b and 4.7c show the performance of 4×8×16 vertically biased asymmetric cuboid for SDR and DDR routers. Figure 4.7b demonstrates a higher throughput gain than compared to the horizontally favoured configuration and symmetrical cube cases. This is clearly evident due to the increased number vertically oriented links that offer increased bandwidth due to the faster clock, hence less load than the SDR URT case under higher injection rates. As with all network configurations, the locality has improved the performance for both SDR and DDR networks.

4.7 Summary

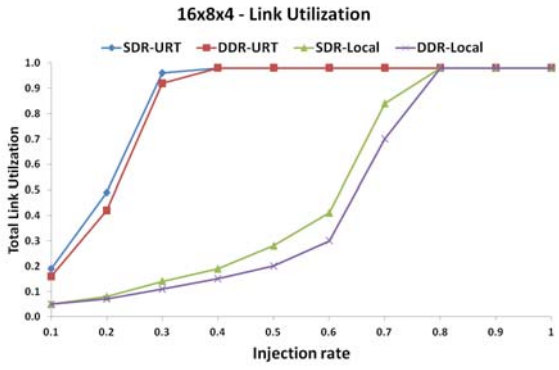
We have carried out an investigation on 3D NoCs where the fast properties of TSVs have been exploited to allow multi-rate clocking of the vertical network links. The physical capability of the vertical links to be clocked several times faster than the horizontal links has been justified. The topological properties of a SDR and DDR



(a) Latency

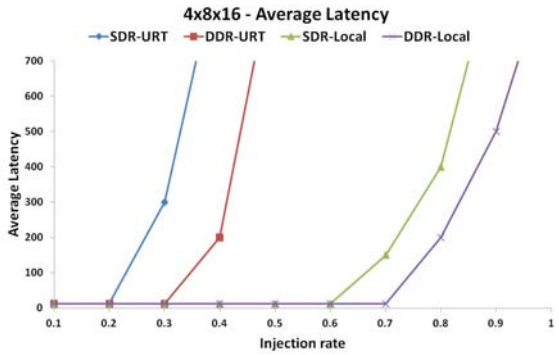


(b) Throughput

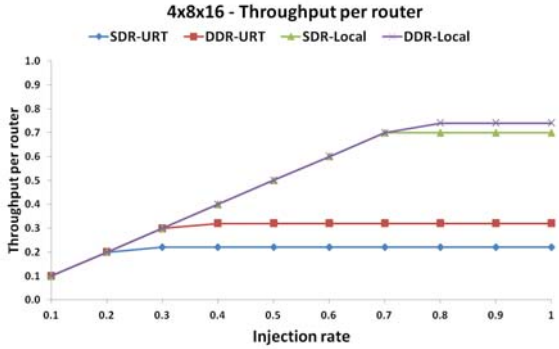


(c) Link utilization

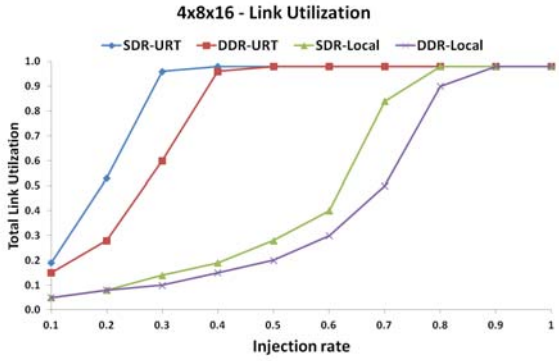
Figure 4.6: Performance of 16x8x4 network



(a) Latency



(b) Throughput



(c) Link utilization

Figure 4.7: Performance of 4x8x16 network

NoC have been analytically investigated. A new DDR router design capable of routing twice as many packets vertically as horizontally per link per cycle has been implemented and cycle-accurate simulations carried out. Network sizes, shapes and injection rates have been altered in conjunction with various traffic patterns to highlight the performance benefits of faster vertical links. The results quantify the significant improvement in throughput and latency as the number of vertical layers increase with the DDR router implementation over the SDR router.

In large networks especially, the increased power consumption from the horizontal links will encourage lower numbers of repeater stages and thus lower clock frequencies, leading to an even greater vertical to horizontal clock ratio as the TSVs do not require any repeater stages or large drivers.

Chapter 5

Performance Models for Deflection Routing

In previous chapters, we have shown that in order to gain performance increase, many-core architectures can be scaled up by increasing the network size, by changing the topologies, or by widening the bandwidth required for inter-core data communication. Each case was validated by running cycle-accurate simulations under various traffic scenarios. Relevant data are extracted from the simulations to measure the performance the architectures in terms of throughput and latency.

In practice, the traffic scenarios represent applications executed in many-core processors. Designers first check if specific applications can efficiently run in selected architectures. Prototyping each architecture is expensive and complex process. Thus the designers are left to depend on analytical models and NoCs simulation platforms to conduct the experiments.

Several methods are used to analytically model network performance. The curve fitting approximation uses simulated data as a basis in order to extract and formulate the models. Such method, even though fast, is dependent on the accuracy of the simulated output. Its usage is also restricted only to the type of simulated architecture. Thus, curve fitting hardly addresses scalability issues. A different method is to measure NoCs performance by deriving the worst case delay [42]. However most of these models are dedicated to NoCs with deterministic routing.

The alternative to the use of analytical modeling is to directly simulate networks with real or synthetic traffic patterns and extract the necessary information from the resultant data. Such extraction, though usually produces accurate information, is time consuming. In addition, since the simulation methodology varies from a designer to a designer, there is no consistency or benchmark to evaluate contrasting systems. Thus, in the absence of accurate models and simulation platforms, measuring the performance of specific applications becomes a key issue in the design of many-core architecture. In this chapter, we address the issue by proposing a zero-load (networks with no traffic) model that predicts NoCs performance based

on average distance ($\overline{\mathbb{H}}$) of networks in many-core architecture. The model captures static properties of the network topology and the spatial distribution of traffic, but does not take into account traffic load and congestion.

The study is based on network latency in relation to average distance ($\overline{\mathbb{H}}$) of the network. The predictive power of $\overline{\mathbb{H}}$ is demonstrated by showing near perfect fidelity. It means that for two equal size networks 'A' and 'B', the model predicts the relative performance of network 'A' is consistently less than network 'B' for any topology with deflection routing, whether homogeneous or heterogeneous, under numerous realistic traffic scenarios.

5.1 Related Works

There are numerous works in the literature that propose analytical models to estimate, or predict latency in various network architectures. Some techniques are applied for zero-load delay, while others consider intermediate network buffering under varying work-load delays. Given that most analytical models take congestion level in to account, the zero-load delay in such models can be deduced by putting the congestion level down to zero. However, such approach doesn't guarantee the accurate prediction of the relative performance under load. Depending on the switching mechanism, and routing algorithm of the network, the fidelity may not hold true all the time.

Also, the application of such models is limited to homogeneous network topology with regular traffic patterns. In particular, the performance and behavior of homogeneous topologies such as k-ary n-cube type networks is widely studied. A pioneering work from Dally et al [19] proposed latency models for store-and-forward and wormhole switching methods with torus topologies. However, for on-chip networks, mesh-like topologies are more practical to design and implement than the torus architectures. Agarwal [3] indicated that the network latency for two-dimensional mesh networks is lower than three or four dimensions under localized traffic. The analysis is performed for zero-load networks that eliminate the effect of congestion on the latency. In the zero-load case, the routers and wires are the only constraints that affect delay. Accordingly, the following expression is reported for the average distance in a k-ary n-mesh:

$$\overline{\mathbb{H}} = \frac{n}{3}(k - \frac{1}{k}) \tag{5.1}$$

In practice, networks come with unequal radices and makes the k-ary n-cube analysis less useful in most cases. For example a 3D cube network of size N with unequal radices, the cores can be arranged with different configurations of k_1, k_2, k_3 radices. In formulating a simple adaptive partitioning strategy with the aim of shortening the average distance for communication cost reduction, Liu et al in [49] derived an expression for average distance in $k_1 \times k_2$ type 2D mesh networks:

$$\overline{\mathbb{H}} = \frac{1}{3}\left(k_1 - \frac{1}{k_1}\right) + \frac{1}{3}\left(k_2 - \frac{1}{k_2}\right) \tag{5.2}$$

When $k_1 = k_2$, Equation (5.2) is equivalent to Agarwal's Equation (5.1).

5.2 Zero-load Average Distance Models

A packet injected with a given traffic pattern to a network without load travels the source-destination distance at zero-load. The general expression for average distance travelled by the packet in any 2D mesh and 3D cube networks including those with unequal radices can be formulated as follows.

Let's consider a 1-D network at zero-load where the packet with source i is sent to destination j with a probability $p_{i,j}$. Then at zero-load, average source-destination distance of the network is the ratio of the sum of distances travelled by each packet to the number of packets formulated as follows:

$$\overline{\mathbb{H}}_{1\times k} = \frac{\sum_{i=1}^{k}\sum_{j=1}^{k} p_{i,j} \times |i-j|}{\sum_{i=1}^{k}\sum_{j=1}^{k} p_{i,j}} \tag{5.3}$$

Details of the derivation of the zero-load average distance can be found in Paper IV [34].
Based on Equation (5.3) we derive average distance models for local random traffic in 2D mesh and 3D cube networks as follows:

$$\overline{\mathbb{H}}_{\alpha,i,j} = \frac{1}{N-1}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(|i-j|P_{i,j}) \tag{5.4}$$

Substituting Equation (2.5) stated for the alpha type local traffic pattern on Page 25 in Equation (5.4) results in:

$$\overline{\mathbb{H}}_{\alpha,i,j} = \frac{1}{N-1}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\frac{|i-j|^{1-\alpha}}{K_s} \tag{5.5}$$

More derivation of average distance for various spatio-temporal traffic patterns can be found in Paper I [79].

The accuracy average distance models are verified for varying traffic and network configurations in 2-D mesh and 3-D cubes. Such consistency of packet latencies in networks of equal size but of different configurations, unequal radices and traffic patterns are explored, modelled and verified.

Table 5.1 compares the average-distance estimated by the Zero-load models and cycle-accurate simulations for various traffic models and network sizes. The results show a good agreement between the formula and the simulation results, where the

discrepancy between the model and simulation is a result of stochastic deviation in the generation of packet destinations and rounding errors.

Zero-load models for other traffic patterns including bit-reverse, bit-complement, URT are also derived with the details published in Paper I [79], and Paper IV [34].

Network size	Traffic Pattern	Avg. Distance		% Error
		Zeroload Model	Simulation	
5x5x5	URT	4.8300	4.813	-0.35
6x6x6	URT	5.8600	5.888	0.47
7x7x7	URT	6.8772	6.971	1.36
8x8x8	URT	7.8900	7.931	0.52
9x9x9	URT	8.9000	8.976	0.85
10x10x10	URT	9.9090	9.894	-0.15
4x8x16	URT	9.9090	10.008	0.99
5x5x5	LRT: α=1.0	3.7900	3.81	0.53
6x6x6	LRT: α=1.0	4.5900	4.555	-0.94
7x7x7	LRT: α=1.0	5.3900	5.418	0.52
8x8x8	LRT: α=1.0	6.1900	6.146	-0.71
9x9x9	LRT: α=1.0	7.0000	6.969	-0.44
10x10x10	LRT: α=1.0	7.8060	7.855	0.62
5x5x5	LRT: α=1.5	3.1800	3.163	-0.53
7x7x7	LRT: α=1.5	4.4781	4.498	0.44
4x8x16	LRT α=1.5	5.3757	5.301	-1.38

Table 5.1: Comparison of average distance calculated by Zero-load model and cycle-accurate simulations

5.3 Evaluation of Regular Traffic Patterns

In our network architecture, the rate at which the packet travels varies depending on the age based priority arbitration scheme. A fresh packet injected from a source core to the network is given the profitable links that lead towards the destination only when there are no older packets competing for the same link. Otherwise, it is deflected and misrouted away from destination core. When there are no competing packets in the network, i.e., zero load delay, a packet arrives its destination at the network delay.

But, when more packets are injected and competing for links, the network traffic increases and leads to congestion. The level of congestion depends on the injection rate, the traffic pattern, and the number of links available in the network. Each network configuration differs from others by its capacity. As defined in Section 3, network capacity is the bandwidth measured by the number of available links.

Normally, each router has directional links connecting to the surrounding neighbors and local links connecting to a resource such as processing elements. However, when placed in a network, not all links of a router get connected. Depending on the position of the router in the network, some links are left unconnected. For 2D

router placed in one corner of a mesh network, only two of its directional links are connected; the other two are useless. However, the local link is always connected to a resource injecting packets. Thus, congestion increases in a network with more unconnected directional links because packets are forced to compete and occupy the few available links. This phenomenon in essence shows the architectural limitations of mesh like networks digestion bandwidth as some links are left unconnected.

To clearly show these limitations, let's quantify the number of links L_{xyz} given in Equation 3.2 on Page 36 into networks of unequal radices of 2D mesh with k_x,k_y dimension and 3D cube of k_x,k_y,k_z dimensions. Here, to calculate 2D meshes, we assume 3D cube (k_z=1).

In the case of same size network with unequal radices, the changes are therefore basically changes in the number of connected links. For 64 core network arranged in 8×8 (network A), there are L_{xyz}=224 directional links. But when arranged in 16×4 (network B) only L_{xyz}=216 directional links are connected. Thus, network B gets congested faster than network A and hence packets travel longer distances in hop-counts.

The second issue is related to the way we measure latency. Normally, hop-count is a measure of network distance. The average latency value in terms of clock cycles can be deduced by taking proportional ratio of the average hop-count to that of number of links in each radix. In 3D architectures with long planar interconnects that require clocked repeaters inserted in between, it may take more than one cycle for a packet to make a single hop from router-to-router. However, for vertical interconnects using TSV, it takes only one cycle to make a single hop as TSVs are short and also repeater insertion is not possible. Such physical properties add complexity to the hop-count analysis.

In order to calculate the latency in terms of cycles, first we find the number of links for each radix.

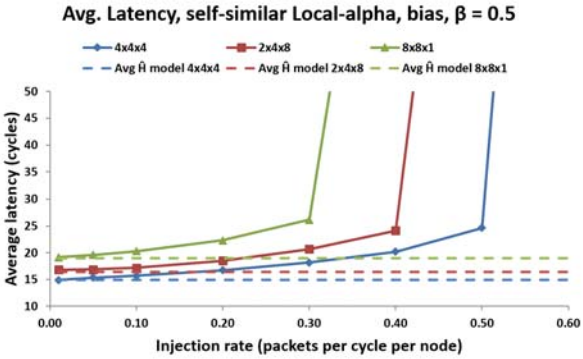
$$L_x = 2k_yk_z(k_x - 1) \tag{5.6}$$

$$L_y = 2k_xk_z(k_y - 1) \tag{5.7}$$

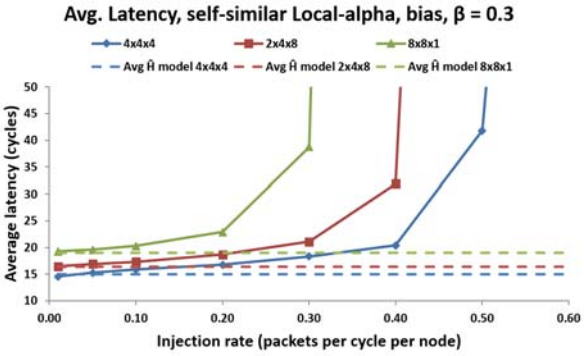
$$L_z = 2k_xk_y(k_z - 1) \tag{5.8}$$

For example, in the 64 core example, a 4×4×4 configuration gives L_{xyz}=288 links. With such equal radices, the vertical links take a third of the links, L_z=96. However, when the configuration is 2×4×8, though it has more vertical links L_z=112, the total number of links is only L_{xyz}=224.

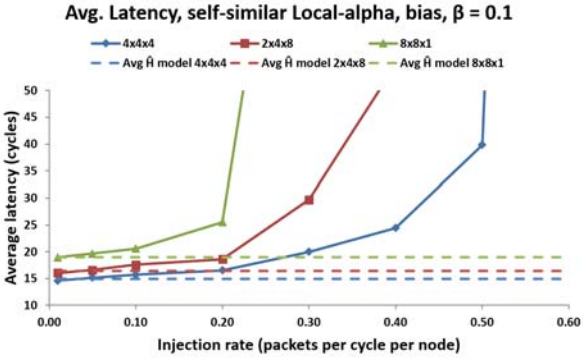
The average distance $\overline{\mathbb{H}}_{xyz}$ can be translated into latency in terms of cycles after separating the individual coordinates. This is done by taking the proportional ratio of any radix link of Equations 5.6, 5.7 and 5.8 to the total link L_{xyz} given in Equation 3.2 on Page 36 as follows.



(a) LRT with α=1 and bias β=0.5



(b) LRT with α=1 and bias β=0.3



(c) LRT with α=1 and bias β=0.1

Figure 5.1: Variation of average latency for LRT with α=1

$$\overline{\overline{\mathbb{H}}}_x = \overline{\overline{\mathbb{H}}}_{xyz} \frac{L_x}{L_{xyz}} \tag{5.9}$$

$$\overline{\overline{\mathbb{H}}}_y = \overline{\overline{\mathbb{H}}}_{xyz} \frac{L_y}{L_{xyz}} \tag{5.10}$$

$$\overline{\overline{\mathbb{H}}}_z = \overline{\overline{\mathbb{H}}}_{xyz} \frac{L_z}{L_{xyz}} \tag{5.11}$$

Then individual hop-count is multiplied by the number of cycles it takes for corresponding radix.

To validate the analysis for zero-load models, we made cycle-accurate simulations of networks of equal size 4×4×4, 2×4×8, and 8×8×1 in spatially localized traffic combined with a bursty traffic model (B-Model) temporal patterns. To our knowledge, no latency models have been published for spatio-temporal traffic patterns.

Figure 5.1a shows how the latency increases with injection rate when the localization coefficient α=1 and the self-similar bias β=0.5 , ensuring uniform streaming of packets under a local traffic pattern. At low injection rates the latency converges to the zero-load average distance as for the other cases. When the bias is set to β=0.3 (Figure 5.1b), or β=0.1 (Figure 5.1c), the resulting temporally skewed traffic causes insignificant changes. This is because strong localization in the traffic generation results in more packets with destinations within a relatively short distance compared to the network dimensions.

In each case, the average distance calculated by using zero-load model is equal to the simulation results at low injection rates. For increased injection rate, the hop-counts in each configuration also grow steadily without crossing each other exhibiting 100% fidelity.

Additional experiments are also conducted and published in Paper I [79] with more traffic patterns including bit-reverse, bit-complement, URT generating the same results and fidelity.

The average distance modeling approach is not only accurate in zero-load, but also shows 100% fidelity for all unsaturated work-loads. In all traffic scenarios, symmetrical configurations, such as 4×4×4, give the lowest average distance compared to other configurations of same network size. This is due to the fact that the architectural configuration of 4×4×4 allows the average distance travelled by a packet to be minimal as described above.

5.4 Evaluation of Irregular Traffic Patterns

In this section, we further validate the zero-load model for networks with irregular traffic patterns as well as irregular networks. We also show how such networks can be configured for optimal performance.

5.4.1 Networks with Hot-Spots

Cores that generate or receive a greater proportion of traffic than other cores are called hot-spots. Hot-spot regions should be designed in such a way that there is sufficient link bandwidth to support worst-case traffic congestion. We explore the optimal placement of hot-spot cores to minimise congestion.

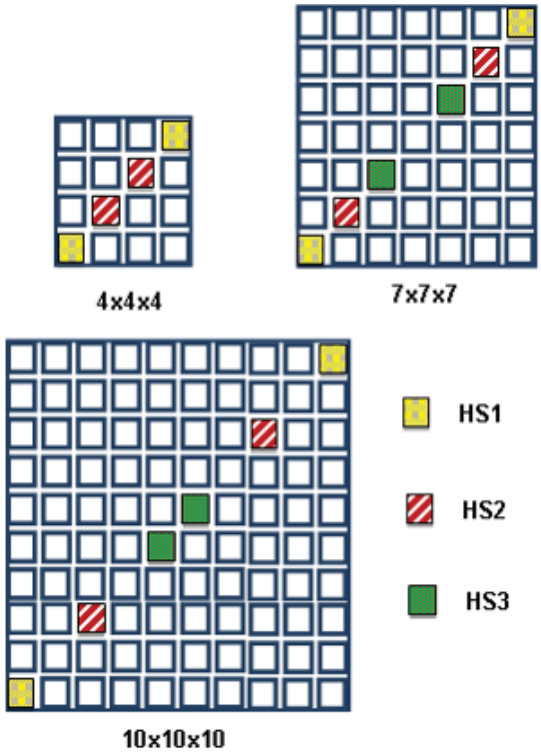


Figure 5.2: Placement of hot-spot cores on top layer

Let's consider the networks in Figure 5.2 showing different configurations of two hot-spot (HS) cores serving as access ports to DRAM either stacked in the same package or placed off-chip. Any processor core in the network can access the memory through these access ports. Since the memory is shared, the combined effect of all cores accessing the memory generates a hot-spot region with heavier traffic in the area surrounding the access ports. The key issue here is where is the best place in the network for the access ports in order to minimize the effect of hot-spot traffic and optimize the performance?

We conducted cycle accurate simulation to find the optimal placement of hot-spot cores. We examined networks of three different sizes, 4×4×4, 7×7×7, and

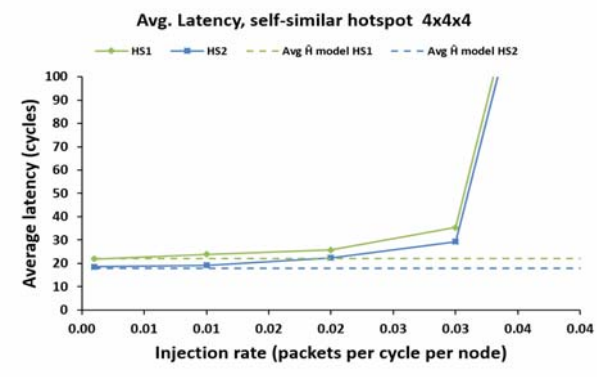


Figure 5.3: Variation of average latency with injection rate for hot-spot traffic in 4×4×4 network with HS1 and HS2 hot-spot placement on top layer

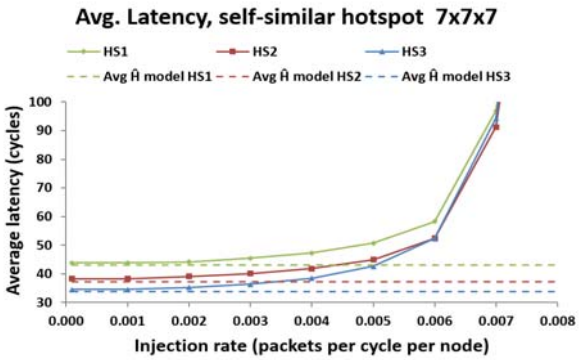


Figure 5.4: Variation of average latency with injection rate for hot-spot traffic in 7×7×7 network with HS1, HS2 and HS3 hot-spot placement

10×10×10. In each case, three selected placements shown as $HS1$, $HS2$, and $HS3$ are evaluated.

In this exercise hot-spot cores receive 80% of the packets generated by the non-hot-spot cores, while the remaining 20% are sent to other non-hot-spot destinations under a uniform random distribution. Figures 5.3 to 5.5 show the results for each network configuration with increasing injection rate from 0.001 up to 0.01 packets per core per cycle.

With increasing injection rate, the average packet latency in each configuration increases without the curves crossing each other. The model again exhibits perfect fidelity for all tested hot-spot configurations.

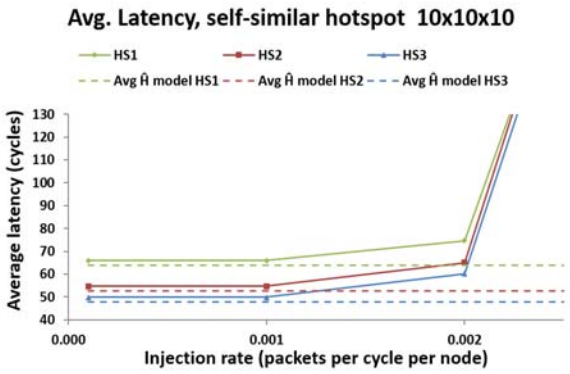
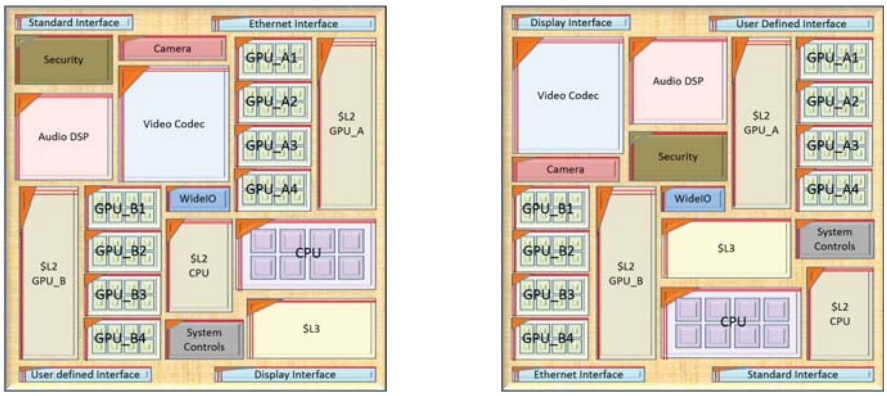


Figure 5.5: Variation of average latency with injection rate for hot-spot traffic in 10×10×10 network with HS1, HS2 and HS3 hot-spot placement

5.4.2 Configuration of Irregular Networks with Irregular Traffic Patterns

The zero-load predictive model is further investigated with an irregular network, based on two configurations of a generic mobile application processor (MAP) shown in Figure 5.6a and Figure 5.6b.



(a) MAP1 (b) MAP2

Figure 5.6: Two configurations of a generic mobile application processor

Table 5.2 shows the spatial probability distribution of traffic used to simulate the two MAP configurations, normalized to the GPU injection rate. For example, the relative injection rate of 0.3 for the security IP-core means that its traffic contribution is only 30% of the maximum traffic contribution by a core (i.e. the

Table 5.2: Traffic probabilities for MAP IP-cores

Source IP cores	Probability to target IP-core	Relative Injection rate
GPU	68% L2 GPU, 2% CPU, 20% Display Interface, 9% total to all other interfaces, 1% System control	1 IR
CPU	40% L2 CPU, 8% All GPU, 10% Audio, 10% Video, 4% Camera, 5% Security 22% all other Interface, 1% System control	0.7 IR
Audio	30% WideIO, 28% Security, 20% CPU, 15% Standard, 3% Ethernet, 3% User, 1% System control	0.2 IR
Video	50% WideIO, 9% Security, 20% CPU, 20% all interfaces, 1% System control	0.8 IR
Camera	30% WideIO, 60% Display, 5% CPU, 4% Security, 1% System control	0.8 IR
Security	60% WideIO, 20% Audio, 14% Video, 5% CPU, 1% System control	0.3 IR
L2 GPU	19% L3, 80% GPU, 1% System control	0.8 IR
L3	26% L2 GPU A, 26% L2 GPU B, 26% L2 CPU, 21% WideIO, 1% System control	1 IR
L2 CPU	20% L3, 79% CPU, 1% System control	0.8 IR
WideIO	48% L3, 15% Audio, 26% Video, 5% Security, 5% Camera, 1% System control	1 IR
System Control	24% CPU, 24% GPU, 4% To every remaining 13 cores,	0.2 IR
Standard Interface	16% GPU, 20% CPU, 46% Audio, 10% Video, 5% Security, 2% Camera, 1% System control	0.5 IR
User	24% GPU, 22% CPU, 24% Audio, 24% Video, 5% Security, 1% System control	0.5 IR
Ethernet	24% GPU, 25% CPU, 15% Audio, 30% Video, 5% Security 1% System control	0.5 IR
Display	64% GPU, 12% Video, 15% CPU, 12% WideIO, 3% Camera, 5% Security,1% System control	0.5 IR

GPU's contribution). The simulation results are shown in Figure 5.7, and the results confirm 100% fidelity of the model for injection rates below saturation.

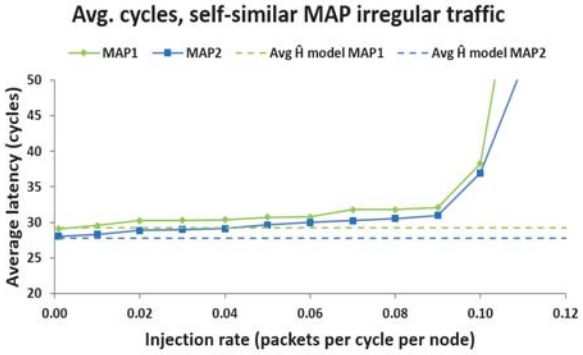


Figure 5.7: Average clock cycles with self-similar irregular traffic pattern for MAP configurations

5.5 Summary

In this chapter, we first derived zero-load average distance models for spatial traffic patterns and tested the fidelity of the models in different network scenarios with cycle-accurate simulations. The model predicts the zero-load delay of a network which is equivalent to the average distance of the network with traffic probabilities taken into account. By using the proposed model, the relative performance of networks is compared for a range of injections. We demonstrated that the models exhibit near-perfect fidelity for all investigated cases up to the network saturation. The fidelity of the models can be used to find solutions for optimum network architecture under various traffic patterns. All results consistently verified the correctness of the models in regular and irregular networks and traffics.

This work significantly extends the knowledge in the area of network modeling, with particular contributions in the extrapolation of average distance models to the optimization of networks architecture.

Chapter 6

Conclusion and Future Directions

6.1 Conclusions

First the thesis presented NoCs simulation platform for heterogenous systems. With each increase in level of heterogeneity, router micro-architecture with new routing algorithms and switching mechanisms is needed to characterize network and communication architecture. By using the simulation model, chip design issues such as system performance, processor-memory configuration, resource utilization etc. can be evaluated early in the design exploration phase. The proposed simulation platform is scalable enough to add more metrics other than performance which has been extensively used in this work in order to serve as a low cost integration and evaluation medium for heterogenous technologies.

We also investigated the scalability and functionality issues in NoCs. We have shown that for many-core architecture of 1000s of cores, deflection routing can be used. However we have also pointed out that in 1000 core processing, the NoCs performance gap keeps widening. Unless new algorithms or models are proposed, the gap will continue to be the major challenge in many-core architecture. The preliminary results shown in the architectural scalability of many-core architecture can be further extended by investigating additional traffic patterns, router architectures, and communication protocols to quantify more clearly the performance differences in the various network topologies.

We proposed a DDR TSV model that enhances the inter-layer communication. To enable the DDR, clock pumping technique is used. As shown through experiments, the model can be used best in vertically oriented network configurations. As an extension of the proposed DDR TSV, models can be prepared as IP core that implement quad-pumping technique to enable sending four inter-layer data packets in a single cycle. To ensure accuracy of the inter-layer data transmission, a vertical communication protocol can be developed as a standard. The protocol specification can be defined as an integral part of the TSV-IP core development.

Finally, we have also shown a zero-load model to predict the relative performance

of deflection routing networks. In designing many-core architecture with 1000-core networks, measuring performance through simulations is complex and expensive. The zero-load model can be helpful during the exploration phase of the design flow.

6.2 Future Directions

Architectural innovations in NoCs will continue to be the key trend in many-core design. The emergence of three-dimensional (3-D) integration technology has opened up new opportunities to explore vertically stacked heterogenous systems in many-core architecture. The integration of circuits using 3D technology keeps eroding the physical and electrical separation between chip, package, and ultimately board. With billions of transistors in a single 3D chip, the design complexity keeps increasing. Third party solutions provided as IP cores with different shapes and functionalities will be the smallest units to integrate.

The future of NoCs will be in providing a shared network resource to enable efficient communication of hundreds of IP cores. With such complexity, designers and system integrators will have to depend on simulation and analysis tools. It would be interesting to develop such comprehensive tool that help designers to evaluate, plan, and allocate resources within the network. All the necessary parameters including the injection bandwidth, the traffic pattern, frequency of communication, and the network size are passed as input to the tool. Models make iterations based on the input parameters to produce results.

A super chip is the 3D integration of dissimilar technology which is considered the ultimate single chip solution. A combination of logic-layer, memory-layer, MEMS layer, Radio communication layer and other layers create a complete system that can be applied for many purposes including chemical processing, neural computing, genetic sequencing, etc. In the future, if new and efficient routing algorithms and solutions are developed, NoCs can have a key role to play in the integration of such dissimilar technologies in a single super chip.

Current router intelligence is hardwired. For example to allow packet flow in irregular networks, the Q-routing algorithm uses reinforcement learning implemented in hardwired reconfigurable LUTs. With increasing router intelligence, future NoC solutions will likely be developed as an operating-system-like network services. The NoC operating-system will be a micro-code designed close to the hardware or inside the routers. This will give the flexibility to reconfigure the network on need basis. The main task of the NoC operating-system will be to manage the network as a shared resource used by all IP-cores and layers, to provide router level and link level services, and to facilitate the intra and inter-layer communication. It would be quite interesting to peruse ‘NoC operating-system’ as a future topic and as an extension of the research works described in the thesis.

Bibliography

[1] Charles Addo-Quaye. Thermal-aware mapping and placement for 3-d noc designs. In *SOC Conference, 2005. Proceedings. IEEE International*, pages 25–28. IEEE, 2005.

[2] V.S. Adve and M.K. Vernon. Performance analysis of mesh interconnection networks with deterministic routing. *Parallel and Distributed Systems, IEEE Transactions on*, 5(3):225–246, Mar 1994. ISSN 1045-9219.

[3] A. Agarwal. Limits on interconnection network performance. *IEEE Transactions on Parallel and Distributed Systems*, 4(6):613–624, 1991.

[4] AMBA ARM. Axi protocol specification (rev 2.0). 2010.

[5] Jun Ho Bahn and Nader Bagherzadeh. A generic traffic model for on-chip interconnection networks. In *NoCArc, First International Workshop on Network on Chip Architectures*, 2008.

[6] K. Banerjee, S. Souri, P. Kapur, and K. Saraswat. 3-d ics: A novel chip design for improving deep-submicrometer interconnect performance and systems-on-chip integration. *Proc. IEEE*, 89(5):602–633, May 2001.

[7] P. Baran. On distributed communications networks. *IEEE Transactions on Communications*, pages 1–9, 1964.

[8] Shane Bell, Bruce Edwards, John Amann, Rich Conlin, Kevin Joyce, Vince Leung, John MacKay, Mike Reif, Liewei Bao, John Brown, *et al.* Tile64-processor: A 64-core soc with mesh interconnect. In *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pages 88–598. IEEE, 2008.

[9] L. Benini and G. De Micheli. Networks on chips: A new soc paradigm. *IEEE Computer, pp.*, pages 70–78, January 2002.

[10] Davide Bertozzi, Antoine Jalabert, Srinivasan Murali, Rutuparna Tamhankar, Stergios Stergiou, Luca Benini, and Giovanni De Micheli. Noc synthesis flow for customized domain specific multiprocessor systems-on-chip. *Parallel and Distributed Systems, IEEE Transactions on*, 16(2):113–129, 2005.

[11] Tobias Bjerregaard and Shankar Mahadevan. A survey of research and practices of network-on-chip. *ACM Computing Surveys (CSUR)*, 38(1):1, 2006.

[12] Evgeny Bolotin, Israel Cidon, Ran Ginosar, and Avinoam Kolodny. Routing table minimization for irregular mesh nocs. pages 942–947, 2007.

[13] Justin A Boyan and Michael L Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. *Advances in neural information processing systems*, pages 671–671, 1994.

[14] F. Catthoor, N.D. Dutt, and C.E. Kozyrakis. How to solve the current memory access and data transfer bottlenecks: at the processor architecture or at the compiler level? In *Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pages 426–433, 2000.

[15] Ching-Che Chung and Chia-Lin Chang. A wide-range all-digital delay-locked loop in 65nm cmos technology. In *International Symposium on VLSI Design Automation and Test (VLSI-DAT*, pages 66–69, 2010.

[16] Henk Corporaal. Ttas: Missing the ilp complexity wall. *Journal of Systems Architecture*, 45(12):949–973, 1999.

[17] W. J. Dally. Performance analysis of k-ary n-cube interconnection networks. *IEEE Transactions on Computers*, 39(6):775–785, 1990.

[18] W. J. Dally and B. Towles. Route packets, not wires: on-chip interconnection networks. In *Proceedings of Design Automation Conference*, pages 684–689, 2001.

[19] William James Dally and Brian Patrick Towles. *Principles and practices of interconnection networks*. Elsevier, 2004.

[20] M. Daneshtalab, M. Ebrahimi, P. Liljeberg, J. Plosila, and H. Tenhunen. Memory-efficient on-chip network with adaptive interfaces. *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems (IEEE-TCAD)*, 31(1):146–159, 2012.

[21] Jose Duato, Sudhakar Yalamanchili, and Lionel M Ni. *Interconnection networks: An engineering approach*. Morgan Kaufmann, 2003.

[22] Denis Dutoit, Christian Bernard, Severine Cheramy, Fabien Clermidy, Yvain Thonnart, Pascal Vivet, Christian Freund, Vincent Guerin, Stéphane Guilhot, Stéphane Lecomte, *et al. A 0.9 pJ/bit, 12.8 GByte/s WideIO memory interface in a 3D-IC NoC-based MPSoC*. 2013.

[23] M. Ebrahimi. *Adaptive Routing Approaches For Networked Many-Core Systems*. Turun Yliopiston Julkaisuja, 2013.

[24] M. Ebrahimi, M. Daneshtalab, P. Liljeberg, J. Plosila, J. Flich, and H. Tenhunen. Path-based partitioning methods for 3d networks-on-chip with minimal adaptive routing. *IEEE Transaction on Computers (IEEE TC)*, 63(3):718–733, 2014.

[25] M. Ebrahimi, M. Daneshtalab, P. Liljeberg, J. Plosila, and H. Tenhunen. A high-performance network interface architecture for nocs using reorder buffer sharing. In *18th Euromicro Conference on Parallel, Distributed and Network-Based Computing (PDP)*, pages 547–550. IEEE, 2010.

[26] Project Elite:. *Extended Large (3-D) Integration Technology*. Retrieved in, 2011. URL `http://cordis.europa.eu/project/rcn/85238_en.html`.

[27] Dong-Rui Fan, Nan Yuan, Jun-Chao Zhang, Yong-Bin Zhou, Wei Lin, Feng-Long Song, Xiao-Chun Ye, He Huang, Lei Yu, Guo-Ping Long, *et al.* Godson-t: An efficient many-core architecture for parallel program executions. *Journal of Computer Science and Technology*, 24(6):1061–1073, 2009.

[28] B. Feero and P.P. Pande. Networks-on-Chip in a Three-Dimensional Environment: A Performance Evaluation. *IEEE Transactions on Computers*, 6, 2008.

[29] Chaochao Feng, Zhonghai Lu, Axel Jantsch, , and Minxuan Zhang. A 1-cycle 1.25ghz bufferless router for 3d network-on-chip. *IEICE Transactions on Information and Systems*, 5(95-D):1519–1522, May 2012.

[30] Chaochao Feng, Zhonghai Lu, Axel Jantsch, Jinwen Li, and Minxuan Zhang. A reconfigurable fault-tolerant deflection routing algorithm based on reinforcement learning for network-on-chip. pages 11–16, 2010.

[31] Crispín Gómez, María E Gómez, Pedro López, and José Duato. Reducing packet dropping in a bufferless noc. In *Euro-Par 2008–Parallel Processing*, pages 899–909. Springer, 2008.

[32] P. Gottschling, Haoyuan Ying, and K. Hofmann. Gsnoc ui - a comfortable graphical user interface for advanced design and evaluation of 3-dimensional scalable networks-on-chip. In International Conference, editor, *on High Performance Computing and Simulation (HPCS), Madrid*. - Spain, 2012.

[33] Matt Grange, Roshan Weerasekera, and Dinesh Pamunuwa. Optimal signaling techniques for through silicon vias in 3-d integrated circuit packages. In *Proc. IEEE Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*, pages 237–240, Oct 2010.

[34] Matt Grange, Roshan Weerasekera, Dinesh Pamunuwa, Axel Jantsch, and Awet Yemane Weldezion. Optimal network architectures for minimizing average distance in k-ary n-dimensional mesh networks. In *Proceedings of the*

Fifth ACM/IEEE International Symposium on Networks-on-Chip, pages 57–64. ACM, 2011.

[35] Matt Grange, Awet Yemane Weldezion, Dinesh Pamunuwa, Roshan Weerasekera, Zhonghai Lu, Axel Jantsch, and Dave Shippen. Physical mapping and performance study of a multi-clock 3-dimensional network-on-chip mesh. In *Proc. IEEE International Conference on 3D System Integration (3DIC)*, pages 1–7, Sept 2009.

[36] Boris Grot, Joel Hestness, Stephen W. Keckler, and Onur Mutlu. Kilo-noc: A heterogeneous network-on-chip architecture for scalability and service guarantees. *SIGARCH Comput. Archit. News*, 39(3):401–412, June 2011. ISSN 0163-5964.

[37] Mitchell Hayenga, Natalie Enright Jerger, and Mikko Lipasti. Scarab: A single cycle adaptive routing and bufferless network. In *Proceedings of the 42nd annual IEEE/ACM international symposium on microarchitecture*, pages 244–254. ACM, 2009.

[38] Ahmed Hemani, Axel Jantsch, Shashi Kumar, Adam Postula, Johnny Oberg, Mikael Millberg, and Dan Lindqvist. *Network on chip: An architecture for billion transistor era*, volume 31. 2000.

[39] John L Hennessy and David A Patterson. *Computer architecture: a quantitative approach*. Elsevier, 2011.

[40] Yang Hu, Shouyi Yin, Leibo Liu, and Shaojun Wei. A mixed-level modeling for network on chip infrastructure in soc design. In Ieee Asia, editor, *Pacific Conference on Circuits and Systems (APCCAS), Kuala Lumpur*. - Malaysia, 2010.

[41] A. Jantsch and H. Tenhunen (Eds.). Networks on chip. *Kluwer Academic Publishers pp.*, pages 7–18, 2003.

[42] A. E. Kiasari, A. Jantsch, and Z. Lu. Mathematical formalisms for performance evaluation of networks-on-chip. *ACM Computing Surveys*, 2013.

[43] Shailesh Kumar and Risto Miikkulainen. Dual reinforcement q-routing: An on-line adaptive routing algorithm. 1997.

[44] Shashi Kumar, Axel Jantsch, Juha-Pekka Soininen, Martti Forsell, Mikael Millberg, Johny Öberg, Kari Tiensyrjä, and Ahmed Hemani. A network on chip architecture and design methodology. pages 105–112, 2002.

[45] Nasser Kurd, Praveen Mosalikanti, Mark Neidengard, Jonathan Douglas, and Rajesh Kumar. Next generation intel core™ micro-architecture (nehalem) clocking. *Solid-State Circuits, IEEE Journal of*, 44(4):1121–1129, 2009.

[46] Patrick Leduca, F De Crecy, M Fayolle, B Charlet, T Enot, M Zussy, B Jones, J-C Barbe, N Kernevez, N Sillon, *et al.* Challenges for 3d ic integration: bonding quality and thermal management. In *International Interconnect Technology Conference, IEEE 2007*, pages 210–212. IEEE, 2007.

[47] S. E. Lee, J. H. Bahn, Y. S. Yang, and N. Bagherzadeh. A generic network interface architecture for a networked processor array (nepa). *In proc. ARCS'0*, 8:247–260, 2008.

[48] Feihui Li, Chrysostomos Nicopoulos, Thomas Richardson, Yuan Xie, Vijaykrishnan Narayanan, and Mahmut Kandemir. Design and management of 3d chip multiprocessors using network-in-memory. *ACM SIGARCH Computer Architecture News*, 34(2):130–141, 2006.

[49] H. Liu, W. Lin, and Y. Song. An efficient processor partitioning and thread mapping strategy for mesh-connected multiprocessor systems. In *Proc. ACM symposium on Applied computing*, 1997.

[50] Gabriel H. Loh. 3d-stacked memory architectures for multicore processors. In *International Symposium on Computer Architecture*, pages 453–464, 2008.

[51] Z. Lu and A. Jantsch. Admitting and ejecting flits in wormhole-switched networks on chip. *Computers and Digital Techniques, IET*, 1(5):546–556, Sept. 2007. ISSN 1751-8601.

[52] Mateusz Majer, Christophe Bobda, Ali Ahmadinia, and Jürgen Teich. Packet routing in dynamically changing networks on chip. In *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, pages 154b–154b. IEEE, 2005.

[53] J.L. Manferdelli, N.K. Govindaraju, and C. Crall. Challenges and opportunities in many-core computing. *Proceedings of the IEEE*, 96(5):808–815, 2008.

[54] Paul Marchal, Geert Van der Plas, Paresh Limaye, Abdelkarim Mercha, Vladimir Cherman, Herman O'Prins, Riet Labie, Bart Vandevelde, Youssef Travaly, and Eric Beyne. Verifying thermal/thermo-mechanical behavior of a 3d stack-challenges and solutions. In *VLSI Design Automation and Test (VLSI-DAT), 2010 International Symposium on*, pages 15–16. IEEE, 2010.

[55] H. Matsutani and M. Koibuchi. Tighly-coupled multi-layer topologies for 3-d nocs. *Int. Conf. on Parallel Processing ICCP, pp.*, 75, 2007.

[56] Christopher Mineo, Ravi Jenkal, Samson Melamed, and W Rhett Davis. Interdie signaling in three dimensional integrated circuits. pages 655–658, 2008.

[57] Thomas Moscibroda and Onur Mutlu. A case for bufferless routing in on-chip networks. In *Proceedings of the 36th Annual International Symposium on Computer Architecture*, ISCA '09, pages 196–207, 2009. ISBN 978-1-60558-526-0.

[58] Erland Nillson. Design and implementation of a hot-potato switch in a network on chip. *Mémoire, Department of Microelectronics and Information Technology, Royal Institute of Technology, IMIT/LECS*, 2002:21–25, 2002.

[59] Dongkook Park, Soumya Eachempati, Reetuparna Das, Asit K Mishra, Yuan Xie, Narayanan Vijaykrishnan, and Chita R Das. Mira: A multi-layered on-chip interconnect router architecture. 36(3):251–261, 2008.

[60] Ocp International Partnership. Open core protocol specification. 2.0 release candidate. 2003.

[61] Sudeep Pasricha. Exploring serial vertical interconnects for 3d ics. In *Proceedings of the 46th Annual Design Automation Conference*, pages 581–586. ACM, 2009.

[62] V.F. Pavlidis and E.G. Friedman. 3-D Topologies for Networks-on-Chip. *IEEE Transactions on Very Large Scale Integration Systems*, 15(10):1081, 2007.

[63] Manas Kumar Puthal, Virendra Singh, Manoj Singh Gaur, and Vijay Laxmi. C-routing: An adaptive hierarchical noc routing methodology. In *VLSI and System-on-Chip (VLSI-SoC), 2011 IEEE/IFIP 19th International Conference on*, pages 392–397. IEEE, 2011.

[64] P. A. Sandborn and H. Moreno. *Conceptual Design of Multichip modules and Systems*. Kluwer Academic Publishers, 1994.

[65] C. Seiculescu, S. Murali, L. Benini, and G. De Micheli. Sunfloor 3d: A tool for networks on chip topology synthesis for 3d systems on chips. In *Proc. of DATE*, pages 9–14, 2009.

[66] Philips Semiconductors. Device transaction level (dtl) protocol specification. *Version*, 2.2, July 2002.

[67] R. Terrill and Gl. Beene. 3-d packaging technology overview and mass memory applications. *IEEE Aerospace Applications Conference, TI Inc TX Dallas pp.*, pages 347–355, 1996.

[68] Anna W Topol, DC La Tulipe, Leathen Shi, David J Frank, Kerry Bernstein, Steven E Steen, Arvind Kumar, Gilbert U Singco, Albert M Young, Kathryn W Guarini, *et al.* Three-dimensional integrated circuits. *IBM Journal of Research and Development*, 50(4.5):491–506, 2006.

[69] Sriram R Vangal, Jason Howard, Gregory Ruhl, Saurabh Dighe, Howard Wilson, James Tschanz, David Finan, Arvind Singh, Tiju Jacob, Shailendra Jain, *et al.* An 80-tile sub-100-w teraflops processor in 65-nm cmos. *Solid-State Circuits, IEEE Journal of*, 43(1):29–41, 2008.

[70] Girish Varatkar and Radu Marculescu. On-chip traffic modeling and synthesis for MPEG-2 video applications. *IEEE Trans. on VLSI Syst*, 12(1):108–119, 2004.

[71] Mengzhi Wang, Tara Madhyastha, Ngai Hang Chan, Spiros Papadimitriou, and Christos Faloutsos. Data mining meets performance evaluation: Fast algorithms for modeling bursty traffic. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 507–516. IEEE, 2002.

[72] Sung-Ho Wang, Jeongpyo Kim, Joonsuk Lee, Hyoung Sik Nam, Young Gon Kim, Jae Hoon Shim, Hyung Ki Ahn, Seok Kang, Bong Hwa Jeong, Jin Hong Ahn, *et al.* A 500-mb/s quadruple data rate sdram interface using a skew cancellation technique. *Solid-State Circuits, IEEE Journal of*, 36(4):648–657, 2001.

[73] R. Weerasekera. *System Interconnection Design Trade-offs in Three-Dimensional Integrated Circuits*. PhD thesis, The Royal Institute of Technology (KTH), Stockholm, Sweden, 2008.

[74] R. Weerasekera, M. Grange, D. Pamunuwa, and H. Tenhunen. On signalling over through-silicon via (tsv) interconnects in 3-d integrated circuits. In *Proc*, pages 1325–1328, Germany, 2010. Design Automation and Test in Europe (DATE) Conference.

[75] R. Weerasekera, M. Grange, D. Pamunuwa, H. Tenhunen, and L. r. Zheng. Compact modelling of through-silicon vias (tsvs) in three dimensional (3-d) integrated circuits. In *Proc*. IEEE International Conference on 3D System Integration (3D IC) p. in press, 2009.

[76] Roshan Weerasekera, Dinesh Pamunuwa, Li-Rong Zheng, and Hannu Tenhunen. Two-dimensional and three-dimensional integration of heterogeneous electronic systems under cost, performance, and technological constraints. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 28(8):1237–1250, 2009.

[77] A. Y. Weldezion, Z. Lu, R. Weerasekera, and H. Tenhunen. 3-d memory organization and performance analysis for multi-processor network-on-chip architecture. In *Proc*. IEEE International Conference on 3D System Integration (3DIC 2009, September 2009.

[78] A. Y. Weldezion, R. Weerasekera, D. Pamunuwa, L. Zheng, and H. Tenhunen. Bandwidth optimization for through silicon via(tsv) bundles in 3d integrated circuits. pages 283–287. 3D Integration Workshop, The Design, Automation and Test in Europe (DATE) Conference, 2009.

[79] Awet Yemane Weldezion, Matt Grange, Axel Jantsch, Hannu Tenhunen, and Dinesh Pamunuwa. Zero-load predictive model for performance analysis in

deflection routing NoCs. *Microprocessors and Microsystems*, 39(8):634–647, 2015. ISSN 0141-9331.

[80] Awet Yemane Weldezion, Matt Grange, Dinesh Pamunuwa, Axel Jantsch, and Hannu Tenhunen. A scalable multi-dimensional noc simulation model for diverse spatio-temporal traffic patterns. In *3D Systems Integration Conference (3DIC), 2013 IEEE International*, pages 1–5. IEEE, 2013.

[81] Awet Yemane Weldezion, Matt Grange, Dinesh Pamunuwa, Zhonghai Lu, Axel Jantsch, Roshan Weerasekera, and Hannu Tenhunen. Scalability of network-on-chip communication architecture for 3-d meshes. In *Proceedings of the 2009 3rd ACM/IEEE International Symposium on Networks-on-Chip*, pages 114–123. IEEE Computer Society, 2009.

[82] A.Y. Weldezion, M. Ebrahimi, M. Daneshtalab, and H. Tenhunen. Automated power and latency management in heterogeneous 3d nocs. Waikiki, Hawaii, USA, December 2009.

[83] A.Y. Weldezion, R. Weerasekara, and H. Tenhunen. Design space exploration of clock-pumping techniques to reduce through-silicon-via (tsv) manufacturing cost in 3-d integration. In *Electronics Packaging Technology Conference (EPTC), 2012 IEEE 14th*, pages 19–22, Dec 2012.

[84] D. Wu, B. M. Al-Hashimi, and M. T. Schmitz. Improving routing efficiency for network-on-chip through contention-aware input selection. In *Proceedings of Asia and South Pacific Conference on Design Automation*, pages 36–41, 2006.

[85] Menwang Xie, Duoli Zhang, and Yao Li. Meshim: A high-level performance simulation platform for threedimensional network-on-chip. In *IEEE 9th International Conference on ASIC (ASICON)*, pages 349–352. Xiamen - China, October 2011.

[86] Young Jin Yoon, N. Concer, and L. Carloni. Ventti: a vertically integrated framework for simulation and optimization of networks-on-chip. *IEEE International SOC Conference (SOCC), pp.*, pages 171–176, 2012.

[87] Zhen Zhang, Alain Greiner, and Sami Taktak. A reconfigurable routing algorithm for a fault-tolerant 2d-mesh network-on-chip. In *Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE*, pages 441–446. IEEE, 2008.