TUCS

Alexander Wei Yin

On Energy Efficient Computing Platforms

Turku Centre *for* Computer Science

TUCS Dissertations
No 160, August 2013

On Energy Efficient Computing Platforms

Alexander Wei Yin

To be presented, with the permission of the Faculty of Mathematics and Natural Sciences of the University of Turku, for public criticism in Auditorium Lambda on August 28, 2013, at 12 noon.

University of Turku
Department of Information Technology
20014 Turku, Finland

2013

Supervisors

Professor Hannu Tenhunen
Department of Information Technology
University of Turku
20014 Turku, Finland

Associate Professor Pasi Liljeberg
Department of Information Technology
University of Turku
20014 Turku, Finland

Professor Jouni Isoaho
Department of Information Technology
University of Turku
20014 Turku, Finland

Reviewers

Professor Maurizio Palesi
Kore University
94100 Cittadella Universitaria
Enna, Italy

Professor Shashi Kumar
Department of Electronics and Computer Engineering
Jönköping University
SE-551 11 Jönköping, Sweden

Opponent

Professor Zheng Yan
Docent, School of Electrical Engineering
Aalto University, Espoo, Finland
Professor, School of Telecommunications Engineering
Xidian University, Xi'an, China

ISBN 978-952-12-2918-3
ISSN 1239-1883

Abstract

In accordance with the Moore's law, the increasing number of on-chip integrated transistors has enabled modern computing platforms with not only higher processing power but also more affordable prices. As a result, these platforms, including portable devices, work stations and data centres, are becoming an inevitable part of the human society. However, with the demand for portability and raising cost of power, energy efficiency has emerged to be a major concern for modern computing platforms.

As the complexity of on-chip systems increases, Network-on-Chip (NoC) has been proved as an efficient communication architecture which can further improve system performances and scalability while reducing the design cost. Therefore, in this thesis, we study and propose energy optimization approaches based on NoC architecture, with special focuses on the following aspects.

As the architectural trend of future computing platforms, 3D systems have many benefits including higher integration density, smaller footprint, heterogeneous integration, etc. Moreover, 3D technology can significantly improve the network communication and effectively avoid long wirings, and therefore, provide higher system performance and energy efficiency.

With the dynamic nature of on-chip communication in large scale NoC based systems, run-time system optimization is of crucial importance in order to achieve higher system reliability and essentially energy efficiency. In this thesis, we propose an agent based system design approach where agents are on-chip components which monitor and control system parameters such as supply voltage, operating frequency, etc. With this approach, we have analysed the implementation alternatives for dynamic voltage and frequency scaling and power gating techniques at different granularity, which reduce both dynamic and leakage energy consumption.

Topologies, being one of the key factors for NoCs, are also explored for energy saving purpose. A Honeycomb NoC architecture is proposed in this thesis with turn-model based deadlock-free routing algorithms. Our analysis and simulation based evaluation show that Honeycomb NoCs outperform their Mesh based counterparts in terms of network cost, system performance as well as energy efficiency.

Acknowledgements

The completion of this doctoral dissertation is an exciting moment in my life. During the past years, I have received enormous support from several people and organizations, to whom I would like to express my sincere gratitude.

First of all, I am extremely grateful to my supervisors Prof. Hannu Tenhunen, Prof. Pasi Liljeberg and Prof. Jouni Isoaho from the University of Turku. From their innovativeness, diligence and professionalism, I have learned not only how to conduct research, but also how to conquer challenges.

I would like to express my appreciation to Prof. Axel Jantsch and Prof. Zhonghai Lu from the Royal Institute of Technology (KTH), Sweden where I was introduced to the Network-on-Chip technology which is one of the corner stones of this doctoral dissertation.

In addition to my supervisors, I would sincerely thank my fellow colleagues at University of Turku, Turku Centre for Computer Science, Turku School of Economics, as well as Eficode Oy.

I would also like to thank Prof. Maurizio Palesi from Kore University, Italy and Prof. Shashi Kumar from Jönköping University, Sweden for the detailed reviews and constructive comments on this dissertation.

Last but definitely not least, my most gratitude goes to my family. Being ICT professionals themselves, my parents have been always supporting and understanding my career and life choices. I would give my most special appreciation and love to my dear wife Wenjing and son Jeremy for all their love, understanding and support.

Contents

Chapter 1

Introduction

Gone are the days when Integrated Circuit (IC) designers focused solely on the creation of faster chips or more powerful processors. With the development of computer technologies and network infrastructure, a large variety of computing platforms are becoming unprecedentedly important in many aspects of the human society. At the same time, energy efficiency has been recognized as the most critical challenge and has drawn significant attention from both industry and academia. In this chapter, we will firstly explain the energy challenges for different types of computing platforms.

1.1 Energy Efficient Computing Platform

A computing platform includes a hardware architecture and a software framework where the combinational efforts allow application software to be run. With the current technologies, all computing platforms are powered by either AC power or batteries. In this section, we introduce the energy concerns for portable devices which are powered by batteries as well as servers and data centres using AC power.

1.1.1 Portable Devices

During the past decades, the demand and popularity of portable devices has been growing tremendously, such as laptops, smart phones, mobile Internet devices, pocket video recorders, digital single-lens reflex cameras and hand-held game consoles. These devices conveniently provide many services including communication, computation, entertainment, information management and Internet access [66]. Smart phones, for example, have gained extreme popularity thanks to the increasing processing capability, powerful functionalities and applications, as well as easy access to Internet via 2G, 3G, Blue-tooth and Wi-Fi networks. Figure 1.1 illustrates the market growth during the previous years and forecast until the year 2015.

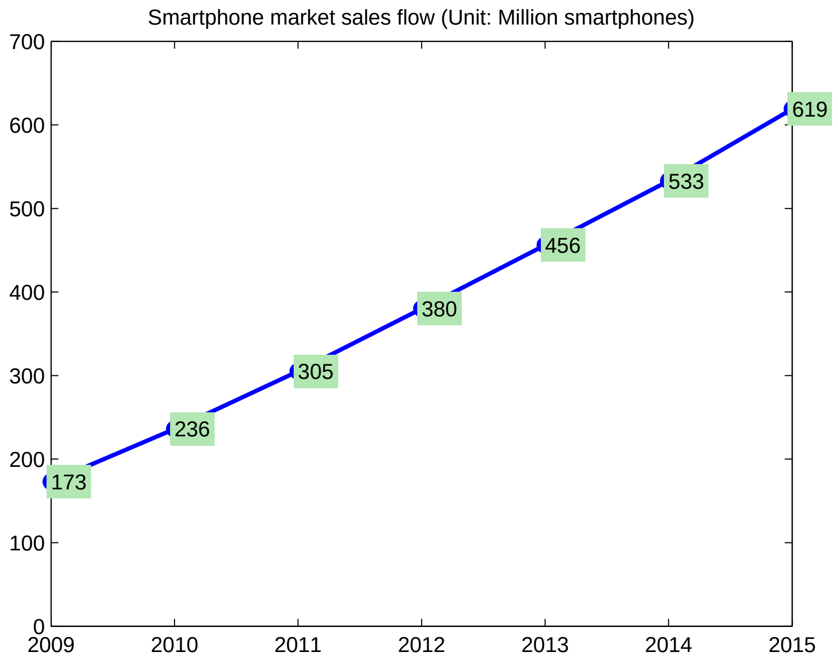


Figure 1.1: Market and Sales Growth of Smart phones

Portable devices commonly run on rechargeable batteries to support user mobility. The requirements for small size and light weight imply that its battery be proportionately small in volume [73]. According to Moore's Law, the number of on-chip integrated transistors has doubled approximately every 18 months and thus power consumption of processors has increased significantly. Over the last 30 years, the power consumption of microprocessors has gone up from under 1 Watt [90] to over 130 Watts (e.g., Intel i7 9-Series). However, the development of battery technology has not kept pace. Since 1990s, battery energy density has only tripled [13].

Therefore, energy efficiency is of obvious importance for portable devices since people like neither replacing or re-charging their batteries frequently, nor carrying heavy batteries with their sleek gadgets.

1.1.2 Servers and Data Centres

Unlike battery operated portable devices, workstations, servers and data centres are usually operated by AC power and are not restricted by battery capacity. However, there are at least two important energy related aspects that make energy efficiency a crucial concern.

Environmental Cost **-** According to the U.S. Environmental Protection Agency's Report to Congress on Server and Data Center Energy Efficiency made in the year 2007 [3], the aggregate electricity use associated with U.S. servers and data centres is expected to rise from 61.4 billion kWh in 2006 to 124.5 billion kWh in 2011, representing about 2.5 percent of total projected nationwide electricity use. Many organizations have expressed severe concerns over the rising electricity consumption attributed to the Internet and to commercial desktop computation including workstations and servers. This is because of not only the rising financial cost, but also the process of electricity generation which is a major source of pollution. Therefore, inefficient energy usage in computing equipment will exacerbate the currently challenging enough environmental problems.

Reliability and Heat Concerns - In addition to environmental cost, there are concerns of reliability and heat dissipation. Computer platforms with high energy consumption tend to increase the silicon temperature, and high temperature will exacerbate various failure mechanisms. According to [34], every 10°C increase in operating temperature roughly doubles an electrical component's failure rate. For modern computing platforms, the die surface temperatures are already over 100°C and thus sophisticated packaging and cooling mechanisms are required. This fact has increased system cost substantially. As presented in [90], a processor with less than 1 W power consumption requires a plastic package costing about 1 cent per pin, while if the power consumption is over 2 W, ceramic packages costing about 5 cents per pin would be required. For data centres, high energy consumption also leads to the usage of large scale and expensive liquid and air cooling solutions [78].

1.2 System-on-Chip Development and Challenges

With the digitalization of modern society, embedded systems have spanned all aspects and play increasingly important roles in people's everyday lives. An embedded system is a computing platform designed for a specific function, often within a larger system such as a personal computer (PC). It is embedded as a part of a complete device which often includes both hardware and software systems.

1.2.1 Development Progress of Embedded Systems and Electronic Devices

The Apollo Guidance Computer (AGC) is usually recognized as one of the first modern embedded systems. It was designed at the MIT Instrumentation Laboratory in early 1960s. Serving as a digital computer produced for the Apollo program, it was installed on-board each Apollo Command

Module and Lunar Module. The AGC provided computation and electronic interfaces for guidance, navigation, and control of the spacecraft.

At its early stages, embedded systems were used intensively for military purposes. Autonetics D-17, an early mass produced embedded system, was the guidance computer for the Minuteman missile, released in 1961. It was built from transistor logic and had a hard disk for main memory. The main reason for the lack of commercial applications was the price for quad nand gate ICs which was as high as 1000 USD per piece.

Another milestone of electronic systems is the invention of Dynamic Random-Access Memory (DRAM) by IBM in late 1960s. It stored each bit of data in a separate capacitor within an integrated circuit. The capacitor can be either charged or discharged; these two states are taken to represent the two values of a bit, conventionally called 0 and 1.

In 1971, Intel launched the first microprocessor known as Intel 4004 which cost 60 USD each. This 4-bit processor was also the first complete Central Processing Unit (CPU) on one chip.

Over the next several years, Intel 8008 and other more capable microprocessors became available in the IC industry. However, processors still required external chips to implement a working systems, which hindered their commercial usages due to high cost. To solve this problem, Texas Instrument released the first micro-controller known as TMS 1000 in 1974, which integrated processor, memory and clock on one chip.

In 1980s, the introductions of Application-Specific Integrated Circuit (ASIC) and Field-Programmable Gate Array (FPGA) brought revolutionary changes to the IC industry. The initial ASICs used gate array technology known as uncommitted logic array which was invented by Ferranti. This technology was successfully commercialized via 8-bit ZX81 and ZX Spectrum low-end personal computers in early 1980s. Later in 1985, the co-founders of Xilinx invented the first commercially viable field programmable gate array the XC2064 which had programmable gates as well as programmable interconnects between gates.

In 1990s, the introduction of intellectual property (IP) blocks and virtual components brought in a new implementation format called System-on-Chip (SoC) where a single chip contained mostly reusable IP based logic blocks. Comparing with the aforementioned systems, SoC is a completely integrated on-chip system and has much powerful processing capabilities.

1.2.2 SoC Overview

SoC is a chip which holds all the necessary hardware and electronic circuitry for a complete system. It normally contains a large number of components integrated on a single substrate, such as microprocessor, micro-controller, memory blocks (ROM, RAM or flash), I/O logic control, data converters, etc. Peripherals such as USB interface and communication modules including radio frequency module and Ethernet components are also commonly used in SoCs.

With the increasing number of integrated components, on-chip communication is undoubtedly one of the most critical issues and can easily become a performance bottleneck. In current SoC based systems, the most commonly used communication medium among different components is based on the on-chip bus architecture.

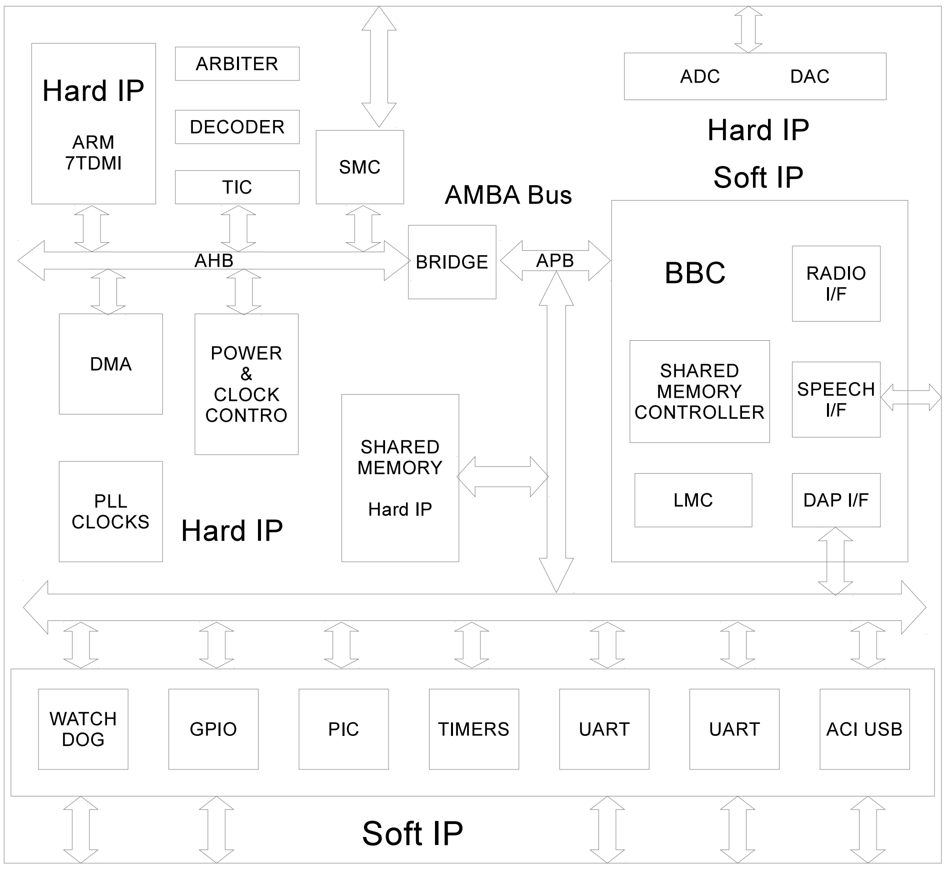


Figure 1.2: An Example of SoC [86]

As an example of SoC implementation, Figure 1.2 shows the block diagram of a mix-signal SoC developed at the University of British Columbia [86]. This SoC is a Bluetooth device for low-power wireless personal area networks (WPANs) and its overall architecture consists of a RF front-end, a baseband controller, and software to implement the Bluetooth protocol stack.

As shown in Figure 1.2, one major advantage of SoCs is the usage of IP cores, which greatly shortens the production lead time. An IP core is usually a hardware circuit design that is lent, sold, or licensed from one designing or engineering company to another. There are two main categories of IP cores, namely, hard IP cores and software IP cores.

Hard cores refer to the IP circuits that are delivered to the customer in the form of a final silicon layout. It is worthwhile to notice that hard cores do not mean the physical chips of silicon, but all the necessary information to manufacture such a chip. This is the most restrictive form for delivery, but also provides the highest reliability. Delivering a hard core leaves little room for the recipient of the IP to make any manufacturing mistakes. Hard cores are usually delivered in the form of a GDS-II database.

Soft cores, also known as synthesizable cores, do not have to be manually ported to specific manufacturing process. Thus, soft cores avoid the problems with process porting that hard cores have. Soft cores can be synthesized for different types of silicon processes, and the conversion is now automated hence becomes the responsibility of the synthesis tools, rather than the IP supplier or customer. Soft cores delivery makes IP cores more universal and less specific to individual customers or their silicon manufacturing processes. However, comparing with hard cores, software cores usually cost more silicon area, manufacturing cost and power consumption since they are not explicitly optimized for a specific technology process.

1.2.3 Challenges for SoCs

During the past years, the transistor counts on a single chip have entered the billion era for both microprocessors and graphical processing units. As a result, the number of IP blocks integrated on chip has been increasing dramatically which leads to an exponential rise in the complexity of their interaction. With this technology trend, it is predictable that the traditional digital system design methods, especially bus based SoC designs, will soon face the following performance bottleneck and design challenges [44] [63].

***Communication verses Computation* -** As the technology develops, the scale of on-chip systems grows exponentially. With the use of reusable IP blocks, the design of computing component and circuit is no longer the most critical challenge in system design since the reusable IP blocks can provide satisfying circuit features and performances. Researches have unveiled that the data transmission and communication have profound effects on system performance and communication often becomes the bottleneck in large scale systems.

***Deep Submicron Effects* -** Signal integrity effects such as interconnect delay, crosstalk, substrate coupling, transmission line effects, etc. did not draw significant attention in the early era of IC design with relatively slow clock speed and low integration density. However, as the transistor density increases, wires are getting neither fast nor reliable [37]. Meanwhile, more noise sources due to inductive fringing, crosstalk and transmission line effect are coupled to other circuit nodes via substrate, common ground and electromagnetic interference [63]. Consequently, signal integrity analysis has become as important as timing, area and power analysis in modern chip designs.

Global Synchronization - IC designs have been following a globally synchronous design style where a global clock tree is distributed throughout the chip, driving logic blocks to function synchronously. However, this style is unlikely to hold for future wire interconnect. With the technology scaling, gate delay has been reduced more significantly than wire delay. Therefore, in large scale SoCs with small feature size, it usually takes several clock cycles for the clock signal to transfer from one component to another. It is predicted in [44] that large chips will no longer be synchronous designs with only one clock domain.

Design and Productivity Gap - The design and productivity gap is the gap between what we are capable of building and what we are capable of designing [63]. According to [9], the development of compiling and synthesizing tools does not keep pace with the IC manufacturing technology. This problem has been alleviated thanks to the reusable IP blocks which can largely reduce the design time of computing components in systems. Therefore, It is naturally expected that the on-chip communication architecture is also reusable in different systems.

In order to solve these challenges, the Network-on-Chip (NoC) architecture was introduced at the beginning of 21st century based on the existing computer network principles. By adopting this architecture, the performance of inter-component intra-chip communications can be largely improved. Since there is no arbitration requirement in a network, more transaction can occur simultaneously and thus let a resource (usually an IP block as a component in the network) transmit a message whenever it is ready, which can decreases the response time (delay) of messages and enhance the usage as well as the throughput of a system. Moreover, as each link in a NoC system is based on point-to-point (P2P) mechanism, the network wires and consequently the communications among resources can be pipelined. In terms of scalability, network architectures can be increased by inserting repeaters without severe performance degradation. The details of NoC architecture will be introduced in the next chapter.

1.3 Thesis Contribution

The main contributions of this thesis are:

1. To achieve higher energy efficiency, different Three Dimensional (3D) NoC implementations are studied with the analysis on benefits and

limitations. We have presented a 3D on-chip DRAM architecture which alleviates the thermal challenges. A quarter connection TSV placement architecture is proposed to reduce the total TSV cost with little performance cost.

2. A 2D/3D change function is presented in this thesis, which quantifies the pros and cons of each architecture. The principle of this function is based on the analysis of perceived crises using 2D NoC architectures and pains of 3D NoC adoption. While illustrating the change function, we have compared 2D and 3D NoCs in terms of benefits, including communication delay, power consumption, area consumption, reduction in the number of metal layers, as well as costs including cooling, TSV manufacturing, yield and testing.

3. A hierarchical monitoring agent based design approach is proposed in this thesis where agents are either software or hardware components monitoring and configuring the operations on a NoC based system. By the joint efforts of all agents at different hierarchical levels, the system is able to perform autonomous optimizations during run-time and thus achieve higher system robustness and energy efficiency. Low power techniques including DVFS and power gating are studied using the proposed agent based architecture. Together with a reinforcement learning adaptive routing algorithm, this approach effectively improves the energy efficiency for complex NoC based systems.

4. Different 2D network topologies have also been explored for lower energy consumption. In this thesis, we propose a Honeycomb NoC architecture together with the deadlock free routing algorithms. Comparing with Mesh based NoCs, the Honeycomb topology provide lower implementation cost, shorter communication delay as well as higher energy efficiency.

1.4 Thesis Organization

The rest of the thesis is organized as follows. The concept and architecture of NoC are illustrated in Chapter 2. Chapter 3 starts with the introduction of energy efficiency for NoC based systems and then explains the power challenges as well as the abstraction layer model for communication on NoCs. In Chapter 4, we study the 3D NoC architecture which provides benefits in terms of both system performance and power efficiency. However, these benefits do not come for free. Thus we propose a change function method which can quantify the pros and cons. A hierarchical monitoring agent based design approach is introduced in Chapter 5. By using this approach, a NoC based system can autonomously adjust the operations to achieve higher robustness and energy performance. DVFS and power gating techniques are then studied in Chapter 6 with their implementation alternatives on agent based NoCs. A reinforcement learning adaptive routing algorithm is also presented with different case studies. In Chapter 7, we propose a Honeycomb NoC architecture and prove that it outperforms the Mesh based architecture in terms of system performance as well as energy efficiency. Chapter 8 concludes the thesis.

1.5 Research Publications

During the course of conducting research presented in this thesis, the following papers have been published in international journals and proceedings of conferences.

1. Alexander Wei Yin, Liang Guang, Pasi Liljeberg, Pekka Rantala, Jouni Isoaho, Hannu Tenhunen. Hierarchical Agent Based NoC with DVFS Techniques. *International Journal of Design, Analysis and Tools for Integrated Circuits and Systems (IJDATICS)* 1(1), pages 32 - 40, 2011.

2. Thomas Canhao Xu, Alexander Wei Yin, Pasi Liljeberg, Hannu Tenhunen. A Study of 3D Network-on-Chip Design for Data Parallel H.264 Coding. *Microprocessors and Microsystems* 35(7), pages 603 - 612, 2011.

3. Alexander Yin, Nan Chen, Pasi Liljeberg, Hannu Tenhunen. Comparison of Mesh and Honeycomb Network-On-Chip Architectures . In *Proceedings of the 7th IEEE Conference on Industrial Electronics and Applications*, pages 1713 - 1717, Singapore, July 2012.

4. Alexander Wei Yin, Thomas Canhao Xu, Bo Yang, Pasi Liljeberg, Hannu Tenhunen. Change Function of 2D/3D Network-on-Chip. In *Proceedings of the 11th IEEE International Conference on Computer and Information Technology*, pages 181 - 188, Pafos, Cyprus, August 2011.

5. Thomas Canhao Xu, Liang Guang, Alexander Wei Yin, Bo Yang, Pasi Liljeberg, Hannu Tenhunen. An Analysis of Designing 2D/3D Chip Multiprocessor with Different Cache Architecture. In *Proceedings of the 28th IEEE Norchip Conference*, pages 1 - 6, Tampere, Finland, November 2010.

6. Thomas Canhao Xu, Bo Yang, Alexander Wei Yin, Pasi Liljeberg, Hannu Tenhunen. 3D Network-on-Chip with on-chip DRAM: an empirical analysis for future Chip Multiprocessor. In *Proceedings of the*

2010 International Conference on Computer, Electrical, and Systems Science, and Engineering, pages 18 - 24, Kyoto, Japan, October 2010.

7. Alexander Wei Yin, Gangming Lv, Cheng Tao, Pasi Liljeberg, Hannu Tenhunen. RF NoC: A New Paradigm for Very Large Scale Three Dimensional On-Chip Interconnect Networks. In *Proceedings of the 3D Integration Workshop in Design Automation and Test Europe Conference (DATE 2010)*, Dresden, Germany, March 2010.

8. Thomas Canhao Xu, Alexander Wei Yin, Pasi Liljeberg, Hannu Tenhunen. A Study in 3D Network-on-Chip Design for Data Parallel H.264 Coding. In *Proceedings of the 27th IEEE Norchip Conference*, pages 1 - 6, Trondheim, Norway, November 2009.

9. Alexander Wei Yin, Liang Guang, Pasi Liljeberg, Pekka Rantala, Ethiopia Nigussie, Jouni Isoaho, Hannu Tenhunen. Hierarchical Agent Based NoC with Dynamic Online Services. In *Proceedings of the 4th IEEE Conference on Industrial Electronics and Applications (ICIEA 2009)*, pages 434 - 439, Xi'an, China, May 2009.

10. Alexander Wei Yin, Liang Guang, Ethiopia Nigussie, Pasi Liljeberg, Jouni Isoaho, Hannu Tenhunen. Architecture Exploration of Per-Core DVFS for Energy-Constrained On-Chip Networks. In *Proceedings of the 12th Euromicro Conference on Digital System Design (DSD 2009)*, pages 141 - 146, Patras, Greece, August 2009.

11. Alexander Wei Yin, Thomas Canhao Xu, Pasi Liljeberg, Hannu Tenhunen. Explorations of Honeycomb Topologies for Network-on-Chip. In *Proceedings of the 6th IFIP International Conference on Network and Parallel Computing (NPC 2009)*, pages 73 - 79, Gold Coast, Australia, October 2009.

12. Alexander Wei Yin, Liang Guang, Pekka Rantala, Pasi Liljeberg, Jouni Isoaho, Hannu Tenhunen. Hierarchical Agent Monitoring NoCs: A Design Methodology with Scalability and Variability. In *Proceedings of the 26th NorChip Conference*, pages 202 - 207, Tallinn, Estonia, November 2008.

Chapter 2

Network-on-Chip

As the predominating on-chip communication architecture, buses can efficiently connect 3-10 communication partners but do not scale to higher numbers [44]. Even worse, for each connected component, the performance of the bus is unpredictable due to the fact that there are many other components waiting for the bus access. In large scale on-chip systems, it has been proved that designing and verifying the inter-task communication are more challenging than achieving higher performances for computation resources. Moreover, each system has its own communication structure, making reuse difficult.

As the consequence, from year 1999, researchers have started investigating a systematic approach to separate the design of communication infrastructure from that of computation resources such as processors and memory banks. The first scientific article with the term Network-on-Chip (NoC) was then published in November, 2000 [36].

2.1 NoC Overview

Like many other computer networks, NoCs are defined based on key system characteristics including topology, flow control, switching mechanism, routing, etc.

Topology of a network means the connection pattern of communication resources including links, routers, interfaces, etc. Different topologies have different characters and thus will result in different system performances. The most commonly used NoC topology is Two-Dimensional (2D) Mesh network. In depth discussion on NoC topologies and their impacts on system performances will be presented later in Section 2.6.

Figure 2.1 shows an example of NoC with 2D Mesh topology. In this 3×3 NoC based system, the core of the communication infrastructure is the Mesh network consisting of 9 routers. Routers are connected to each other via bi-directional links. Attached to each router is one or several Processing Elements (PEs). The functionalities of PEs can be heterogeneous, such as processor cores, DSP cores, memory banks, specialized I/O blocks including Ethernet or Bluetooth protocol stack implementations, graphics processors, FPGA blocks and even bus based subsystems. Routers and PEs are connected by Network Interfaces (NIs) which provide communication capability between the two parties, based on the chosen communication protocol such as AMBA AXI or OCP.

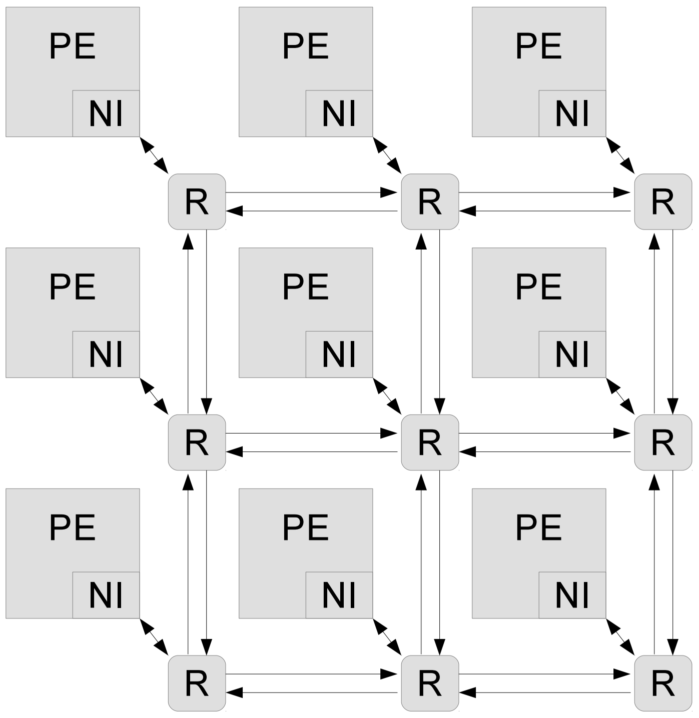


Figure 2.1: An example of NoC with 2D Mesh topology

2.2 Flow Control

Flow control determines the allocation of network resources such as link bandwidth and buffer capacity for data traversing the network. A good flow control method allocates the resources in an efficient way so that the network achieves higher real bandwidth and delivers data with low and predictable latency.

Figure 2.2 shows the units in which flow control is conducted. At the top level is the message, which is a logically continuous group of bits containing the information sent from source node to the destination. However, messages are usually too long to be allocated to network resources. Therefore, in practice, they are divided into one or more packets that have a pre-defined maximum length.

As the basic unit for routing and sequencing, a packet consists of a segment of message to which a packet header is prepended. The packet header includes routing information (RI) and, if needed, a sequence number (SN). A packet can be further divided into flits which are the basic units of bandwidth and storage allocation. Thus, a packet consists of a head flit which allocates link state, one or more body flits which carry the real information, and a tail flit which deallocates the network resources.

Furthermore, a flit can be divided into one or more phits which are the unit of information that is transferred across a link in a single clock cycle. Phits are usually between 1 bit and 64 bits in size, with 8 bits being typical [24].

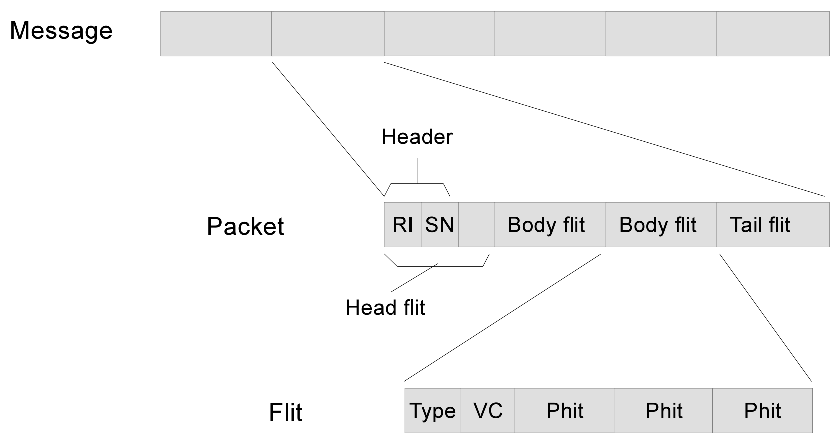


Figure 2.2: Units of resource allocation

In general, there are two categories of flow control mechanisms, namely, bufferless flow control and buffered flow control.

Bufferless flow control is the simplest form of flow control which uses no buffering and simply act to allocate link state and bandwidth to packets. As a result, when there are more than one packets requesting for the same link, the flow control method must perform an arbitration to decide which packet can get access to the link. After the arbitration, the winning packet is sent through the link while the losing packets are either misrouted which will waste network bandwidth by sending packets into wrong directions or even dropped.

To overcome these challenges, more commonly, buffers are added to NoC for more efficient flow control. When a packet loses in the arbitration process, instead of being dropped or misrouted, it will be stored in the buffer while waiting for the link.

2.3 Switching Mechanisms

Switching is the mechanism of how a message traverse its route. There are two main switching mechanisms, namely, circuit switching and packet switching. Circuit switching is a form of bufferless switching that operates by first allocating links to form a circuit from source to destination and then sending one or more packets through this circuit. Four phases are involved during a circuit switching process. The source node firstly sends a request to propagate to the destination and allocate the links. After the circuit is allocated, an acknowledgement is transmitted back to the source. Once the acknowledgement is received, the circuit is successfully established and all packets are sent along the circuit. Finally, when there is no further packet to be sent, a tail flit is sent from source to destination deallocating the reserved links as it passes. For on-chip systems, circuit switching has two major weaknesses comparing to packet switching: high latency and low throughput.

In a packet switching process, each packet is transmitted individually and may even follow different routes to its destination. Once all packets are received by the destination, they are recompiled according to the given order into the original message. Typical packet switching mechanisms include store-and-forward, virtual cut-through and wormhole.

With store-and-forward switching, each router along the communication path waits until a packet has been completely received and stored in the buffer and then forward the packet to the next router. Two resources must be allocated to the packet before it can be sent forward: a packet-sized buffer on the other side of the link, and the exclusive use of the link. While waiting to acquire these resources, no link needs to be held idle and only a packet-sized buffer on the current router is occupied. It is obvious that the store-and-forward switching has the major drawback of very high latency.

Virtual cut-through overcomes the latency penalty of store-and-forward switching by forwarding a packet as soon as the header is received and the needed buffer and link have been allocated. As with store-and-forward switching, virtual cut-through allocates buffers and links in units of packet-size. However, transmission over each hop is started as soon as possible without waiting for the whole packet to be received and stored.

Wormhole switching operates similarly to virtual cut-through, but with link and buffer allocations in the unit of flit rather than packet. When the head flit of a packet arrives at a router, it requires the network resources and can be forwarded immediately along a route. The body flits and tail flit then follow the route allocated to the head flit, while the tail flit releases the resources as it passes. Comparing with virtual cut-through, wormhole switching makes far more efficient usage of buffers which reduces the sizes, costs and power consumption of routers. The main disadvantage of wormhole switching comes from the fact that only the head flit has routing information. In case the head flit is blocked due to resource contention, all trailing flits will also be blocked along the route which may further block other transmissions.

2.4 Routing

Routing is the progress to select a communication path from the source node to the destination in a given network. Topology determines the ideal performance for a given network, whereas routing is one of the key factors which determines how much of the potential can be achieved. A good routing algorithm will lead to more balanced loads on the links across the network, with shorter communication delay. The more balanced the loads are, the closer the network throughput is to ideal.

Based on how the routing path is selected among the set of possible paths between source node and destination, routing algorithms are divided into three main categories, namely deterministic routing, oblivious routing and adaptive routing.

Deterministic routing algorithms always choose the same path between given source node and destination, even if there are more than one possibilities. These routing algorithms take into account neither the routing diversity nor the real-time network situation. As a result, poorly balanced loads on links are to be expected across the network. However, these algorithms are common in practice mainly because they are easy to implement and deadlock-free can be achieved with less efforts. The most commonly used deterministic routing algorithm is the X-Y routing for Mesh topology.

Oblivious routing algorithms, which include deterministic algorithms as a subset, have the performance advantages thanks to the consideration of path diversity. However, the real-time network situation is still not considered when making the routing decision. An example of oblivious routing algorithms is the random algorithm which uniformly distributes packets across all possible paths.

As the most advanced category of routing algorithms, adaptive routing algorithms take the real-time network situation into consideration when making routing decisions. Therefore, they are able to efficiently balance the link loads across the network, and react to the topological changes or faulty nodes during run-time. The considered information may include states of a router or link, lengths of waiting queues, historical routing information and etc.

2.5 Quality of Service

In the field of computer networking and other packet-switched telecommunication networks, the traffic engineering term quality of service (QoS) refers to resource reservation control mechanisms rather than the achieved service quality. Quality of service is the ability to provide different priorities to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow [111]. For example, a required bit rate, delay, jitter, packet dropping probability or bit error rate may be guaranteed.

A network or protocol that supports QoS may agree on a traffic contract with the application software and reserve capacity in the network nodes, for example during a session establishment phase. During this session it may monitor the achieved level of performance, such as the data rate and delay, and dynamically control scheduling priorities in the network nodes. It releases the reserved capacity during the tear down phase [111].

Many problems are prone to occur to packets as they travel from origin to destination, some of them are as follows:

***Dropped Packets* -** Routers may fail to deliver some packets if they arrive when all buffers are already full. Depending on the state of the network, some of, or even all, the packets can be dropped, which is very hard to predict in advance. The receiving application may require these packets to be retransmitted, possibly causing severe delays in the overall transmission.

***Delay* -** It might take a long time for a packet to reach its destination, because it gets held up in long queues, or takes a less direct route to avoid congestion. In some cases, excessive delay can render an application, such as VoIP or on-line gaming, unusable.

***Jitter* -** Packets from the source will reach the destination with different delays. A packet's delay varies with its position in queues of the routers along the path between source and destination and this position can vary unpredictably. This variation in delay is known as jitter and can seriously affect the quality of streaming audio and video.

***Out-of-Order Delivery* -** When a collection of related packets is routed through a network, different packets may take different routes, causing different communication delays. The result is that the packets arrive in a different order than they were sent. This problem requires additional protocols responsible for rearranging out-of-order packets to an isochronous state once they reach their destination.

***Error* -** Sometimes packets are misdirected, or combined together, or corrupted, while being delivered. The receiver has to detect these errors and if any error is found, it should be able to correct it or ask the sender to send again.

In certain interconnection networks, it is useful to divide network traffic into a number of classes to more efficiently manage the allocation of resources to packets. Different classes of packets may have different requirements; some classes are latency-sensitive, while others are not. Some classes can tolerate latency but not jitter [26].

The traffic classes in NoC based systems fall into two broad categories: guaranteed service classes and best effort classes.

Guaranteed service classes guarantee a certain level of performance as long as the traffic they inject complies with a set of restrictions. There is a service contract between the network and the client. Once the client satisfies the restrictions in the service contract, the network will deliver the performance. The client side of the agreement usually restricts the volume of traffic that the client can inject, that is, the maximum offered throughput.

In contract, the network makes weak promises to the best efforts packets. Depending on the network, these packets may have arbitrary delay or even be dropped. As depicted by its name, the network will make its best effort to deliver the packets to their destination.

2.6 NoC Topologies

Interconnect networks are composed of a set of shared routers and channels, and the topology of the network refers to the arrangement of these elements. The topology of an interconnect network is analogous to a roadmap where the channels (like roads) carry packets (like cars) from one router (like intersection) to another [23]. A good topology exploits the characteristics of given network resources, providing high bandwidth while maintaining low communication latency. In this section, we present and study different existing NoC topologies including Ring, Torus, Mesh, Spidergon and Fat-tree.

2.6.1 Ring

A ring network is a network topology in which each router connects to exactly two other routers, forming a single continuous pathway for signals through each router. Figure 2.3 shows a case of eight-router ring network. Comparing with bus-based systems, a ring network does not require a central node to manage the connectivity between routers, and therefore provides better performance under heavy network load. However, since a ring topology provides only one pathway between any two nodes, the entire network may be disrupted by the failure of a single link or router. Moreover, the scalability of ring topologies is limited because removing, adding and changing of a router can affect the whole network. From the energy point of view, the power consumption of routers in Ring topology is expected to be lower than in many other topologies. However, in communication intensive systems, the total energy consumption can be higher due to the potentially long network delays. Moreover, larger buffers are needed to be implemented in the routers in order to store the delayed packets, which will become a major static energy consuming source.

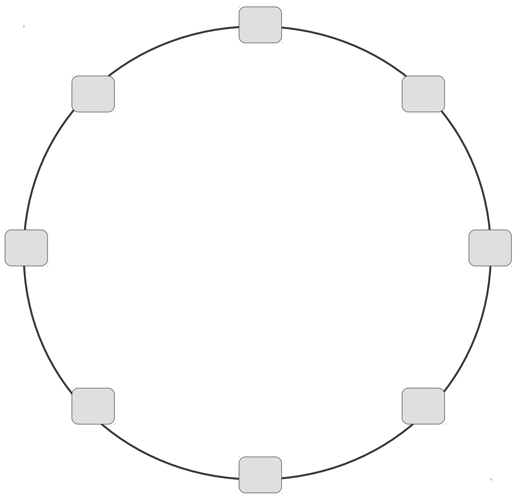


Figure 2.3: An example of Ring topology

2.6.2 Torus

Another well-known symmetric topology is hypercubes which is a k-aray n-cubes network consisting of $N = k^n$ nodes arranged in an n-dimensional cube with k nodes along each dimension. Ring topology can be regarded as a subset of hypercubes where $n = 1$ and k equals the number of nodes in the ring. When $n = 2$, the hypercube becomes a radix-k 2D Torus network where each node is assigned with an n-digit radix-k address $\{a_{k-1}, \ldots, a_0\}$ and is connected by a pair of channels to all nodes with addresses that differ by ± 1 (mod k). The regular physical arrangement in Torus network is well matched to on-chip packaging constraints. The main drawback of this topology is the long wrap-around links which require not only higher implementation cost but also power consumption. Figure 2.4 shows a 3-ary 2-cube Torus network which is also often known as 3×3 Torus.

2.6.3 Mesh

As shown in Figure 2.5, a Mesh network is subset of Torus network without the connecting channels between address a_{k-1} and address a_0 in all directions. Thus Mesh outperforms Torus by lower power consumption and easier implementation and has become the mostly used NoC topology. However, it gives up the edge symmetry of Torus network which can cause load imbalance for many traffic patterns. More detailed studies of Mesh topology

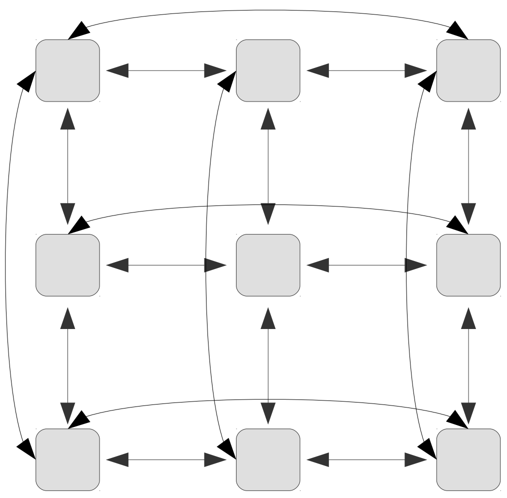


Figure 2.4: An example of Torus topology

will be elaborated and compared with the proposed Honeycomb topology in the later sections.

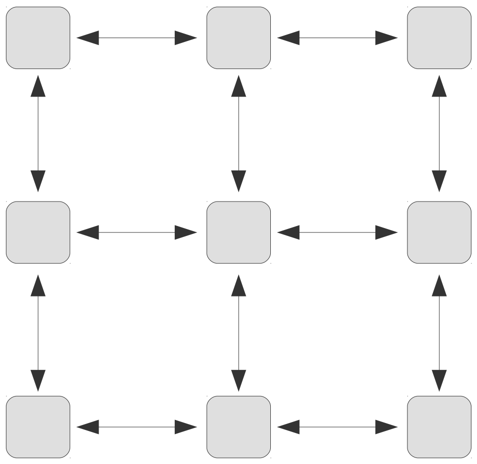


Figure 2.5: An example of Mesh topology

2.6.4 Spidergon

Spidergon is another on-chip communication architecture which was proposed in [47]. As shown in Figure 2.6, a basic Spidergon unit consists of 8 nodes and 12 bidirectional links. The advantages of Spidergon topology include two-hop communication between any pair of nodes, higher aggregate throughput under certain conditions, simpler routing algorithm and less wiring than a crossbar based interconnect. The main constraint of Spidergon topology is its scalability especially when the number of nodes is more than eight which will result in complex physical layout.

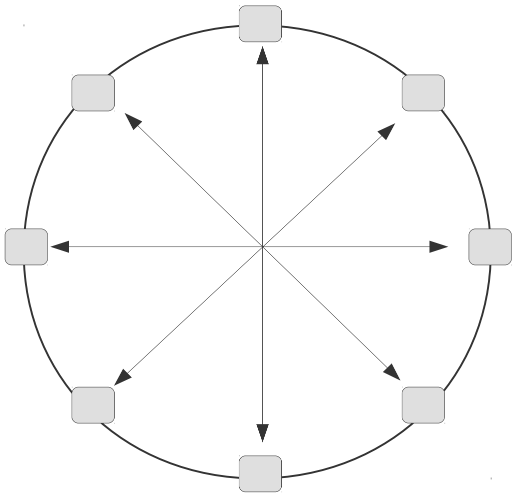


Figure 2.6: An example of Spidergon topology

2.6.5 Fat-tree

The fat-tree network is a universal network topology designed for efficient communication. It has been adopted by on-chip networks as an alternative of Torus-based architectures [2]. In a fat-tree based NoC, as shown in Figure 2.7, several PEs (4 PEs in this example) are connected to a router at the lowest level, which is then connected to the rest of the network through multiple upper links. By judiciously choosing the fatness of links, the network can be tailored to efficiently use any bandwidth made available by packaging and communications technology. The advantage of this topology is that the number of switches is smaller than other topologies. However, routers at higher level are prone to become bottlenecks in communication intensive scenarios which will result in lower energy efficiency and unbalanced heat dissipation.

2.7 Chapter Summary

In this chapter, we introduce the NoC architecture which can solve the challenges that current SoC based systems are facing, as described in the previous chapter. With the maturity of technologies, NoC based systems have gained wide acceptance not only in the academia and research institutes, but also in industries with emerging commercial applications. As the

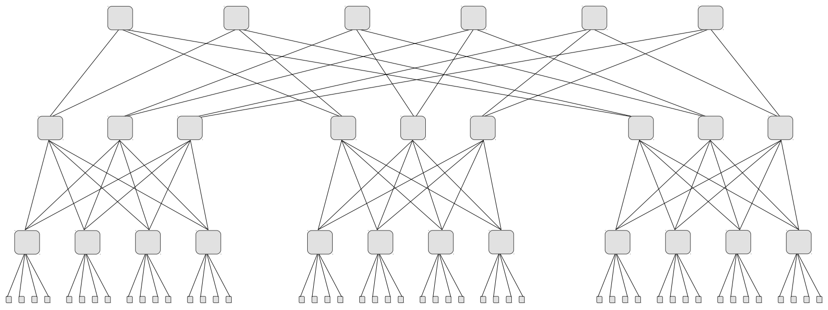


Figure 2.7: An example of fat-tree topology

number of on-chip components grows, NoC is regarded as one of the most promising architecture for future IC designs.

Thereafter, several key system characteristics of NoC architectures are discussed, including flow control, switching mechanism, routing, quality of service as well as topologies. Each of these characteristics has multiple attributes, the choice of which are dependent on the system requirements including performance, power consumption, etc.

Chapter 3

Energy Efficiency in NoCs

NoC is a communication platform specially designed for large scale SoCs with many densely integrated on-chip electrical components. Therefore there are two main reasons making energy efficiency crucially important. First, the biggest market for NoC based systems is the portable devices and thus it will confront the bottleneck of battery capacity and energy limitation. Second, with the shrink of feature sizes, the integration density of on-chip components are getting higher than ever, which makes heat hard to dissipate. Hence, low power consumption and heat dissipation are two of the most important design concerns in NoC based systems.

3.1 Sources of Energy Consumption

In order to improve the energy efficiency, in this section, we firstly examine the sources of energy consumption.

Equation 3.1 shows the main sources of power dissipation in CMOS devices

$$P = \underbrace{(\frac{1}{2}C_L V_{DD}^2 + Q_{SC} V_{DD}) f N}_{P_{Dynamic}} + \underbrace{I_{leak} V_{DD}}_{P_{Leakage}} \qquad (3.1)$$

where V_{DD} is the supply voltage, $P_{Dynamic}$ is the dynamic power consumption caused by changes to the outputs of the CMOS circuit, such as transitions from logic 0 to 1 or vice versa, $P_{Leakage}$ is the static power consumption which is dissipated whenever the circuit is powered on. The leakage power consumption is independent of the switching activity and operating frequency and therefore can be assumed to be constant for a certain circuit.

3.1.1 Dynamic Power Consumption

There are two major components of dynamic power consumption, namely, switching power (P_{SW}) and short circuit power (P_{SC}), as shown in Equation 3.2. Both sources of dynamic power consumption are dependent on the switching activity, i.e. the number of gate output transitions per clock cycle N, and the operating frequency f.

$$P_{Dynamic} = P_{SW} + P_{SC} \tag{3.2}$$

In the following of this chapter, we take an inverter shown in Figure 3.1 as a general model of any CMOS gate.

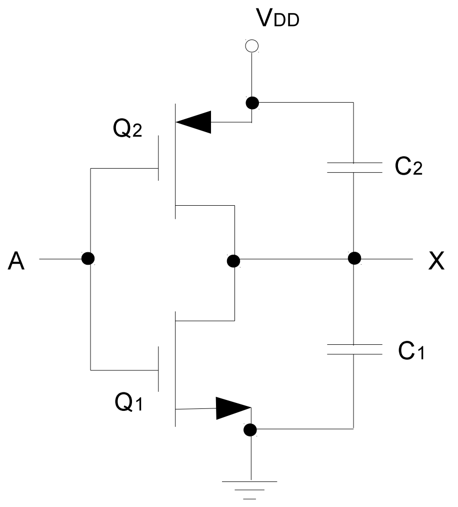


Figure 3.1: An Example of Inverter

3.1.2 Switching Power Consumption

The switching power is dissipated by charging and discharging the nodes capacitance C_L, and therefore it is also referred as capacitive power consumption. For the sake of simplicity, we assume all capacitances are at the output side. As shown in Figure 3.1, the total capacitance C_L consists of C_1 which is the capacitance to ground and C_2 which is the capacitance to V_{DD}. When the input A is at logic 0, the n-channel transistor Q_1 is open and the p-channel transistor Q_2 is closed, leading to a logic 1 at the output X. At the same time, the capacitance C_2 is emptied, shortened by Q_2 and the capacitance C_1 is charged. A change on A from logic 0 to 1 causes Q_1 to close, which will discharge C_1 and Q_2 to open, which will charge C_2. During this single transition, a charge of C_2V_{DD} was taken from the power supply. On the next input change the charge C_1V_{DD} will be taken from the power supply. Assuming that $C_L = C_1 + C_2$, we can say that after one input transition from 0 to 1 and back to 0 the inverter has taken a charge of $(C_1 + C_2)V_{DD}$ from the power supply. Therefore, we can average this for one single output transition to $\frac{1}{2}C_L V_{DD}$. The power dissipated by this is $\frac{1}{2}C_L V_{DD}^2$ which leads to Equation 3.3 for the switching power P_{SW} .

$$P_{SW} = \frac{1}{2} C_L V_{DD}^2 f N \tag{3.3}$$

3.1.3 Short Circuit Power Consumption

During the input transition of a CMOS circuit, there is a period in which both the nMOS and pMOS will conduct, causing a short circuit to flow from power supply to ground, as shown in Figure 3.1 for an inverter without load. This current flows as long as the input voltage V_{in} is higher than a threshold voltage V_T and lower than $(V_{DD} - V_T)$ [100].

The short circuit current is also depending on the gain factor β (amplification) of a MOS transistor and input rise and fall times τ. According to [100], the mean current during a time T (equal to one period of the input signal) can be seen as Equation 3.4.

$$I_{SCmean} = \frac{1}{12} \frac{\beta}{V_{DD}} (V_{DD} - 2V_T)^3 \frac{\tau}{T} \tag{3.4}$$

In CMOS circuit, power is the product of current and supply voltage, and operating frequency $f = \frac{1}{T}$. Therefore, the short circuit power consumption can be calculated by Equation 3.5.

$$P_{SC} = \frac{\beta}{12} (V_{DD} - 2V_T)^3 \tau f \tag{3.5}$$

3.1.4 Leakage Power Consumption

Leakage consumption is the main source of static power consumption. It is caused by the leakage currents of transistors and pn-junctions. In Figure 3.1, when the input signal is logic 0, the n-channel transistor Q_1 is open. The leakage power is determined by the leakage current I_{0n} of Q_1. Likewise, when the input signal is logic 1, the n-channel transistor Q_2 is open. The leakage power is determined by the leakage current I_{0p} of Q_2. Therefore, the leakage power can be calculated by Equation 3.6.

$$P_{Leakage} = P_S = V_{DD} \frac{I_{0n} + I_{0p}}{2} = I_{leak} V_{DD} \tag{3.6}$$

With the development of CMOS technology, the threshold voltage, channel length and gate oxide thickness are reduced from generation to generation. At the same time, however, leakage currents increase accordingly [84].

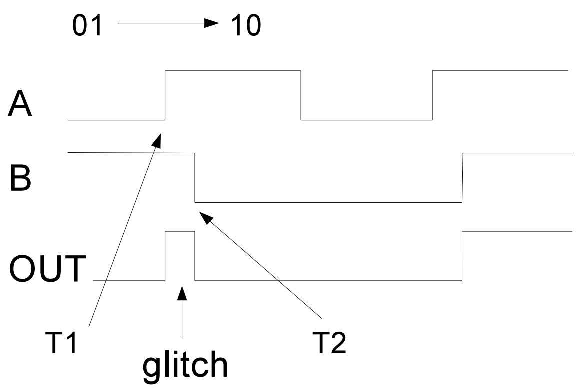


Figure 3.2: An Example of Glitch

Since dynamic power consumptions with more advanced CMOS are lower, it is projected that leakage power will become the dominant part of the overall power consumption [27].

3.1.5 Glitch Power Consumption

Glitches are unwanted transitions of a signal after an input change until the final output value is reached. This behaviour is caused by having input that are not switching simultaneously, called logic hazard. Glitches occur when the output node temporarily has a value that is not the steady-state value.

Figure 3.2 shows an example of glitch in an "AND" gate $OUT = AB$. When input signals come, they arrive at the "AND" gate at different time stamps, i.e., signal A arrive at time stamp T1 while signal B at time stamp T2. Therefore, when input pattern changes from "01" to "10", an unexpected behaviour happens, which is the logic 1 at the gate output between time stamps T1 and T2.

In CMOS circuits, glitches can cause significant dynamic power consumption since they represent at least two binary transitions per clock period.

3.2 Power vs. Energy

Before beginning the discussion of energy efficiency in NoCs, it is worthwhile firstly to review the difference between two concepts, namely, power and energy.

Energy has units of Joules and can be related to the amount of work done or electrical resources expended to perform a computation. Power, on the other hand, is a measure of the rate at which energy is consumed per unit time, and is typically expressed in units of Watts, or Joules/second [10]. Equation 3.7 shows the relationship between these two concepts.

$$Power = Energy/Time \tag{3.7}$$

3.2.1 Low Power Systems

Reducing power consumption is obviously an important aspect in optimizing energy efficiency, which keeps the design of low power systems at the top on the list of design issues for researchers in both academia and industry. Some main stream power optimization methods are described as follows [95] while more innovotive solutions are to be explored.

Using digital circuits whenever possible **-** During the 1960s, analog circuits were preferred whenever possible in order to avoid the large standby power drain of bipolar digital circuit [68]. With the development of CMOS technology, the static power consumption of digital circuits was greatly reduced, and therefore, digital CMOS circuits have received the primary focus in low power circuit design.

Use the lowest possible supply voltage for digital circuit **-** As shown in Equation 3.3, switching power consumptions are heavily dependent on supply voltage V_{DD} and operating frequency f. Order of magnitude power reduction has been achieved by scaling the both V_{DD} and f. Therefore, for certain applications such as wireless sensor networks or radio frequency identifiers, this method offers an ideal solution which prevents power waste by providing "just enough" computing capabilities.

Use the smallest possible feature size **-** As can be seen from the previous section, several sources of power consumption are dependent on capacitances. Since capacitance decreases with decreasing feature size, improvements in IC fabrication technology yield more efficient power dissipation. As device geometries continue to shrink, power consumption will be further optimized.

Optimize circuit architectures **-** To achieve higher system performance, especially the computing speed, different architectural techniques have been proposed which trade computing speed with area costs. Parallel processing and pipelining, as two well-known techniques, can be utilized to lower power consumptions by reducing the supply voltages and operating frequencies without majorly affect system performances.

Parallel processing uses duplicated hardware components to achieve better circuit performance, thus also allows power reduction by lowering supply voltages and operating frequencies. Considering a datapath with supply voltage V_{DD}, frequency f and capacitance C, the power consumption is then $P_0 = CfV_{DD}^2$. If the operation is parallelized to N identical modules, each operates at the frequency of $\frac{f}{N}$, the supply voltage can be reduced accordingly since the speed requirement for each module is reduced. Assume the new voltage is αV_{DD} where α is the voltage reduction factor. Theoretically, the total capacitance increases also by a factor of N, to NC. Therefore, the new power consumption can be calculated by $P = (NC)\frac{f}{N}(\alpha V_{DD}^2) = \alpha^2 P_0$.

Pipelining is another technique in which a synchronous digital circuit is constructed by partitioning the combinational logic into stages and inserting a set of memory elements, generally called an internal register, or simply a register or synchronizer, to provide temporary storage for the computed data between successive pipeline stages [14]. By doing so, the delay of the critical path can be reduced at the cost of longer latency introduced by the registers. Smaller delay allows reduced supply voltage and therefore power consumption can be lowered.

Very often, parallel processing and pipelining are combined in a circuit to multiply the performance and power benefits.

Exploit low power circuit techniques **-** Standard techniques include dynamic voltage and frequency scaling, clock gating, power gating, glitch reduction, intelligent selection of algorithms and numeric representations, multi-threshold voltages, etc. These techniques require changes and optimizations of both hardware and software. Designs at all levels of an electronic system are involved including circuit level, architecture level, algorithm level and even the application program which is run on top of the system.

Use dedicated hardware **-** Dedicated hardware always achieves lower power consumption than programmable circuits. Consequently, an optimal low-power design provides only as much programmability as absolutely necessary. Moreover, dedicated hardware can be optimized for the software system. An example is Apple's laptop which have much longer battery life than other laptops. One of the main reason is that it is dedicated to Apple's operating system and thus has optimized power consumption solution.

3.2.2 Energy Efficiency

As depicted in Equation 3.7, energy is the product of power and time. Therefore, low power consumption of electronic systems can not be always translated into energy efficiency.

When supply voltages and operating frequencies are reduced, the power consumption of an electronic system is surely lowered according to the equations shown in the previous subsection. However, as speed is reduced approximately by the factor of $(V_{DD} - V_T)$ [93], the duration to complete a task is expected to increase, and therefore there is potential risk of higher energy consumption. A major contributor is the leakage power consumption which is projected to become a dominating energy consumer in the future.

Therefore, energy efficiency is a balanced factor between power and performance, especially system speed. We introduce the concept of power performance product to reflect this factor, as will be shown in the following chapters.

3.3 Power Challenges of NoC based Systems

As it is in all embedded systems, power consumption remains as one of the key challenges in NoC based systems, where power is consumed by processing elements, communication infrastructure and leakage current.

3.3.1 Processing Element

Processing elements are made of logic and registers which are usually implemented as digital CMOS circuits. Therefore, the power consumption sources listed in Section 3.1 also applies for NoC based systems.

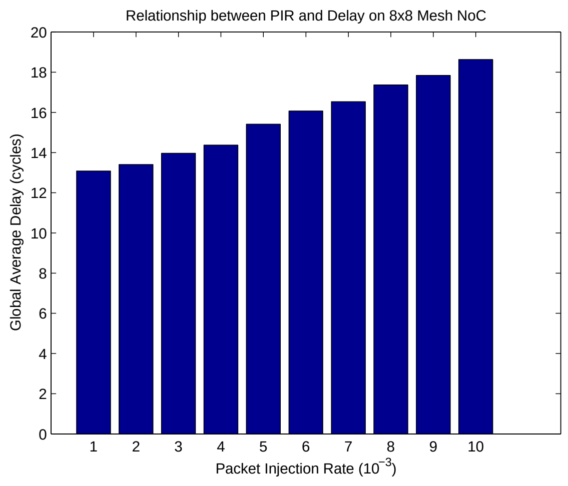
Furthermore, NoC is especially designed for electronic systems where a large number of processing elements, such as processors, memory banks, blue-tooth, etc., are integrated on a single chip. It is obvious that the total power consumption of the chip will be higher, making both battery life and heat dissipation more challenging design concerns.

Besides, one of the main advantages of NoC based systems is its capability of reusing processing elements by defining network interfaces. However, this admirable feature does not come for free. Reusing the pre-defined IP blocks always means that we use something more general, thus less optimal, than necessary for a particular task [44]. Consequently, the processing elements in NoC based systems may consume more power comparing with dedicatedly designed circuits.

Apart from the methods mentioned in Section 3.2.1, one effective way to reduce NoC power consumption is to reduce the amount of data sent over the network which can be achieved by at least two methods at the processor level. The first method is to optimize the mapping of tasks as well as processing elements. In a tile based NoC, a tile can consist of multiple processing elements. By implementing most frequently communicating components in the same tile, less data will be transmitted to the communication network. The other commonly used method is data compression. During the recent years, researchers have proposed several effective ways of data compression at the cache and network levels [49].

3.3.2 Communication

In SoCs, interconnect wires account for a significant fraction, which can be as high as 50%, of the total power consumption [58]. As the technology scales into the nanometer regime, it is projected that the delay and energy



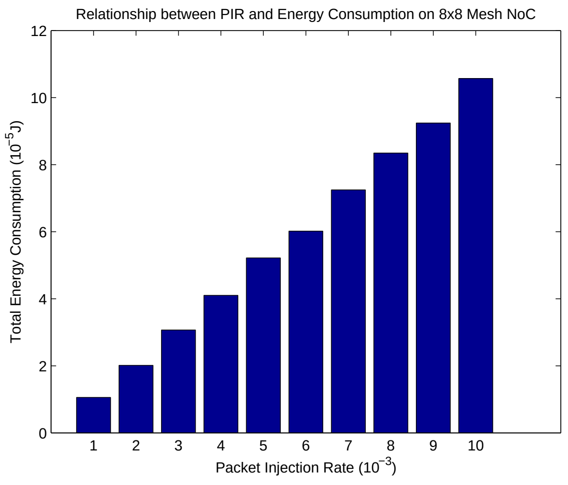


Figure 3.3: Relationships between delay, energy and PIR

consumption of global interconnect structures will consume larger shares of power [81].

Comparing with bus-based SoCs, NoC based systems have higher power efficiency due to the avoidance of long global wires. However, the interconnection network remains as a major power consumer out of all the power contributors. For the Alpha 21364 processor with distributed memory, the integrated switches and link circuits consume 23W out of the total 125W chip power [87]. On the 16-tile RAW processor, the interconnection network accounts for as high as 35% of the total chip power [51].

To illustrate the different reactions of communication delay and consumed energy against changes in the number of packets in the network, we perform a network level simulation using Noxim with different packet injection rates (PIRs), starting from 0.001 to 0.01. The PIR is based on memory less Poisson distribution, and the network is of 8x8 Mesh topology.

Figure 3.3 shows the result of this simulation. It is obvious that under this non-congested network environment, the consumed energy grows much faster than the communication delay with the increasing PIR.

3.3.3 Leakage Power Consumption

As technology scales to deep sub-micron processes, leakage power becomes increasingly significant as compared to dynamic power. In NoCs, there are two main leakage power consumers, namely, on-chip routers and links. According to the simulation results shown in [15], links only consume approximately 5% of the leakage power. This is due to the fact that wires do not dissipate leakage power, and thus the leakage consumption of just the drivers is minimal, comparing with routers. However, in NoCs with more long or global wirings, the inserted repeaters will consume a large proportion of leakage power, especially with smaller feature sizes. Thus, the avoidance of global wiring is desirable in order to achieve high energy efficiency.

In routers, the major leakage power consumers are buffers, crossbars and arbiters, out of which buffers consume over 64% of the leakage energy [15]. Therefore, to reduce the leakage power consumption, optimization of buffers is of crucial importance.

3.4 Power Models

To optimize the energy efficiency in NoCs, it is worthwhile to investigate into the communication layers and find the power sources in each phase.

3.4.1 OSI 7-Layer Model

Comparing with the conventional bus based on-chip systems, NoCs enable the inter-component communication to be simpler and more predictable, by leveraging the existing technologies of computer networks. Therefore, before introducing the power consumers in NoCs, we firstly present the standard abstract layers in communication systems, namely, Open Systems Interconnection model (OSI model).

There are seven abstract layers in the OSI model performing different roles in data communication. Therefore, this model is also known as OSI 7-layer model. The roles of the layers are described as follows.

Application Layer - This layer supports application and end-user processes. Communication partners are identified, quality of service is identified, user authentication and privacy are considered, and any constraints on data syntax are identified. Everything at this layer is application-specific.

This layer provides application services for file transfers, e-mail, and other network software services. Telnet and FTP are applications that exist entirely in the application level.

***Presentation Layer* -** This layer provides independence from differences in data representation (e.g., encryption) by translating from application to network format, and vice versa. The presentation layer works to transform data into the form that the application layer can accept. This layer formats and encrypts data to be sent across a network, providing freedom from compatibility problems. It is sometimes called the syntax layer.

***Session Layer* -** This layer establishes, manages and terminates connections between applications. The session layer sets up, coordinates, and terminates conversations, exchanges, and dialogues between the applications at each end. It deals with session and connection coordination.

***Transport Layer* -** Usually TCP (the top half of TCP/IP). This layer provides transparent transfer of data between end systems, or hosts, and is responsible for end-to-end error recovery and flow control. It ensures complete data transfer.

***Network Layer* -** Typically IP (the bottom half of TCP/IP). This layer provides switching and routing technologies, creating logical paths, known as virtual circuits, for transmitting data from node to node. Routing and forwarding are functions of this layer, as well as addressing, internetworking, error handling, congestion control and packet sequencing.

***Data Link Layer* -** Ethernet, ATM, Frame Relay, etc. At this layer, data packets are encoded and decoded into bits. It furnishes transmission protocol knowledge and management and handles errors in the physical layer, flow control and frame synchronization. The data link layer is divided into two sub-layers: The Media Access Control (MAC) layer and the Logical Link Control (LLC) layer. The MAC sub-layer controls how a computer on the network gains access to the data and permission to transmit it. The LLC layer controls frame synchronization, flow control and error checking.

***Physical Layer* -** This layer conveys the bit stream - electrical impulse, light or radio signal – through the network at the electrical and mechanical level. It provides the hardware means of sending and receiving data on a carrier, including defining cables, cards and physical aspects. Fast Ethernet, RS232, and ATM are protocols with physical layer components.

Figure 3.4 shows a scenario where a sender transmits data to a receiver via communication network. The data firstly go through the seven layers from the sender's side until it reaches the physical communication links which transfer data bit by bit. The data will then be transmitted upwards via the seven layers at the receiver's end until reaching final destination. In a typical communication system, energy will be consumed at every layer during the communication process.

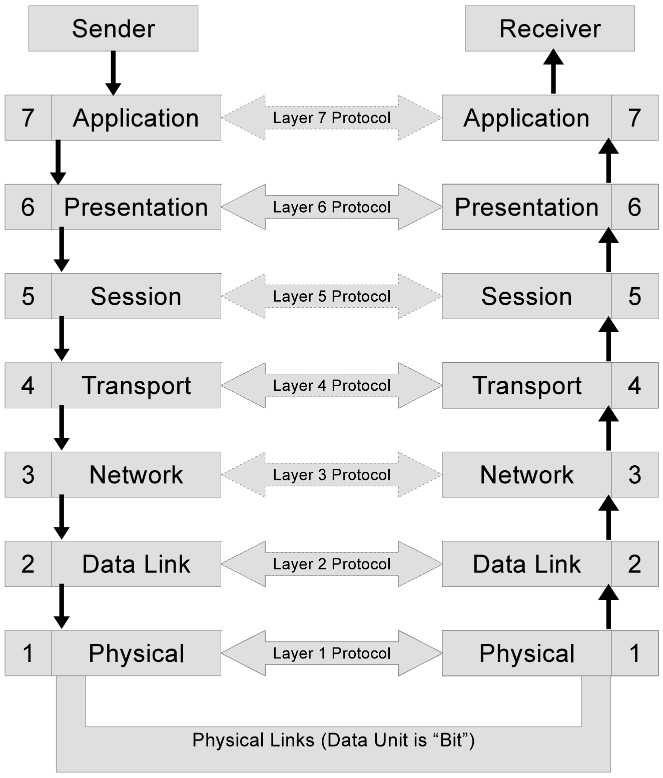


Figure 3.4: Communication Conforming to OSI 7-Layer Model

3.4.2 Abstraction Layers in NoC Communication

Inspired by the OSI 7-layer model, NoC researchers have proposed similar layered stacks. Instead of seven layers, NoC design focuses mostly on five layers, namely, application layer, transport layer, network layer, data link layer and physical layer [45], [111].

Application Layer

For on chip communication, the functionalities of the three highest layers in OSI model can be merged into this layer which comprises both operating system and application software. One important feature of operating system is dynamic power management which enables users (people or application software) to control the power consumption via power management API.

Moreover, task mapping is performed at this layer for NoC based systems. Applications to be executed are often modelled as communication task graphs which show the interactions and dependencies among tasks. There are two types of dependencies, namely, control dependencies and data dependencies. Control dependencies mean that one task has to wait another one to be finished first, while data dependencies indicate that one task has to receive data from another one in order to be executed. The right side of Figure 3.5 shows an example of communication task graph.

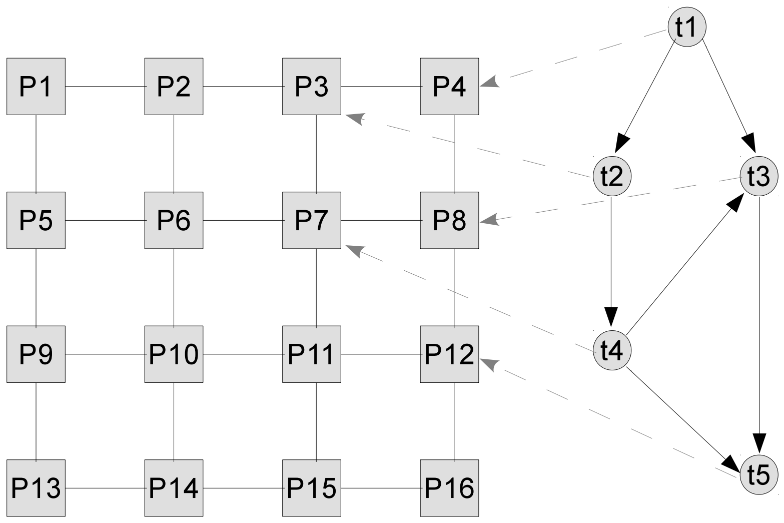


Figure 3.5: An Example of Task Mapping on NoC

After having the communication task graphs, tasks need to be assigned to the PEs in order to be executed. The process of assigning and ordering tasks and communication transactions to PEs and network resources is called mapping. Mapping plays an important role in energy optimization since it can affect the communication distance, delay and even the total execution during of an application. Figure 3.5 illustrates the process of mapping tasks to PEs on a NoC.

Transport Layer

In NoCs, the transport layer handles the establishment of communication and delivery of messages using lower level layer. It also provides the traffic control, i.e. the load of the network is detected and overload of the network is avoided. By detecting and avoiding congestions, communication delay can be reduced and thus the related energy consumption is optimized.

Another important role of this layer is decomposition of messages into packets at the source, and reassembly at the destination. In NoC based systems, packets must be ordered in sequence before they can be reassembled. The ordering of packets is associated with power cost. Another issue is the packet size. Packet energy per hop increases with packet size, but at the same time storage energy is reduced. The optimized packet size can be found using system simulation at design time.

Network Layer

In NoCs, two of the most important power related issues are network architectures and routing algorithms.

Different network architectures have different characteristics in terms of delay, throughput, power consumption, etc. Router architectures can be also different which largely affect the network power consumption since routers consume a great proportion of switching power as well as leakage power. The impacts of network architectures will be further elaborated in Chapter 7.

As one of the most important design issue for NoCs, routing algorithms have significant effects on power consumption. A suitable routing algorithm can not only reduce the communication delay which lower the consumed energy but also avoid traffic congestion which alleviate the heat dissipation challenge.

Figure 3.6 shows an example of how routing algorithms can affect the power consumption. Noxim is used to perform six sets of simulation on a 8x8 Mesh network. We use different routing algorithms, namely, X-Y Routing, West First Routing, North Last Routing, Negative First Routing, Odd-Even Routing and Fully Adaptive Routing, with the Poisson packet injection rate of 0.01, in this experiment. The details of the routing algorithms and parameter configurations are illustrated in [74].

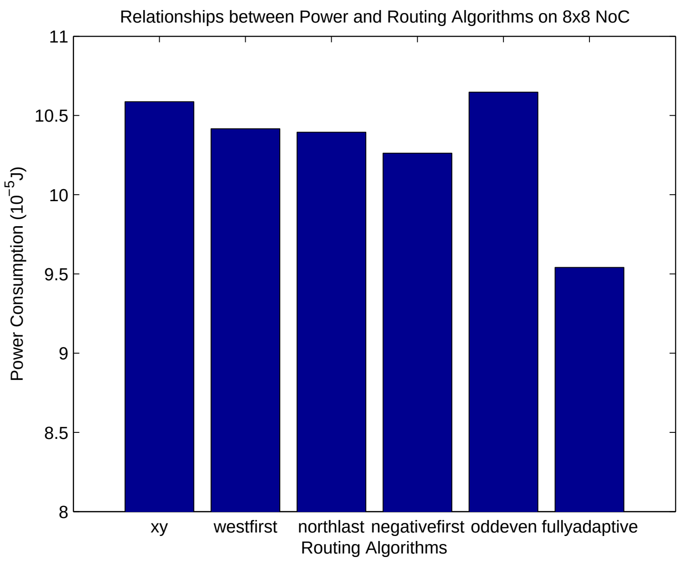


Figure 3.6: Relationships between Power and Routing Algorithms

Data Link Layer

In NoCs, the purpose of this layer is to provide reliable information transfer across the physical link. The layer has the functionality of transferring one word of information from one node to a connected node without error. Since the two connected units may work asynchronously, the data link layer also has to take care of the hardware synchronization. The last responsibility of the data link layer is the data encoding or data rate management for controlling power consumption, etc.

Physical Layer

This layer is concerned with physical properties of the physical medium used for connecting switches and resources with each other. In a NoC system, it specifies voltage levels, length and width of wires, signal timings, number of wires connecting two units, etc.

As at the data link layer, power consumption mechanisms on the physical layer in NoCs are similar to those in conventional ICs. However, in NoCs, the average lengths of links are expected to be shorter due to the avoidance of long wires in NoC's nature.

3.5 Chapter Summary

In this chapter, we introduced the basics of energy efficiency in on-chip systems, especially NoC based systems.

To better understand the concept, at the beginning of this chapter, we explain the sources of power consumptions in digital circuits, as well as the differences between power and energy.

Power challenges for NoC based systems are presented for processing elements, communication and leakage consumption. Among these factors, in this thesis, we put more emphases on the switching power consumed during communication process and the leakage power consumption.

At the end of this chapter, we present the OSI 7-layer model which serves as the fundamental of communication layers in NoCs. Then we discuss the abstract layers in NoCs together with the power sources and optimization techniques at each layer.

Chapter 4

3D Computing Platform

Within the last 10 years it has been clear that Moore's law can not be kept by using only transistor scaling technologies. Therefore, 3D IC has become an essential technology to design and construct more advanced microelectronic systems. Comparing with the System in Package technique where different chips are stacked vertically using off-chip signalling, 3D IC provides a single chip solution where the communication is achieved with only on-chip links, both horizontally and vertically. The energy consumption in a 3D IC is much lower than its 2D counterpart, thanks to the fact that off-chip communication and long wires are less utilized when components are integrated more densely.

4.1 Development of Three Dimensional Circuits

The concept of 3D ICs was firstly introduced in the 1980s when scientists projected that significant reductions in signal delay and power consumption could be achieved by stacking multiple layers of electronic circuits vertically [4]. A 3D IC is a chip where multiple layers of thinned-active 2D ICs are stacked, bonded, and electrically connected with vertical communication channels through silicon and oxide layers.

Among different communication schemes, Through-Silicon Vias (TSVs), which are the inter-layer connections, is widely regarded as the most appropriate communication solution in the 3D IC technology. A TSV is formed by aligning, defining, and etching a cavity between two layers to expose an electrode in the lower layer; lining the side-walls of the cavity with an insulator; and filling the cavity with metal or doped polysilicon to complete the connection [75].

Figure 4.1 (a) illustrates an expanded view where a 3D IC consists of 2D ICs that are thinned, bonded together, and interconnected with TSVs distributed within the planes of the 2D ICs. Thereafter, a cross-section of a

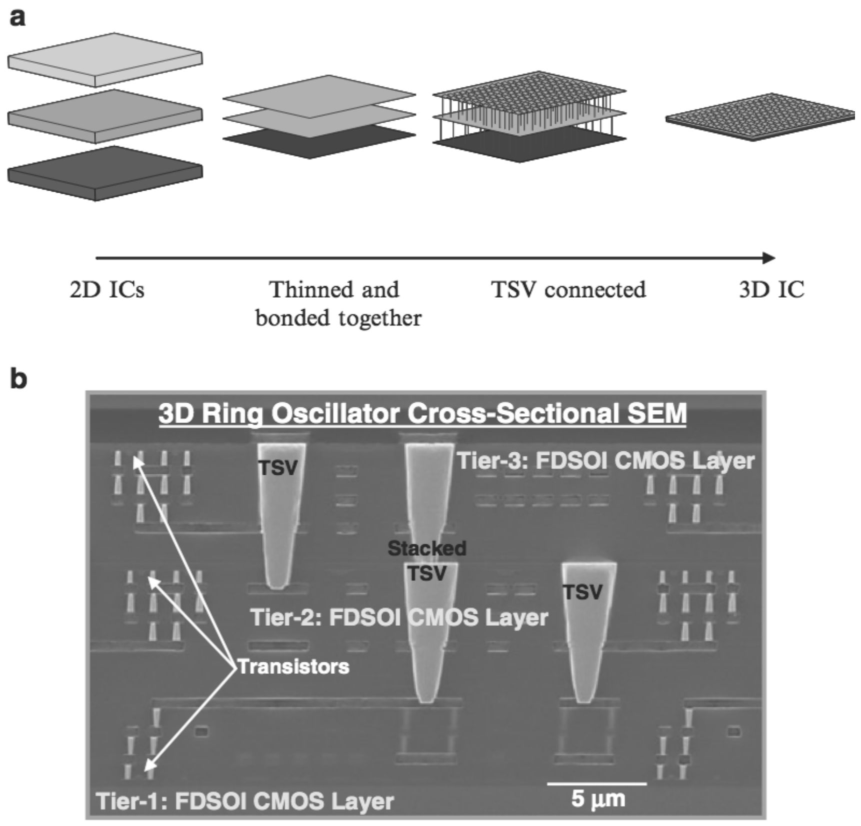


Figure 4.1: A symbolic illustration of a 3D IC [75]

3D ring oscillator built with a fully depleted silicon-on-oxide technology is presented in Figure 4.1 (b).

4.1.1 Stacking Approaches

When stacking vertical layers to form a 3D IC, there are mainly three different technologies, namely Die-to-Die (D2D) stacking [61], Die-to-Wafer (D2W) stacking and Wafer-to-Wafer (W2W) stacking.

D2D stacking is a chip level integration approach where components are built on multiple dies, which are then aligned and bonded. Vertical connections in D2D stacking can be achieved using wire bonding as shown in Figure 4.2 or TSVs as shown in Figure 4.3. One advantage of D2D stacking is that each component die can be tested first, so that one bad die does not ruin an entire stack. However, D2D stacking suffers from the highest cost among the three approaches.

For wafer level stacking including W2W and D2W, the most commonly used vertical interconnects are TSVs. In W2W stacking, two or more complete wafers are stacked together and dies are extracted after the assembling. This approach provides the highest process throughput. However, the extracted dies from different wafers must have the same size which is hardly the case in reality especially for heterogeneous integrations. Besides, the assembly is "blind" since chips cannot be tested before being assembled.

In D2W stacking, components are firstly implemented on two wafers. A die is then singulated from one wafer and placed on top of another die which is still a part of the other wafer. Comparing with W2W, D2W stacking allows testing dies individually before 3D assembly. This is also known as the Known-Good-Die (KGD) assembly which provides better yield. Furthermore, it supports different chip sizes on different layers with heterogeneous technologies.

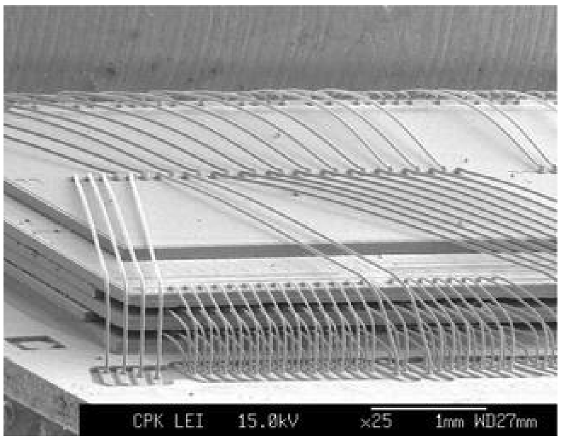


Figure 4.2: 3D stacking using wire bonding [82]

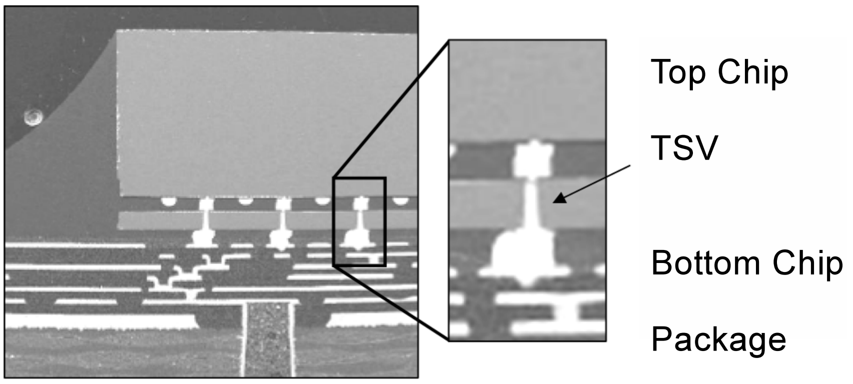


Figure 4.3: 3D stacking using TSVs [12]

Due to the manufacturing cost consideration, W2W and D2W stacking approaches are more commonly used in the current 3D integration processes.

Table 4.1: Comparison of W2W and D2W stacking approaches [88]

	Wafer-to-Wafer	Die-to-Wafer
Wafer and die size	Wafer and die must have the same size	Different sizes are supported
Throughput	Wafer scale	Die scale
Yield	Lower than the lowest yield wafer	KGD can be used to improve yield
Alignment accuracy	$< 2\mu m$ global alignment	$\approx 10\mu m$ for $> 1000dph$, $< 2\mu m$ for $< 100dph$

A comparison of these two techniques are shown in Table 4.1.

As shown in Figure 4.4, apart from cost and testing considerations, the choice of W2W or D2W also depends on two other key requirements of chip size and alignment accuracy. When high precision alignment is required in order to achieve high density vertical integrations, W2W is preferred by performing a wafer level alignment while having an acceptable throughput. W2W is also more appreciated when chip sizes get smaller [88].

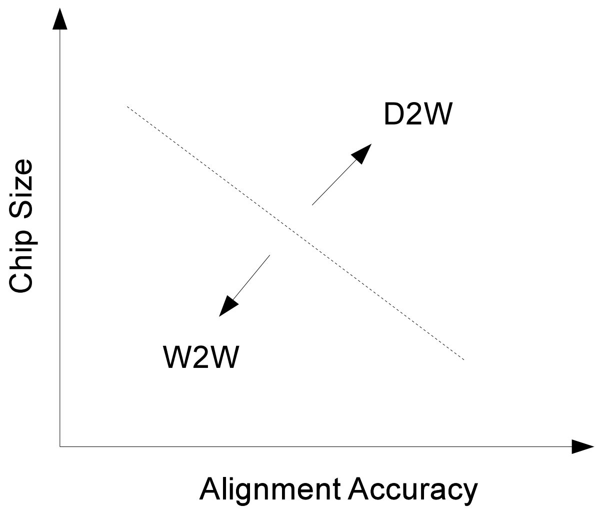


Figure 4.4: The choice between W2W and D2W stacking approaches

4.2 Survey of 3D Architectures

Like in many other vertically stacked on-chip systems, different layers in 3D NoCs can be heterogeneous in terms of layout, functionalities, technology processes, digital/analog, etc. Different layers are connected together by communication channels which are implemented onto the routers.

4.2.1 Single Layer Circuit

Before introducing different 3D IC architectures, it is worthwhile to examine their predecessor which is still the major system component, namely, single layer circuit. In this section, we focus on the router architecture as well as crossbars for the later studies and comparisons.

Figure 4.5 illustrates the generic router architecture for 2D Mesh NoCs. The core of the router is a 5×5 crossbar which connects input and output ports in five different directions, namely, Local, South, East, West and North. A pre-determined number of Virtual Channels (VCs) are implemented at the incoming ports of the router. In addition, a Virtual channel Allocation unit (VA) is needed to arbitrate between all packets competing for the same VC and choose a winner. The Switch Allocation unit (SA) then arbitrates between all VCs requesting to access the crossbar. Thereafter, the winning packet can traverse the crossbar and move to the corresponding output direction according to the routing decision made by the Routing Computation unit (RC).

4.2.2 3D NoC-Bus Hybrid Architecture

The most intuitive implementation of 3D NoC is via the hybrid NoC-Bus architecture. Among different on-chip communication channels, bus architecture is one of the most mature technology with relatively low cost. This method was firstly introduced in [55] where it was used in a 3D UNCA L2 Cache for CMPs.

Figure 4.6 illustrates the architectural view of 3D NoC-bus hybrid model as well as the crossbar structure. Routers on the same layer are connected via conventional 2D NoC, while routers on different vertical layers are connected via a number of buses. For each router, there are dedicated input and output ports to the relevant vertical bus through which flits can be transferred to other layers.

Despite the maturity of bus architecture, 3D NoC-bus hybrid model provides also lower area overhead for routers, as shown in Table 4.2. However, since bus is a shared medium, it can only be used by a single flit at any given time. As a result, this model suffers from a major drawback which is that it does not allow concurrent communication in the third dimension.

4.2.3 Symmetric 3D NoC Architecture

Another 3D architecture is symmetric 3D NoC in which both intra- and inter-layer data communication bear the same characteristic, namely, hop-

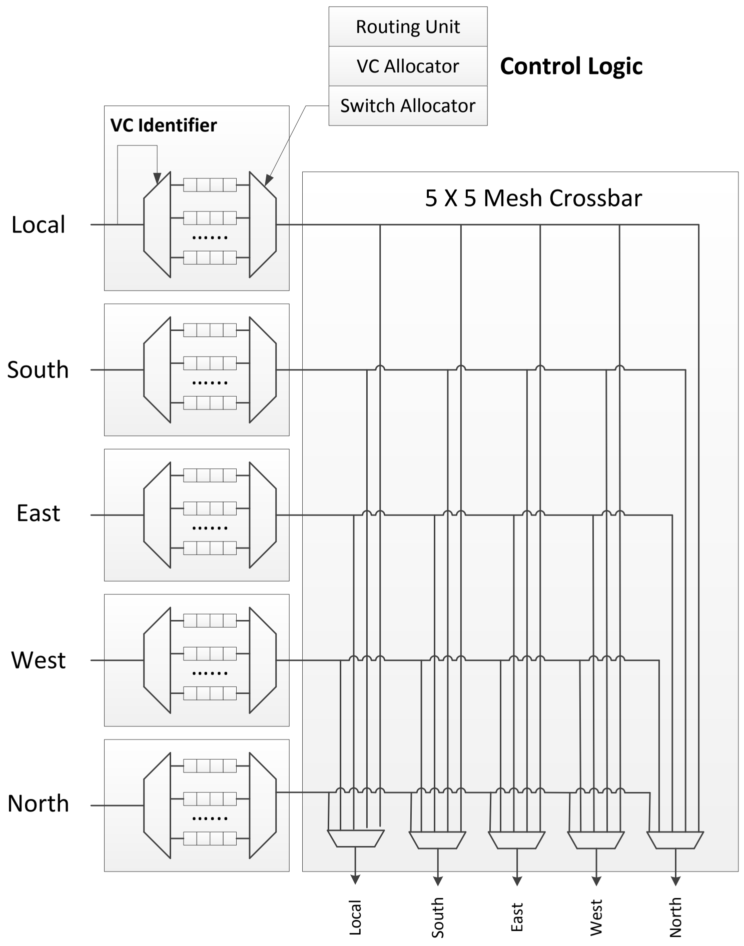


Figure 4.5: Router architecture for single layered Mesh NoCs

by-hop traversal, as shown in Figure 4.7. In this model, two additional physical ports are added to each router to provide communication to upper and lower vertical layers. In addition, hardware modules such as the associated buffers, arbiters (VC arbiters and switch arbiters) and crossbar extension are also integrated into the routers.

The symmetric 3D NoC architecture outperforms NoC-Bus hybrid model by its higher scalability for communication among different vertical layers. However, it suffers from two main drawbacks. Firstly, it takes more than one clock cycle to transfer a flit from one layer to an non-neighbouring layer. When there are a large number of vertical layers, it will take a correspondingly large number of hops for a flit travelling. Furthermore, each flit must undergo buffering and arbitration at every hop, adding more overall delay and power cost when moving upward or downward. Secondly, the sizes of crossbars scale upward quickly with more input and output ports. In this

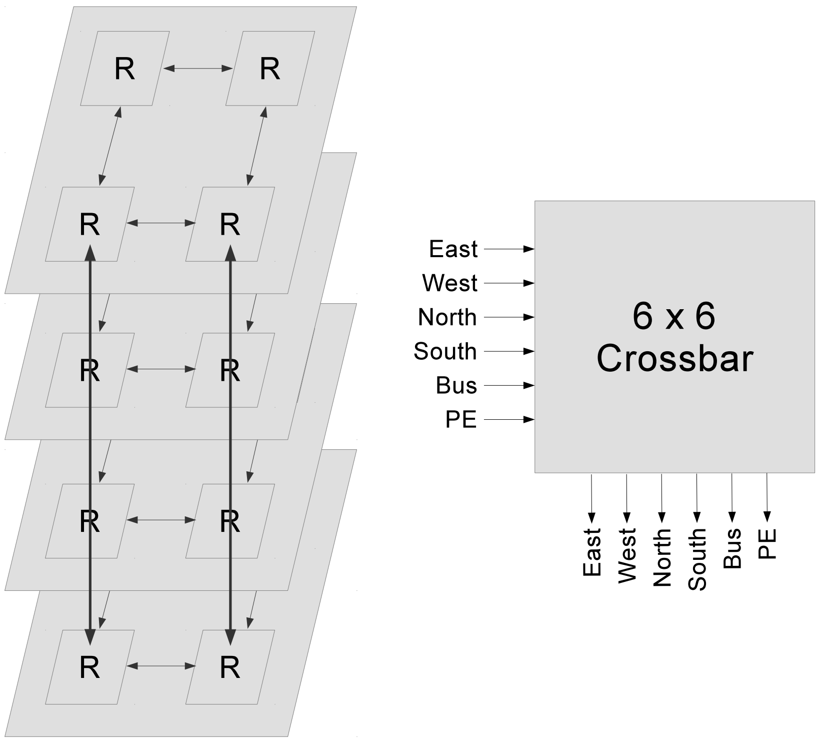


Figure 4.6: 3D NoC-bus hybrid architecture

architecture, each router has a 7×7 crossbar which incurs higher area and power consumptions. The comparison with other architectures is shown in Table 4.2.

4.3 Change Function of 2D and 3D NoCs

The supplier-centric technology world is in crisis. The solution is to switch the focus to a user-oriented model. When deciding whether to adopt a new technology, users always have their weighing machines of perceived crisis versus the pain of adoption. If the crisis is less than the perceived pain of adoption, there will and should be no change. On the contrary, if the crisis is greater than the total perceived pain of adoption, changes will and should occur [21].

4.3.1 Overview

In the highly competitive IC industry, it is more than ever important to think from users' point of view before making major R&D decisions. Very often, without rigorous consideration beforehand of the pain of adoption for more

Table 4.2: Area and power comparisons of 2D NoC, 3D NoC-bus and symmetric 3D NoC architectures (with 90 nm technology and 50% switching activity) [88]

Router Type	Area Consumption	Power Consumption
2D NoC	211705 μm^2	56.9668 mW
3D NoC-Bus Hybrid Model	284690 μm^2	76.3205 mW
Symmetric 3D NoC	367890 μm^2	94.0159 mW

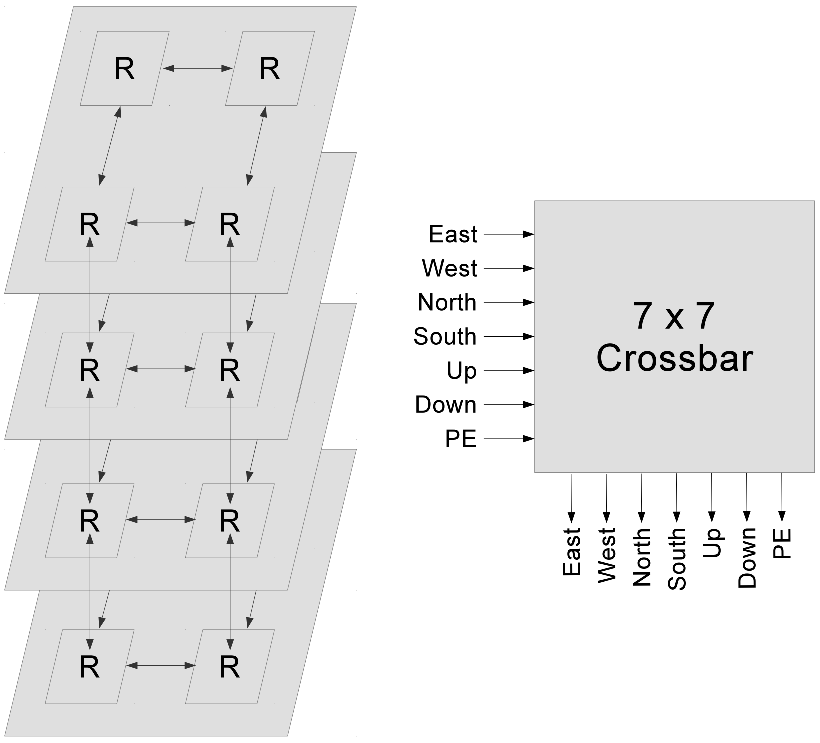


Figure 4.7: Symmetric 3D NoC architecture

advanced technologies, researchers find out that their brilliant inventions have been abandoned by companies, or companies find out that their heavily invested products have not turned into successes. In this thesis, we apply the idea of change function into the area of NoC and discuss the choice of 2D or 3D NoCs under certain requirements and situations. By using the change function, decision makers or stakeholders will have a clear understanding whether their expectations, especially business expectations, can be met by changing the 2D NoC into 3D systems. The proposed change function for

NoCs is shown in Equation 4.1

$$\epsilon = \sum \omega_i \delta_i - \sum \omega_j \zeta_j \tag{4.1}$$

where δ_i is one of the advantages of 3D NoCs over traditional 2D NoCs, ζ_j is one of the pains, ω is the weight of the corresponding factor which is decided by business managers, and ϵ is the changing factor. When considering whether to implement NoCs in the 3D manner, designers should have a clear view of their expectations T_{exp} as the changing threshold. 3D NoCs should be used only in situations where $\epsilon > T_{exp}$. To make different parameters comparable, the advantages and pains discussed in this thesis are evaluated as the proportion of improvement or deterioration comparing with the 2D designs.

4.3.2 System Performance

Throughput is one of the most important parameters when analysing different network topologies. It is a metric that quantifies the rate in which messages can be sent across a communication network. The definition of throughput is the average number of flits arriving per PE per clock cycle, and thus the maximum throughput of a topology is directly related to the peak data rate that a system can sustain. In NoCs, throughput is closely related to two other parameters, namely, number of links (L) and average number of hop counts (H).

The number of links for Mesh-based 2D and 3D NoCs are presented in Equation 4.2 and Equation. 4.3, respectively.

$$L_{2D} = 2N_X N_Y - (N_X + N_Y) \tag{4.2}$$

$$L_{3D} = N_X N_Y (N_Z - 1) + N_X N_Z (N_Y - 1) + N_Y N_Z (N_X - 1) \tag{4.3}$$

where N_X, N_Y and N_Z are the numbers of routers in X, Y and Z dimension, respectively. Table 4.3 shows the number of links for different Mesh-based 2D and 3D NoCs, as well as the percentage increase by using 3D topologies.

With a larger number of links, a 3D NoC, comparing with its 2D counterpart, is able to contain more flits concurrently and therefore, transmit a greater number of messages. Furthermore, it also provides more alternative routing paths between PEs and thus is more fault tolerant in case of link or router failures.

The average number of hop counts depends on the average distance between all pairs of routers in a network. Based on the assumption of uniform random traffic pattern, Equation 4.4 and Equation 4.5 show the average

Table 4.3: Number of links

2D NoC	3D NoC	L_{2D}	L_{3D}	Percentage Increase
8×8	4×8×2	112	136	21.43%
8×8	4×4×4	112	144	28.57%
12×12	6×6×4	264	348	31.82%
18×18	6×9×6	612	828	35.29%
27×27	9×9×9	1404	1944	38.46%

number of hop counts in Mesh-based 2D and 3D NoCs. In addition, as shown in Equation 4.6 and Equation 4.7, the average number of hop counts in 3D NoCs can be divided into two components, which are the horizontal and vertical hop counts.

$$H_{2D} = \frac{(N_X N_Y - 1)(N_X + N_Y)}{3(N_X N_Y - 1)} \tag{4.4}$$

$$H_{3D} = \frac{N_X N_Y N_Z (N_X + N_Y + N_Z) - N_Z(N_X + N_Y) - N_X N_Y}{3(N_X N_Y N_Z - 1)} \tag{4.5}$$

$$H_{3D,H} = \frac{N_Z (N_X + N_Y)(N_X N_Y - 1)}{3(N_X N_Y N_Z - 1)} \tag{4.6}$$

$$H_{3D,V} = \frac{(N_Z^2 - 1) N_X N_Y}{3(N_X N_Y N_Z - 1)} \tag{4.7}$$

Table 4.4 shows the average number of hop counts for different Mesh-based 2D and 3D NoCs, as well as the percentage improvement by using 3D topologies.

Table 4.4: Average Number of Hop Counts

2D NoC	3D NoC	H_{2D}	H_{3D}	Percentage Improvement
8×8	4×8×2	5.33	4.44	19.70%
8×8	4×4×4	5.33	3.81	28.52%
12×12	6×6×4	8.00	5.17	35.38%
18×18	6×9×6	12.00	6.87	42.75%
27×27	9×9×9	18.00	8.90	50.56%

Having much smaller number of hop counts, 3D NoCs allow flits to traverse less number of stages between source and destination nodes than the 2D counterparts, which results in shorter communication delay and lower power consumption. Moreover, with a lower hop count, a wormhole-routed packet will utilize fewer links, thus leaving more free links to increase the maximum sustainable traffic.

It is obvious from the previous two tables that 3D NoCs outperform the 2D counterparts from the topological point of view and under the zero-load assumption. In practice, however, the system throughput and network delay are also highly dependent on network congestion. In this thesis, we compare three different 2D/3D NoC topologies with the same amount of PEs, namely, 8×8 2D NoC, 4×8×2 3D NoC and 4×4×4 3D NoC. This comparison is based on the assumptions that the 3D NoCs are fully connected and the time distribution of traffic is the memory-less Poisson distribution. Five different packet injection rates have been used to show the global average throughput and delay under different network loads. From the results of the comparisons as shown in Figure 4.8, it is clear that the performance benefits are more obvious as the network load increases.

To summarize, the system throughput and delay are improved significantly as the number of vertical layers increases. There are other parameters to evaluate the performance of different NoC topologies such as path diversity. However, in this chapter, we only use network delay D to reflect the performance improvement, as shown in Equation 4.8.

$$\delta_{delay} = \frac{D_{2D} - D_{3D}}{D_{2D}} \tag{4.8}$$

4.3.3 Power Consumption and Thermal Concerns

As discussed in the previous chapter, there are three power consumption components for NoC based systems, namely, dynamic power consumption, short circuit power and leakage power consumption, which can be calculated by using Equations 4.9 - 4.11. Dynamic power consumption is the dissipated power due to the charge and discharge of an interconnect and input gate capacitance during a signal transition. Short circuit power is due to the DC current path that exists in a CMOS circuit during a signal voltage changes between V_{gnd} and V_{dd}. The leakage power is comprised of two power components, the sub-threshold and gate leakage currents. The sub-threshold power consumption is due to current flowing in the cut-off region, causing I_{sub} current to flow. The gate leakage component is due to current flowing through the gate oxide, denoted as I_g [79].

$$P_{dynamic} = \alpha f C V_{dd}^{2} \tag{4.9}$$

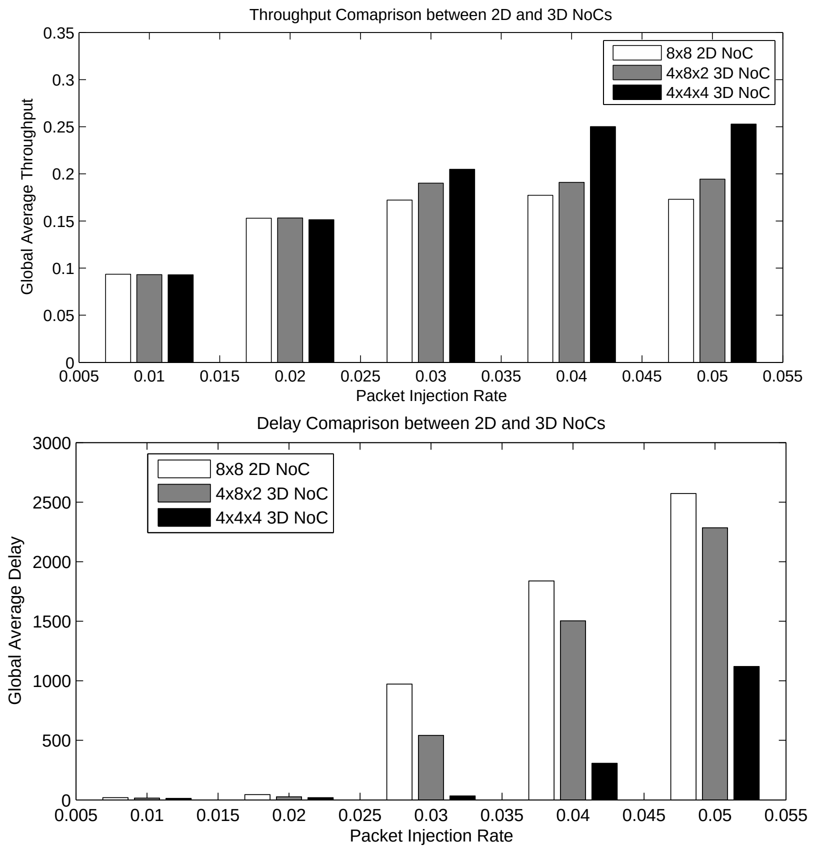


Figure 4.8: Comparisons of throughput and delay between 2D and 3D NoCs

$$P_{short\_circuit} = \frac{4\alpha f I_{d0}^2 t_{ri}^2 V_{dd} k_i h_i^2}{V_{dsat} G C_{effi} + 2 H I_{d0} t_{ri} h_i} \tag{4.10}$$

$$P_{leakage} = h_i k_i V_{dd}(I_{sub0} + I_{g0}) \tag{4.11}$$

Comparing with 2D NoCs, 3D architectures are expected to consume less power, especially dynamic power, due to the difference in circuit capacitance. In [112], authors compare ICs with different number of layers using different technologies. Up to 20% power reduction has been shown using 35nm technology and with 4 layers. It is worthwhile to notice that there is a limit on how much the reduction can be achieved by 3D integration. Experiments show that integrating too many device layers may increase the power consumption. Therefore, architectures of 3D NoCs should be carefully analysed to achieve the optimal power advantage (shown in Equation 4.12)

$$\delta_{power} = \frac{P_{2D} - P_{3D}}{P_{2D}} \tag{4.12}$$

However, although 3D ICs may bring power advantages, heat dissipation remains as one of the most critical concerns [53] [42]. In high-performance 2D ICs, for the communication components such as links and routers as well as processing elements including computation units and memories, thermal effects can significantly impact both reliability and performance. This issue is expected to be exacerbated by the reduction in chip size. Assuming that a 3D NoC generates the same heat as its 2D counterpart, the heat dissipation and cooling are much more challenging due to its smaller footprint, less inter-layer space, harder air circulation and higher power density. According to our previous research in [110], without direct contact with heat sink, the peak chip temperature of 3D design raises by 29°C comparing with the 2D design, which is infeasible for many applications.

Therefore, higher cooling cost is required in 3D NoCs. This cost is determined by the cooling solutions and the thermal dissipation of the system. In [113], a cooling cost model is proposed as

$$C_{cool} = K_c t + c \tag{4.13}$$

where K_c and c are the cooling cost parameters, and t is the temperature. To conclude, 3D NoCs require more cooling cost than 2D NoCs, and the negative factor is shown in Equation 4.14.

$$\zeta_{cool} = \frac{C_{cool,2D} - C_{cool,3D}}{C_{cool,2D}} \tag{4.14}$$

To alleviate the thermal challenge, we adopt a solution where processors and cache are separated into different vertical layers. With the current IC technology, an increasing proportion of chip size is used by memories. Take IBM Power7 processor [41] an example, even with 8 four-thread processing cores, more than half of the die area is still attributed to the embedded L2 and L3 caches and memory controllers. Therefore, it is natural to apply 3D chip integration by partitioning all the processing cores to one layer and other components to other layers. In consideration of heat dissipation, the processor layer should be placed on the top layer which is close to the heat sink. The detailed architecture design will be illustrated in Section 4.4.

4.3.4 TSV Analysis

Although providing the convenient communication links for layers in vertical direction, TSVs suffer from severely increased yield loss. Due to the immature fabrication and bonding technologies [65], the failure rate for each TSV varies from 50 *ppm* to even 5% [16], with the current TSV process technology. At the same time, the number of TSVs increases dramatically as the complexity of designs increase. In large scale 3D NoCs, thousands of TSVs are required for a single layer [38]. Based on these two factors, the yield of TSV can be calculated as

$$Y_{TSV} = (1 - f)^{N_{TSV}(N_Z - 1)} \tag{4.15}$$

where N_{TSV} is the number of TSVs per layer and f is the failure probability for one TSV.

Via basic calculations, it is obvious that the yield is too low in large scale 3D NoCs. A simple and commonly used solution to alleviate this problem is redundant TSV insertion [38]. In this solution, m number of TSVs are implemented for the same signal. Thus, the yield is increased to

$$Y_{TSV\_after\_insertion} = (1 - f^m)^{N_{TSV}(N_Z - 1)} \tag{4.16}$$

where N_{TSV} remains the same as it is in Equation 4.15 since the number of signals remains the same.

However, higher manufacturing cost is required by having a larger number of TSVs. There are three major tasks of 3D integration processing: the first is TSV formation for vertical interconnects, the second is wafer thinning and backside processing, and the third is stacking of chips. These tasks are major costs for processing a 3D wafer. It is shown in [101] that based on IMEC 3D cost model, TSV processing cost is the dominating cost for a 3D wafer. Assuming a CMOS processing technology of 65nm with 200mm silicon wafers, 46% to 65% costs are spent on TSV processing, with annual production volume varies from 10,000 to 300,000 wafers respectively.

It is also important to note that the TSV process does not scale with the CMOS technology. TSV diameters and pitches are two to three order of magnitude larger than transistor gate lengths. When moving towards newer technologies, the area cost for TSVs do not change. And to make it worse, the percentage of area cost by TSVs will become more significant as the feature sizes shrink.

4.3.5 Circuit-level Analysis

From the circuit-level point of view, the 3D integration technology reduces both die area and the number of required metal layers, especially in large scale systems. The estimation of die area is shown in Equation 4.17

$$A_{die} = N_g * A_g \tag{4.17}$$

where A_{die} is the estimated die area, N_g is the number of gates, and A_g is an empirical parameter that shows the proportional relationship between area and gate counts. In [29], based on the empirical data in the industrial designs, the authors claim that $A_g = 3125 \times \lambda^2$, in which λ is half of the feature size for a specific technology node. In 3D ICs, since the gates are distributed to several layers, the $N_g$of each layer is much smaller and thus the die area is reduced significantly comparing with the traditional 2D ICs.

The number of metal layers depends on the complexity of the interconnect and size of the dies. A simple metal layer estimation model is proposed in [104], deriving from the average wire length:

$$n_w = \frac{f_g \overline{R_m} P_w}{e_w d_g} \tag{4.18}$$

where f_g refers to the average gate fan-out, $P_w$to wire pitch, $e_w$to the utilization efficiency of metal layers, $\overline{R_m}$to the average wire length, d_g to the gate dimension, and $n_w$is the estimated number of metal layers. According to Donath's model [28], the average wire length can be calculated by

$$\overline{R_m} = \begin{cases} \frac{2}{9}\frac{1-4^{(p-1)}}{1-N_g^{(p-1)}}\left(7\frac{N_g^{(p-0.5)}-1}{4^{(p-0.5)}-1} - \frac{1-N_g^{(p-1.5)}}{1-4^{(p-1.5)}}\right) & \text{if } p \neq 0.5 \\ \frac{2}{9}\frac{1-4^{(p-1)}}{1-N_g^{(p-1)}}\left(7log_4 N_g - \frac{1-N_g^{(p-1.5)}}{1-4^{(p-1.5)}}\right) & \text{if } p = 0.5 \end{cases} \tag{4.19}$$

where p is the "Rent exponent" [18], which depends on the circuit topology.

However, this estimation equation suffers from two major limitations, namely, the assumption of same utilization efficiency and wire width on different layers, and the lack of TSV area overhead consideration. In [29], an improved 3D router model is proposed based on the wire length distribution rather than average wire length. In this model, the available routing area of each metal layer is estimated at first, and designers try to route as many shorter interconnects as possible on lower metal layers. Table 4.5 shows the estimated reduction in the number of metal layers by using different gate counts and number of layers.

It is obvious from Table 4.5 that the metal layer reduction is more significant as the design complexity increase. Considering the Moore's Law, this reduction will become increasingly valuable for IC manufacturers. Therefore, we include the metal layer reduction δ_{MLR} in the change function as a positive factor.

$$\delta_{MLR} = \frac{N_{2DML} - N_{3DML}}{N_{2DML}} \tag{4.20}$$

where N_{2DML} and N_{3DML} refer to the number of metal layers in 2D and 3D designs, respectively.

Table 4.5: The Estimated Number of Metal Layers in 65 nm Technology [29]

Gate Counts	2D	2-layer 3D	3-layer 3D	4-layer 3D
5M	5	5	5	4
10M	6	5	5	5
20M	7	6	5	5
50M	8	7	7	6
100M	10	8	7	7
200M	12	10	9	8

4.3.6 Yield Analysis

The IC yield modelling has been under research for several decades. In 1960s, researchers proposed a simple but well-known Poisson yield model for chip level yield analysis, using the chip area and defect density, as shown in Equation 4.21 [71].

$$Y = e^{-D_0 A} \tag{4.21}$$

where D_0 is the mean defect density and A is the die area. This model is based on the approximation of a binomial random variable by a Poisson random variable and it is assumed that defects are uniformly distributed over an entire chip.

However, this model tends to be overly optimistic and in reality, it can rarely be used. In this thesis, we use the 2D yield model proposed in [70] using a gamma function based distribution function, as shown in Equation 4.22.

$$Y_{2D} = (1 + \frac{D_0 A_{2D}}{\alpha})^{-\alpha} \tag{4.22}$$

where the model parameter $\alpha = (\mu_D/\sigma_D)^2$, which is an experimental value depending on the complexity of the manufacturing process, can be used to account for defect clustering. By varying α , this model covers the entire range of yield prediction. For $\alpha = 1$ the yield reduces to the Price formula (exponential weighting), and for $\alpha = \infty$ it becomes the simple Poisson formula (no clustering).

A 3D yield model has been proposed in [69], as

$$Y_{3D} = [1 + \frac{D_0}{\alpha}(\frac{A_{2D}}{N_Z} + A_{overhead})]^{-\alpha} Y_S^{N_Z} \tag{4.23}$$

where $A_{overhead}$ is the area overhead cost by 3D integration, and Y_s is the yield of single layers.

In this thesis, based on this model, we use the models shown as Equation 4.24 and 4.25, to support wafer-to-wafer (W2W) and die-to-wafer (D2W) stacking technologies, with the consideration of area costs for TSVs and DFT.

$$\begin{aligned} Y_{W2W} &= [1 + \frac{D_0}{\alpha}(\frac{A_{2D}}{N_Z} + A_{TSV} + A_{DFT})]^{-\alpha} Y_{TSV} \\ & \prod_{i=1}^{N_Z-1} Y_{s,W2W,i}(1 + \frac{D_0(\frac{A_{2D}}{N_Z} + A_{TSV})}{\alpha})^{-\alpha} \end{aligned} \quad (4.24)$$

$$Y_{D2W} = [1 + \tfrac{D_0}{\alpha}(\tfrac{A_{2D}}{N_Z} + A_{TSV} + A_{DFT})]^{-\alpha} Y_{TSV} \textstyle\prod_{i=1}^{N_Z-1} Y_{s,D2W,i} \quad (4.25)$$

To summarize, due to more complicated manufacturing processes, the yield of 3D NoCs are lower than that of 2D NoCs. Therefore, the yield effect in the change function is negative, as shown in Equation 4.26.

$$\zeta_{yield} = \frac{Y_{2D} - Y_{3D}}{Y_{2D}} \quad (4.26)$$

where Y_{3D} can be either Y_{W2W} or Y_{D2W}, depending on the technology used.

4.3.7 Testing Costs

As the complexity of on-chip systems increase, the testing cost of chips has become more significant. In this section, we quantitatively compare the testing costs of 2D and 3D NoCs. The models and analysis are based on the research results in [72] (for 2D designs) and [16] (for 3D designs). In these papers, researchers focus on the analysis of general ICs. However, since NoCs is a subset of general ICs, these models can be applied to NoCs with little modification.

For both 2D and 3D NoCs, the total testing costs are calculated using Equation 4.27.

$$C_{test} = C_{prep} + C_{exec} + C_{silicon} + C_{quality} \quad (4.27)$$

Test preparation cost (C_{prep})

C_{prep} represents all fixed costs of test generation, testing program generation, and any design efforts for test-related purposes including software systems. For 3D NoCs, C_{prep} also includes the preparation cost for pre-bond testing of TSVs. In this thesis, both W2W and D2W bonding are considered, which results in different cost models for 3D NoCs.

With the assumption that the cost associated with tester program preparation is proportional to the cost of test generation, the preparation cost of 2D NoCs is calculated as

$$\begin{aligned} C^{overall}_{prep,2D} &= \frac{C_{test\_gen} + C_{test\_prog} + C_{DFT\_design}}{Y_{2D}} \\ &= \frac{1}{Y_{2D}}[(1+\beta)\frac{R_{person\_hour} K_{test\_gen} e^{A}}{V} + C_{DFT\_design}] \end{aligned} \tag{4.28}$$

where C_{test_gen} is the test-pattern generation cost, C_{test_prog} is the tester-program preparation cost, C_{DFT_design} is the additional design cost for DFT, Y_{2D} is the yield of 2D NoCs, R_{person_hour} is the cost of one person-hour for test generation, V is the total number of ICs produced, K_{test_gen} is a constant coefficient, A is the die area, and β is a factor of less than 1 which models the difficulty of translating test vectors to the tester's format.

For 3D NoCs using W2W bonding, since no pre-bonding test will be conducted, all of the cost factors should target at the whole chip, instead of a single layer, as shown in Equation 4.29.

$$C^{overall}_{prep,W2W} = \frac{C^{overall}_{test\_gen} + C^{overall}_{test\_prog} + C^{overall}_{DFT\_design}}{Y_{W2W}} \tag{4.29}$$

If D2W bonding is used, each layer will be tested before bonding, and thus the cost model for test preparation changes correspondingly.

$$C^{overall}_{prep,D2W} = \frac{Y_N C_{prep}(N) + \sum_{i=1}^{N-1} C_{prep}(i)}{Y_{D2W}} \tag{4.30}$$

where $C_{prep}(i)$ is the preparation cost for the i_{th} layer and calculated according to Equation 4.30.

Test execution cost (C_{exec})

C_{exec} consists of costs of test-related hardware such as probe cards and the cost incurred by testers.

$$C_{exec} = \frac{(C_{hw} + C_{tester})}{Y} \tag{4.31}$$

The cost models of probe cards for 2D and 3D NoCs are:

$$C_{hw,2D} = C_{hw,W2W} = \frac{Q_{probe}\lceil V/N_{probe\_life}\rceil}{V} \tag{4.32}$$

$$C_{hw,D2W} = \frac{Q_{probe}\lceil NV/N_{probe\_life}\rceil}{V} \tag{4.33}$$

where Q_{probe} is the unit price of a probe card and N_{probe_life} is the number of ICs a probe card can test.

The cost of testers can be calculated as

$$C_{tester} = (\gamma + \frac{1-\beta_{util}}{\beta_{util}})\frac{K_{cap}K_{pins}\sqrt{A}\beta_{depr}}{T_{sec\_per\_year}T_{test}} \tag{4.34}$$

where γ is the cost ratio between active and inactive testers, β_{util} is the tester utilization factor, K_{cap} is tester price per pin, K_{pins} relates the number of pins to die area, β_{depr} is the annual depreciation rate, $T_{sec\_per\_year}$ is the number of seconds per year, and T_{test} is the testing time. For 2D, W2W and D2W bonding 3D NoCs, die area A is calculated as

$$A_{2D} = A_{D2W}(i) = A(i) + A_{DFT}(i) \tag{4.35}$$

$$A_{W2W}(i) = \begin{cases} A_i, & \text{if } i < N_Z \\ A_i + A_{DFT}, & \text{if } i = N_Z \end{cases} \tag{4.36}$$

where A_{DFT} is the area cost for DFT purpose.

Silicon overhead ($C_{silicon}$)

Testing usually requires some extra circuits to be implemented on chips, and therefore costs additional silicon area overhead. Equation 4.37 models the silicon overhead for 2D, W2W and D2W bonding 3D NoCs.

$$C_{silicon} = \frac{Q_{wafer}}{\pi R_{wafer}^2 \beta_{waf\_die}} \lceil \frac{A_{DFT}^{overall}}{Y} \rceil \tag{4.37}$$

where Q_{wafer} is the wafer cost, R_{wafer} is the wafer radius, β_{waf_die} is the percentage of wafer area that can be divided into dies, and

$$A_{DFT}^{overall} = \begin{cases} A_{DFT,2D} & for\ 2D\ NoCs \\ A_{DFT,2D} & for\ W2W\ bonding \\ \frac{2N_Z-1}{N_Z}A_{DFT,2D} & for\ D2W\ bonding \end{cases} \tag{4.38}$$

Imperfect test quality ($C_{quality}$)

There are several cost factors due to imperfect test quality, such as the loss in profit from performance degradation, cost of test escape and the cost of good dies being deemed faulty. In this thesis, however, only test escape is considered. In [107], a model of test escape rate is modelled as

$$E_r = 1 - Y^{1-f} \tag{4.39}$$

where f is the fault coverage. This model applies for both 2D NoCs and 3D NoCs using W2W bonding. For D2W bonding, 3D NoCs, the model is adapted as

$$E_{r,D2W} = 1 - \prod_{i=1}^{N_Z} Y_i^{1-f_i} \tag{4.40}$$

The cost of testing due to imperfect test quality is then modelled as

$$C_{quality} = E_r \alpha_{escape}(C_{prep} + C_{exec} + \frac{Q_{wafer}}{\pi R_{wafer}^2 \beta_{waf\_die}} \lceil \frac{A}{Y} \rceil) \tag{4.41}$$

To summarize, testing in 3D NoCs are more expensive than it is in 2D architectures. Therefore, as shown in Equation 4.42, testing is regarded as a negative factor in the change function.

$$\zeta_{test} = \frac{C_{test,3D} - C_{test,2D}}{C_{test,2D}} \tag{4.42}$$

4.3.8 Functional Integration

So far, we have analysed and compared 2D and 3D NoCs in terms of benefits, including communication delay (δ_{delay}), power consumption (δ_{power}), area consumption (δ_{area}), reduction in the number of metal layers (δ_{MLR}), as well as costs including cooling (ζ_{cool}), TSV manufacturing (ζ_{TSV}), yield (ζ_{yield}) and testing (ζ_{test}). Therefore, by integrating these factors, the change function in Equation 4.1 becomes

$$\begin{aligned}\epsilon &= \omega_{delay}\delta_{delay} + \omega_{power}\delta_{power} + \omega_{area}\delta_{area} \\ &\quad + \omega_{MLR}\delta_{MLR} - \omega_{cool}\zeta_{cool} - \omega_{TSV}\zeta_{TSV} \\ &\quad - \omega_{yield}\zeta_{yield} - \omega_{test}\zeta_{test}\end{aligned} \tag{4.43}$$

As mentioned in Section 4.3.1, only when the weighted sum of the percentage benefits and costs is larger than a user defined threshold T_{exp} (or simpler 0), is it worthwhile to adopt a 3D NoC architecture instead of its 2D counterpart. The reason for having this change function is to emphasize that utilizing the latest technology is not without cost and thus designers should always consider whether it is worthwhile to make the change. This idea of assessing cost might not be new, however, how to quantify the trade-off between new technology and cost remains unsolved. To the extent of our knowledge, it is the first time that change function is applied in the field of IC design.

Undoubtedly the following issues need to be clarified beforehand in order to make the most accurate assessments. Firstly, for each factor, no matter positive or negative, the weight needs to be decided according to the business needs. It is very likely that for a certain factor, its weight varies in different circumstances. For example, the weight for power consumption in wireless sensor networks would be larger than it is for basic home desktop computers. Secondly, in this thesis, we are providing a list of factors as a reference usage of change function, and thus it does not cover all the possibilities. For instances such as military or aerospace electronics, the system reliability in extreme situations should be considered carefully.

4.4 Case Study: 3D Integration with On-Chip DRAM Memory

As explained in Section 4.3.3, 3D ICs suffer very much from the thermal and heat dissipation challenges. Therefore, it is recommended to place processing units on the top layer which is close to the heat sink and reserve other layers for cache and memory which do not generate much heat. In this section, a case study of 3D integration with on-chip DRAM memory is presented which can serve as a guideline for future 3D on-chip memory. Comparing with the traditional off-chip memory designs, 3D on-chip DRAM enables significant power reduction since transferring a signal via on-chip links can reduce its power consumption by 10 to even 100 fold [22]. Additionally, shorter wires also reduce power consumption by producing less parasitic capacitance. Moreover, faster communication and higher system performance shorten the processing time of an application, which further optimizes the total consumed energy.

4.4.1 Introduction of 3D NoC with DRAM

Static Random Access Memory (SRAM) is widely used as on-chip cache for CMPs. It uses bistable latching circuitry to store bits, avoiding periodical refreshments. The circuits used to store one bit in a SRAM contain typically six transistors. Unlike SRAM, Dynamic Random Access Memory (DRAM) stores each bit by using only one transistor. Hence the storage density is much higher. The main disadvantage of DRAM is its requirements of periodical refresh processes to keep the leak charge of capacitors within the accepted range. Thus, under the same technology era, the performance of DRAM is lower than SRAM.

There is a significant concern for the memory bandwidth, especially with the growing number of memory requests as the number of cores increases. In the era of Pentium 3 and 4, the processor has only one core (later versions such as Pentium-D has two cores on one die), memory bandwidth requirement is thus not so critical. As the number of processor cores grows, the requirement of memory bandwidth grows as well. As shown in Table 4.6, Core 2 Duo doubles the requirement of memory bandwidth to fit the requests of two cores. The system performance will decline if the memory bandwidth cannot sustain the growth rate requested by processor cores. By plugging two identical DIMM modules on the motherboard, dual channel can be configured to provide double bandwidth. In the dual channel, data is transferred in a 128-bit flavour instead of conventional 64-bit in one cycle. Triple channel is introduced with DDR3 memory, providing 192-bit data transfer in a clock cycle. Configured with triple channel PC3-8500 DDR3 memory, the maximum theoretical memory bandwidth for Intel Core i7 980X is thus 25.6GB/s [43].

Table 4.6: Processor and memory bandwidth for one channel

Processor	**Core**	**Typical memory**	**Typical BW**
Pentium 3	1	PC-133 SDRAM	1.066 GB/s
Pentium 4	1	PC-1600 DDR	1.6 GB/s
Core 2 Duo	2	PC2-3200 DDR2	3.2 GB/s
Core 2 Quad	4	PC2-6400 DDR2	6.4 GB/s
Core i7 980X	6	PC3-8500 DDR3	8.5 GB/s

Increasing the memory bandwidth by using DDR4 seems to be a solution, quadrupling or even quintupling the number of memory channels is another solution. However, as mentioned earlier, triple channel configuration requires at least three DIMM modules, which increases cost, fault rate and power consumption. Another constraint is the pin count limitation. It is predicted by the ITRS roadmap that pin count will increase by about 10% each year only, comparing with the number of cores that is expected to double every 18 months [8].

Brian M. Rogers et. al. [83] developed a mathematical model to evaluate the impact of memory bandwidth on CMP scaling in different technologies. However, the authors focus only on the theoretical studies in this work. In [105], the organization and performance of 3D memory in NoC are analysed. They assumed a simple NoC model with uniform random traffic and local traffic. Gabriel H. Loh presented a novel 3D-stacked memory architecture for CMP [59]. It is claimed that a 1.75$\times$ speed-up is achieved over previous approaches. Nevertheless the paper presumed a conservative quad-core configuration.

In this section, we investigate the design of 3D NoC with memory on chip. Figure 4.9 shows an example of traditional off-chip memory design for 2D NoC architecture. When reading data from or writing data to the memory, a transmission delay is incurred. The delay for modern system is usually hundreds of cycles (e.g. 200-300). Comparing with on-chip cache, the off-chip memory is usually tens of times slower.

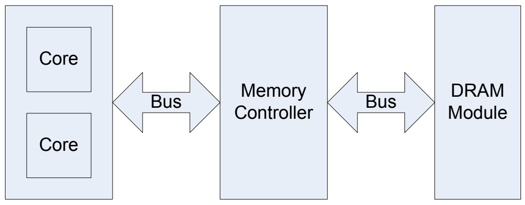


Figure 4.9: An example of 4×4 traditional NoC design using mesh topology, memory controllers are on-chip, memory modules are off-chip.

4.4.2 Modelling of 3D NoC with DRAM Integration

It is expected that since the processors consume overwhelming majority of power in a chip, stacking multiple processor layers could be unwise for heat dissipation. As discussed in Section 4.3.3, heat dissipation is a major problem by stacking multiple processor layers even if processors are interlaced vertically. For example, as shown in [110], without direct contact with heat sinks, the peak chip temperature of 3D design may raise by up to 29°C comparing with the 2D design, which is infeasible for some applications. However, by stacking more memory layers instead of processor layers, the thermal constraint is supposed to be alleviated. Gian Luca Loi et. al. shows that, even for 18 stacked layers (1 of processors, 1 of cache and 16 of memory), the maximum temperature for a 3D chip increases only 10°C comparing with 2D chip [60]. It is estimated that 15% lower core frequency of a 3D chip could compensate the thermal drawback [60].

The floorplan of modern multi-core chips such as third-generation Sun SPARC [97], IBM Power 7 [41], AMD Istanbul [6] show the possibility of 3D NoC. The total area of Sun SPARC chip is 396mm^2 with 65nm fabrication technology. Scaled to 32nm technology, each core has an area of 3.4mm^2. We simulate the characteristics of a 64MB, 64 banks, 64-bit line size, 4-way associative, 32nm cache by CACTI [89]. Results show that the total area of cache banks is 204.33mm^2. A cache bank, including data and tag, occupies 3.2mm^2. We also simulate the characteristics of a 1GB, 8 banks, 32nm DRAM memory by CACTI [89]. It is revealed that the total area of the memory is 212.79mm^2.

On account of the aforementioned analysis, we use a 3D NoC model based on 32nm fabrication technology, with one layer of processor, one layer of cache and several layers of memory. In consideration of heat dissipation, the processor layer should be on top of the chip (near heat sink). The top layer is a 8×8 mesh of Sun SPARC cores. The cache layer has a 8×8 mesh of cache banks. It is noteworthy that routers are quite small compared with processors and cache banks, e.g. scaled to 32nm, a 7-port 3D router is estimated to be only 0.096mm^2 [76]. The total area of the chip is supposed to be around 230mm^2.

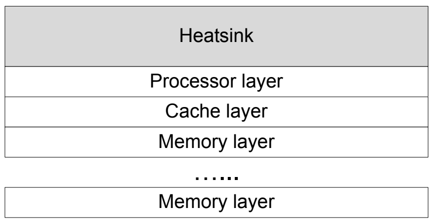


Figure 4.10: Schematic diagram of a 3D chip with multiple stacked layers. The heat sink is attached with processor layer

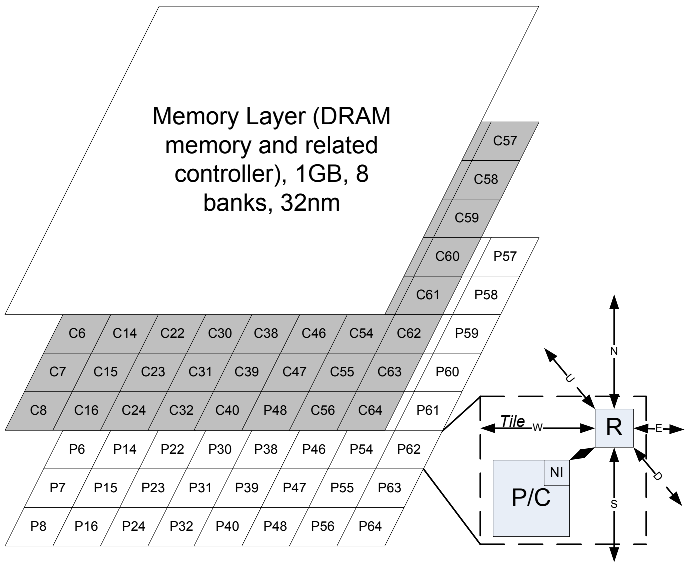

Memory Layer (DRAM memory and related controller), 1GB, 8 banks, 32nm
C57
C58
C59
C60
C61
C6 C14 C22 C30 C38 C46 C54 C62
C7 C15 C23 C31 C39 C47 C55 C63
C8 C16 C24 C32 C40 P48 C56 C64
P57
P58
P59
P60
P61
P6 P14 P22 P30 P38 P46 P54 P62
P7 P15 P23 P31 P39 P47 P55 P63
P8 P16 P24 P32 P40 P48 P56 P64
Tile
R
NI
P/C
N
E
S
W
U
D

Figure 4.11: 3D NoC with one processor layer (Px), one cache layer (Cx) and one memory layer, layers are fully connected by TSVs (not shown in figure)

Figure 4.11 shows the aforementioned 3D NoC with three layers, however more layers of memory can be stacked.

4.4.3 Experimental Setup

In this section, we present the experimental evaluation under different memory configurations. Applications are selected from SPLASH-2 [108] and PARSEC [11]. SPECjbb [91] and TPC-H [96] are used as synthetic benchmarks.

The simulation platform is based on a cycle-accurate 3D NoC simulator which can produce detailed evaluation results. The platform models the routers, horizontal links and vertical pillars accurately. The state-of-the-art router in our platform includes a routing computation unit, a virtual channel allocator, a switch allocator, a crossbar switch and four input buffers. Deterministic routing algorithm has been selected to avoid deadlocks.

Table 4.7: System configuration parameters

Processor configuration	
Instruction set architecture	SPARC
Number of processors	64
Issue width	1
Cache configuration	
L1 cache	Private, split instruction and data cache, each cache is 16KB. 4-way associative, 64-Byte line, 3-cycle access time
L2 cache	Shared, distributed in 64 nodes, unified 64MB (64 banks, each 1MB). 64-Byte line, 6-cycle access time
Cache coherence protocol	MOESI
Cache hierarchy	SNUCA
Memory configuration	
Size	4GB DRAM
Access latency	260 cycles
Requests per processor	16 outstanding
Network configuration	
Router scheme	Wormhole
Flit size	128 bits

We use a 128-node network which models a single-chip CMP for our experiments. The 3D architecture in this paper has one layer for processors, one layer for shared cache memories and five layers of DRAM memory (one layer for logic) (for simplicity, Figure 4.11 shows only three layers). A full

Table 4.8: Benchmark descriptions

SPLASH-2	Standford's benchmark suite of parallel programs. Cholesky, FFT, FMM, LU, Ocean, Radix, Raytrace and Water-Nsq are used.
PARSEC	Princeton's benchmark suite for shared-memory computers based on CMP. Swaptions, a workload that employs Monte Carlo simulation to compute prices; and x264, an application that encodes H.264 videos are used.
SPECjbb	SPEC's standard Java server. Sun JRE-SE 1.4.2 with 16 warehouses is executed.
TPC-H	TPC's ad-hoc, decision support benchmark that examines large amount of data, executes queries and gives answers to critical business questions. MySQL v5.0.67 with 1GB of reference database, query 1 is used.

system simulation environment with 64 processors and 64 L2 cache nodes has been implemented. The simulations are run on the Solaris 9 operating system based on SPARC instruction set in-order issue structure. Each processor is attached to a wormhole router and has a private write-back L1 cache. The L2 cache shared by all processors is split into banks. The size of each cache bank node is 1MB; hence the total size of shared L2 cache is 64MB. The simulated memory/cache architecture mimics SNUCA [48]. A two-level distributed directory cache coherence protocol called MOESI based on MESI [77] has been implemented in our memory hierarchy in which each L2 bank has its own directory. The protocol has five types of cache line status: Modified (M), Owned (O), Exclusive (E), Shared (S) and Invalid (I). Orion [102], a power simulator for interconnection networks, is used to evaluate detailed power characteristics. A wormhole router is modelled in Orion, with corresponding input/output ports, buffers and the crossbar. Power consumption of routers is analysed. We use Simics [64] full system simulator as our simulation platform. The detailed configurations of processor, cache and memory configurations can be found in Table 4.7. Workloads used in this paper are shown in Table 4.8.

4.4.4 Performance Evaluation

The normalized full system simulation results are shown in Figure 4.12 and 4.13. The evaluation is based on average link utilization and the number of execution cycles. Since the interconnect network is consuming a significant proportion of NoC based system, reducing the link utilization is an efficient method for power optimization. Furthermore, lower link utilization incurs lower risk of communication faults and thus improves the system fault tolerance.

As is shown in Figure 4.12, our proposed design outperforms the traditional design in terms of average link utilization. Average link utilization is calculated with the number of flits transferred between NoC resources per cycle. Under the same configuration and workload, lower utilization means mitigated network load, which is favourable. Buffers in the routers are major consumers of static power consumption. Therefore, in networks with more mitigated load and thus less required buffers, lower power consumption can also be expected. Comparing with the traditional design, the average link utilization for our proposed design is reduced by 10.25%, on average. FFT and Cholesky have the most significant reduction of average link utilization, 13.34% and 12.81% respectively.

The results in Figure 4.13 show that our proposed design outperforms the traditional design in terms of executed cycles under all workloads. On average, our proposed design costs 1.12% less cycles than the traditional design, and the cycle reduction reaches 2.29% for LU workload and 1.77% for Radix respectively. The improvements of executed cycles can be interpreted as the result of the increased memory bandwidth and reduced memory access latency which commensurate with the number of memory accesses. The improvement of executed cycles is less remarkable comparing with average link utilization since local operations (e.g. core and cache) are not related with network operations.

4.4.5 TSV Evaluation

Due to the aforementioned high cost of TSVs, it is crucial to analyse the possible TSV deduction and the resulting impact on the system performance. In this subsection, we compare three different kinds of TSV configuration, namely, full connection, half connection and quarter connection, as shown in Figure 4.14 (for simplicity, only two layers are displayed).

The normalized average network latencies with different number of TSVs are shown in Figure 4.15. Not surprisingly, the design of full connection outperforms others in terms of network latency. The improvement is more notable in Cholesky (chol), FMM and TPC-H benchmarks, with 9.69%, 8.04% and 8.8% reduced latency, respectively, compared with the quarter

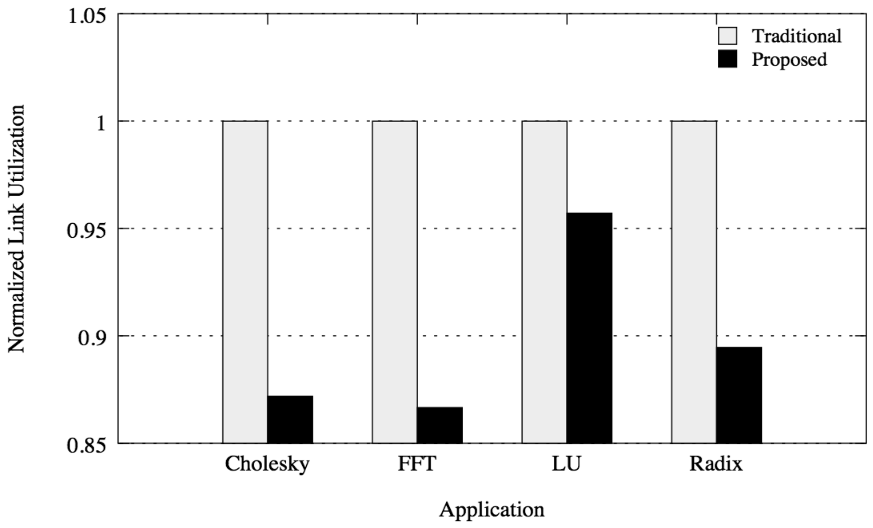


Figure 4.12: Normalized average link utilization with different configurations

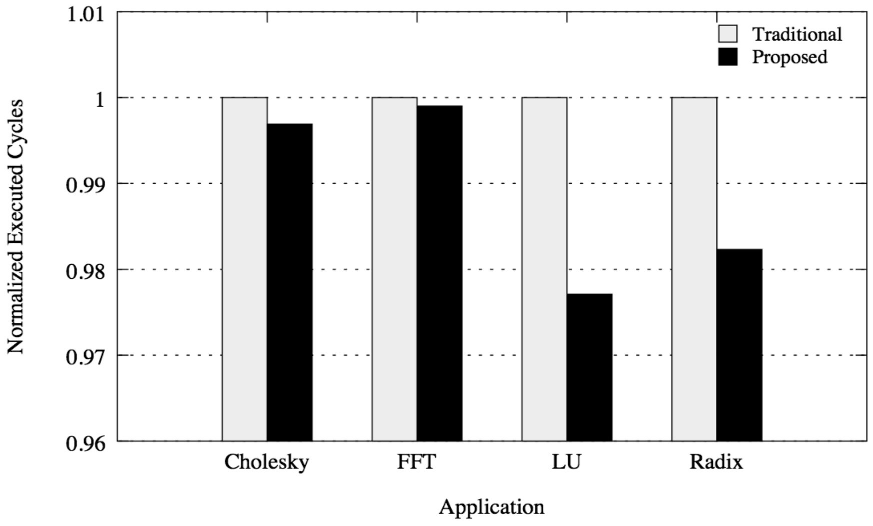


Figure 4.13: Normalized executed cycles with different configurations

design. This is primarily due to the reduced hop count of processor-cache data access in the full pillar connection than the other designs. We notice that performance difference for some of the applications, e.g. FFT and Radix (radi), is not that significant, the network latency is reduced by 1.09% and 2.93% respectively comparing with the quarter design. The reason is that, these applications have a lighter network load compared with the other applications. On average, compared with the quarter design, the network latency is reduced by 5.24% and 2.18% for full and half designs, respectively.

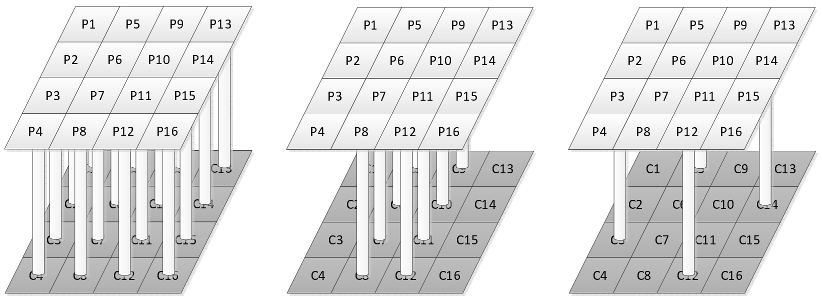


Figure 4.14: 3D NoCs with different number of TSVs

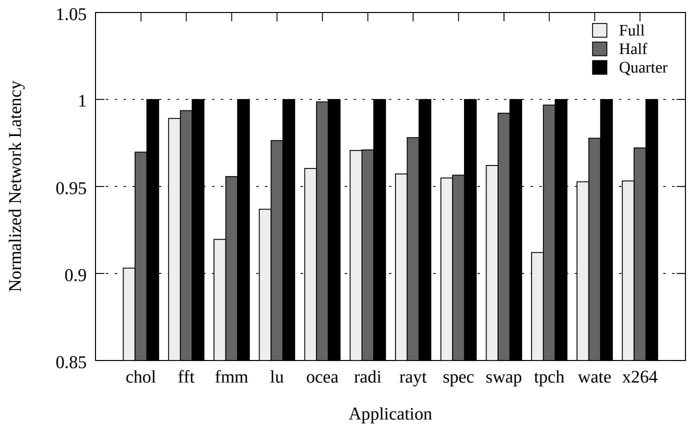


Figure 4.15: Normalized average network latency with different number of TSVs

To summarize, considering the relatively small impact on the network latency, quarter connection should be used in applications which are not performance-centered. According to our previous research in [109], the placement of quarter connection TSVs shown in Figure 4.14 is the optimal solution.

4.5 Chapter Summary

Three dimensional architecture is widely regarded as the trend of future computing platforms. By stacking and connecting multiple vertical layers, it enables higher integration density, smaller footprint, shorter communication delay, higher bandwidth, heterogeneous integration, etc. Moreover, thanks to the avoidance of long wires and reduction of off-chip communication, 3D NoCs provide higher energy efficiency comparing with their 2D counterparts.

In this chapter, we firstly present different stacking approaches with their pros and cons followed by the survey of two main 3D NoC architectures. Although 3D NoCs outperform 2D designs from many aspects, the benefits do not come for free. Therefore, we introduced the concept of change function for the choices of 2D and 3D NoCs which provides system designers and stakeholders a tool to quantify the feasibility of using 3D architecture in their NoC based systems. While illustrating the change function, we elaborated several key factors with the comparisons between 2D and 3D NoC architectures.

At the end of the chapter, a case study of 3D integration with on-chip DRAM memory is presented where processing units and memories are placed separately on different layers to alleviate the thermal challenges. In our study, we have shown that the 3D on-chip DRAM can significantly improve the system performances as well as energy consumption. Since the implementation of TSVs on 3D NoCs is costly, we have proposed a quarterly-connected TSV insertion architecture. Comparing with the fully-connected designs, the proposed method has much lower implementation cost and higher yield, with little performance loss.

Chapter 5

Agent based System Design

In this chapter, a hierarchical agent based NoC design approach with dynamic on-line services is proposed. It enables the system to autonomously adjust itself in order to achieve high energy efficiency as well as system reliability. Agents are functional units that monitor and control the NoC based system at different hierarchical levels, i.e., from cell level up to the whole system level. This architecture aims at the enhancement of the system performance in both power consumption as well as fault and variation tolerance aspects. It also provides a wide design and synthesis space for the realization of agents at each level.

5.1 Background of Agent based System

Considering the dynamic nature of the network traffic, run-time optimization techniques are most favourable in order to achieve maximum power efficiency, fault/variation tolerance and system flexibility. The demand for efficient system optimization while maintaining system scalability and low overhead brings in a trade-off between distributed and centralized monitoring. On one hand, local circuits need to be provided with distributed monitoring modules in order to achieve higher self optimizing and correcting efficiency. For urgent monitoring services, the distributed architecture not only enables the local agent operation at fine granularity, but also shortens the distance between agent modules and the monitored targets. Moreover, since the communication occurs only locally, this solution also prevents the potential communication bottleneck in the network. On the other hand, however, despite the system size, centralized monitoring is still an indispensable complement to localized monitoring schemes. Theoretically, a centralized monitor, with the knowledge of all on-chip resources, is able to coordinate and balance the functioning of all components with the aim of optimizing the overall system performance. In practice, for example, a single processing

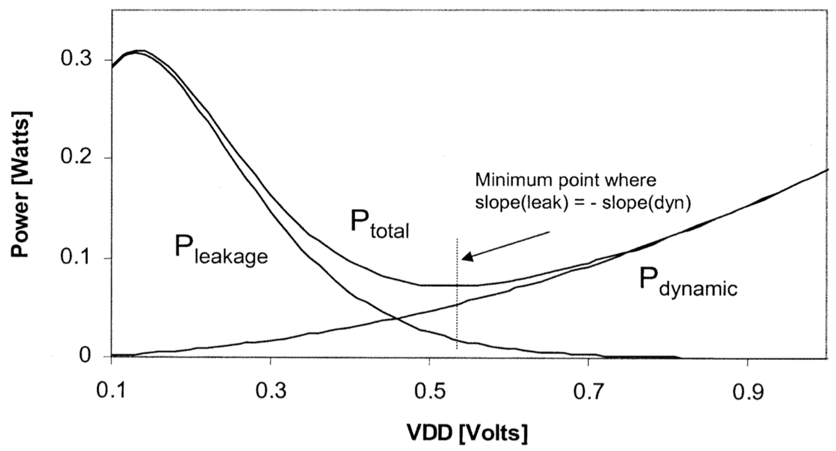


Figure 5.1: Dynamic and sub-threshold leakage power components for a fixed operating frequency [46]

unit for dynamic testing operations and a global level scheduler was adopted in [94] for better controllability thanks to the information stored centrally. For either distributed or centralized monitoring scheme, the energy efficiency of monitoring services should be maximized.

For a digital circuit, it is possible to trade-off dynamic power and sub-threshold leakage by balancing between V_{DD} and V_t to maintain performance [46]. Figure 5.1 shows theoretical curves for power versus supply voltage, where the V_t is implicitly adjusted to maintain a fixed frequency. The total power is the sum of the dynamic power curve and the leakage power curve and the minimum operating power corresponds to the V_{DD} choice where the slope of the two curves are opposite in sign but equal in magnitude. Therefore, run-time monitoring techniques can be used to keep the total power consumption near the optimal energy point.

5.2 Hierarchical Monitoring Agents

One of the key contribution of NoC architecture is the separation of computation and communication phases. In the proposed design method, as shown in Figure 5.2, a third phase which we call autonomous phase is added onto the NoC platform. It is implemented using a hierarchical monitoring agent based approach. The autonomous phase interacts actively with system resources including routers, links which are parts of the communication phase as well as processing units, memories, etc. which are within the computation phase.

Monitoring aware NoC design has been discussed in [20] where monitoring services are provided by monitoring agents distributively implemented

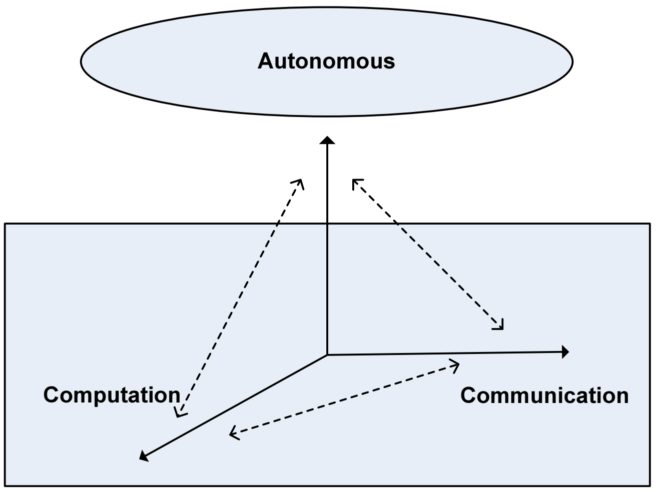


Figure 5.2: Autonomous phase in NoC designs

across the NoC platform and agents offer services on the same level. The major constrain of this approach is that it can only utilize either global or local information, but not both of them at the same time, which is desirable in large scale NoCs. Therefore, in this thesis, we propose hierarchical monitoring agent architecture to perform optimization at different levels.

5.2.1 Agent Hierarchy

There are four levels of agents in the proposed NoC architecture, namely, application agent, platform agent, cluster agent and cell agent, as shown in Figure 5.3. The application agent is the top level agent and thereby unique in a NoC platform. It is a software module capturing the application functionality and run-time performance requirements and constraints. Another unique component in a NoC is the platform agent. Based on the specification from the application agent and resource availability, the platform agent (re)configures both the network and Processing Elements (PEs). The entire NoC is divided into a number of clusters, each of which is monitored and controlled by a cluster agent. A cluster is a group of PEs with accompanying components such as caches, scratchpad memories, switches, links, etc. It is logically divided into cells which are the basic units in our architecture, consisting of a PE, a switch and the corresponding links. The cells are equipped with their own local monitors, the cell agents, which trace and adjust the local circuit conditions.

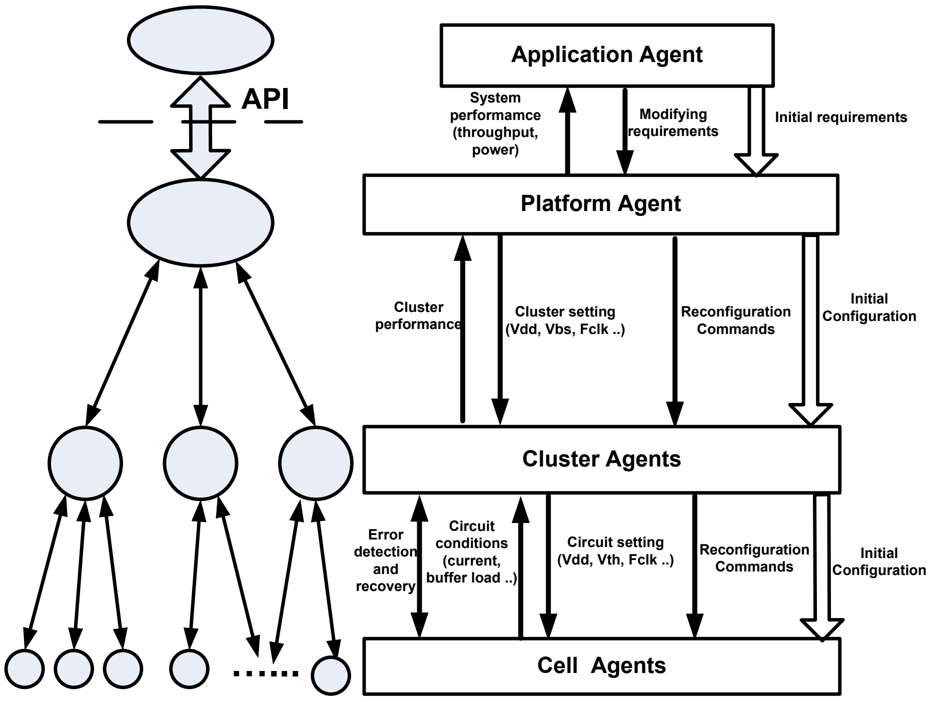


Figure 5.3: Hierarchical agent approach

5.2.2 Hierarchical Monitoring Approach

The proposed architecture highlights hierarchical approaches to various monitoring services. By the joint efforts of all levels of agents, the agent based NoC is able to monitor the real-time performances and thus autonomously adjust system parameters and operations in order to optimize the energy consumption and improve system robustness.

Before execution, the platform agent configures the network based on the initial application requirements with power and performance awareness [39]. A number of resources are reserved as spares in case of component failures. The initial configuration is enforced from the platform agent to the cluster and then cell agents.

After the application starts running, the cell agents are tracing their local circuit conditions, such as current (including leakage current for idle components), workload, and any faults or failures (for instance a link failure or a malfunctioning processing unit). Cell agents attempt to fix the errors if feasible (for example by retransmission in case of transient crosstalk-induced error [54]). The traced circuit conditions along with not-yet-solved failures are sent to cluster agents which attempt to adjust the cell settings based on these information. For instance they may scale the voltage supplies of a certain cell (commonly known as DVFS) or the threshold voltage by using

Adaptive Body Biasing (ABB) [67]. If a component has failed to work, they will acquire spare components and configure them into the cluster. Cluster agents send cluster performance to the platform agent. The information concerning cluster performance is represented at a coarser granularity than those sent between cluster and cell agents, for example, the power consumption of the cluster, or average network workload within the cluster, the error rate of the cluster. Based on these information, the platform agent may reconfigure the system, for instance assigning more spares into a failure-prone cluster, or scale down the voltage and frequency of a cluster with overwhelming power consumption. The overall system performance including the network throughput and power consumption, is reported by the platform agent to the application agent, which may modify the real-time application requirements. Figure 5.3 illustrates these hierarchical monitoring interactions.

The hierarchical agent-based monitoring approach is distinctive as being scalable and implementation-flexible for any-sized NoCs. The distributed cell agents, as physically adjacent to the functional units and exclusively responsible for local monitoring, can provide fast and fine-grained monitoring services to local circuits. The cluster agents are exclusively responsible for their own clusters, thus cluster-level monitoring is still low-latent and requires limited amount of processing capacity. The platform agent, though monitoring the whole system, only handles the resources at a coarse granularity. For instance, in terms of fault-tolerance, only errors which can not be fixed by low-level agents are reported to and handled by the platform agent. In this manner, no communication or processing bottleneck will appear in the large-scale platforms. The supervision of higher level agents over lower-level ones ensures the optimal overall system performance. Hierarchical monitoring approach also provides a wide design and synthesis space for implementing various management algorithms and circuits. Low-level circuit optimization methods, such as power or clock-gating can be implemented as dedicated circuits triggered by cell agents. High-level component management methods, such as DVFS or ABB, can be enforced by cluster level agents. Low-level circuit optimization should be simple in terms of synthesis to offer fast operation with small overhead. High-level operations can require more processing power since they are typically much less frequent than low-level operations. As the highest-level monitor, the platform agent configures the system with optimal general settings, for instance, an appropriate network connection to reduce inter-cluster communication. Thanks to the monitoring agent hierarchy, various optimization methods can be implemented efficiently with different design and synthesis constraints.

5.2.3 Implementation of Agents

Figure 5.4 illustrates the mapping of hierarchical agents on a tile-based regular Mesh NoC structure. The size and number of clusters and the location of the cluster agents are all application and platform dependent choices. In this example, we divide the platform into four clusters, each of which is a 2×2 Mesh network. The cluster agents are placed on the lower right switches within the cluster. As mentioned earlier, the basic monitored units are cells which are comprised of a PE, Network Interface (NI), switch and the corresponding links. It is intuitive to allocate a cell agent for each cell by sharing the physical area of the PE. Other cell level monitoring units may be allocated at other particular places within a cell, such as power gating sleep-transistors on the links. The cluster agent is allocated at a fixed position which is decided at the design time. Since a cluster agent has more sophisticated functionality and controlling algorithms than a cell agent, it requires more resources such as area, power, communication bandwidth, etc. Therefore, a cluster agent replaces one of the PEs in the cluster. To minimize the communication latency and balance the workload of the system, the platform agent and application agent which monitor and control over the whole system are located at the geographic center of the platform. In order to offer scalability for extremely large scale NoC systems, clusters can be further divided into hierarchical sub-clusters and similar monitoring functional partition can be applied.

5.3 Communication Scheme

Communication plays an increasingly important role in modern on-chip systems. In the proposed agent based architecture, besides computational data communication, the monitoring information as well as controlling signals need to be transferred among agents and processing units. Therefore, in this section, three different monitoring communication schemes are proposed, analysed and compared.

5.3.1 Monitoring Communication Interconnect Alternatives

Agents exchange monitoring information with their higher or lower level counterparts as illustrated in Figure 5.4. The monitoring communication needs to be reconfigurable so new cells can be incorporated to certain clusters at the run-time. Some conventional interconnection does not support reconfiguration (for instance, the star-like network as in Figure 5.5). In this chapter, we consider three interconnect alternatives which all support run-time reconfiguration but have different area, energy and latency overheads. Throughput is not a prioritized design constraint, since the monitor-

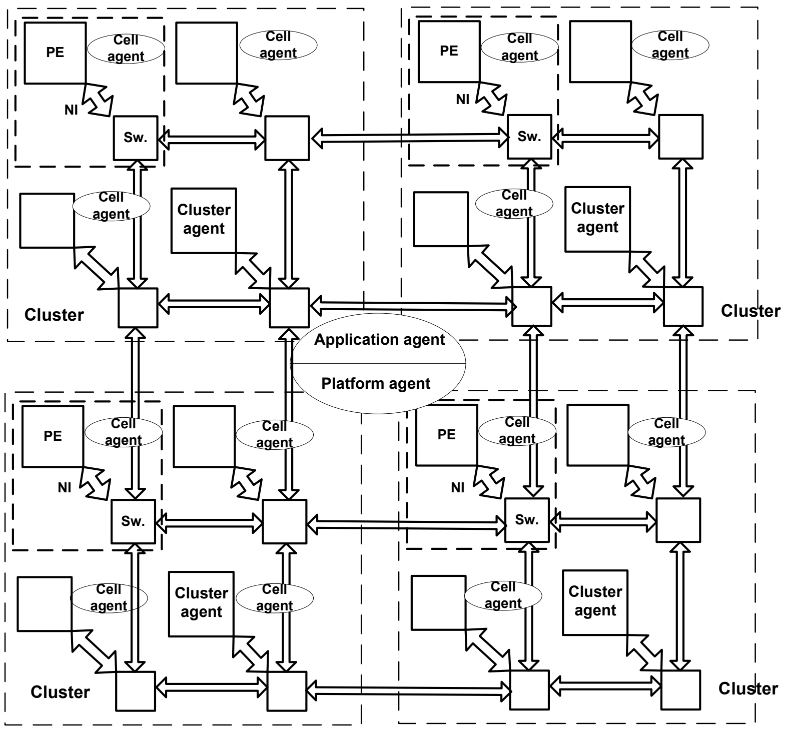


Figure 5.4: Hierarchical agent implementation on NoCs

ing communication is low in data volume ([19] reports 8% and 5% debugging monitoring traffic overhead for two streaming applications).

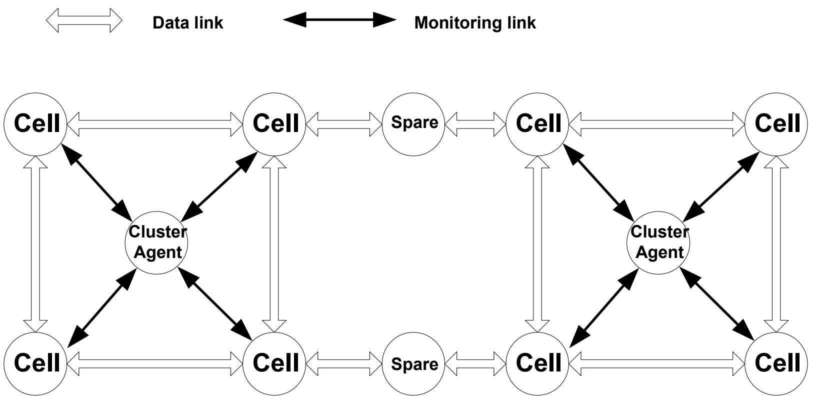


Figure 5.5: Star-like monitoring interconnect architecture

The first alternative is to realize monitoring communication as Time Division Multiplexing (TDM) based virtual channel upon existing links. This option not only incurs design complexity in virtual channel arbitration and

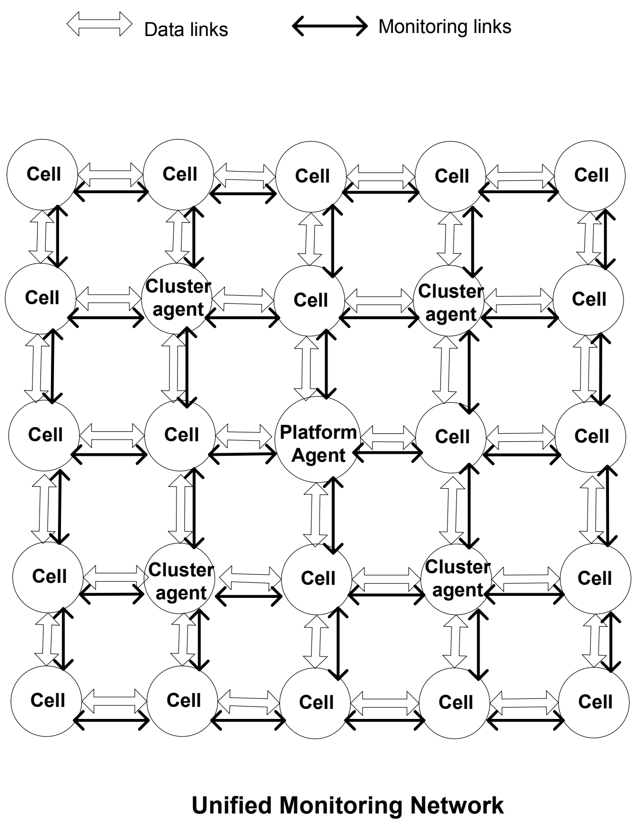


Figure 5.6: Unified dedicated monitoring interconnect architectures

allocation, but also increases the switch latency of both monitoring interconnect and data communication. The virtual channel arbitration and allocation also incur energy overhead. Wiring overhead, however, is kept to the minimum though the switch area is moderately increased.

The second alternative is to adopt a unified dedicated monitoring network for monitoring communication, as shown in Figure 5.6. It is called unified since monitoring communication between both cluster-cell agents and platform-cluster agents is transmitted on the same dedicated network. This option utilizes more wiring resources but simplifies the switch arbitration between data and monitoring communication, thus reducing the communication energy and latency.

The third alternative is to adopt separate dedicated monitoring networks for monitoring communication as shown in Figure 5.7. Comparing with the unified monitoring network, this option adds another network connecting the single platform agent to a small number of cluster agents. As a result, the communication between platform and the cluster agents is simplified with very limited wiring overhead.

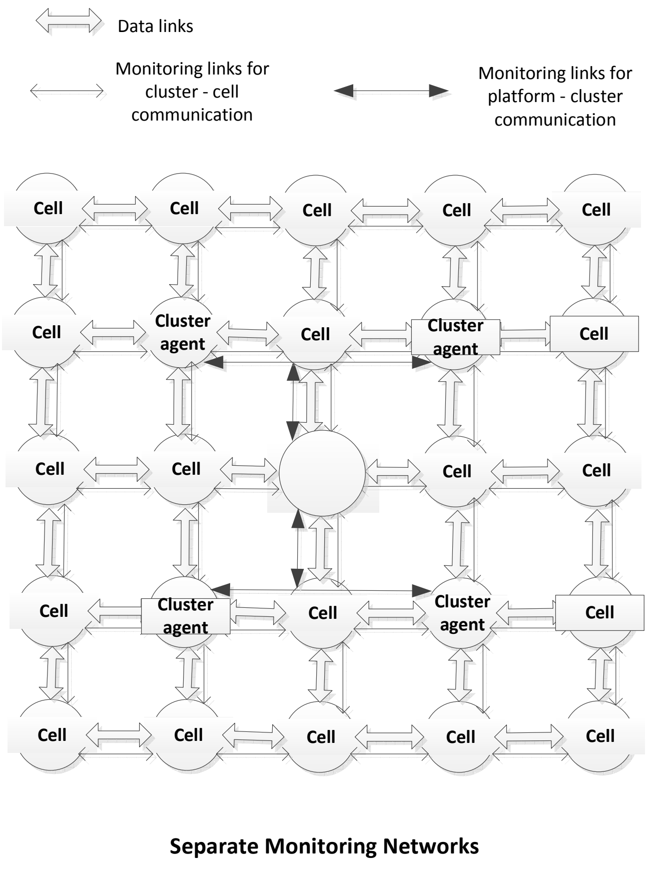

Data links
Monitoring links for cluster - cell communication
Monitoring links for platform - cluster communication
Cell
Cluster agent
Separate Monitoring Networks

Figure 5.7: Separate dedicated monitoring interconnect architectures

5.3.2 Performance Analysis

To quantitatively compare the implementation overhead of three monitoring interconnect architectures, we model a network similar to the TeraFLOPS processor in 65nm technology. The network has 8×8 processing elements mapped on a regular tile-based Mesh topology. We assume input-buffered pipelined switches with the structure suggested by [80] with matrix crossbar [103]. For TDM channels, each input buffer is 4-flit long while the unified separate network has 2-flit-long input buffer considering the higher traffic load of data communication. The other dedicated network for communication between cluster agents and the platform agent assumes no buffer since the traffic on this network is exclusive and infrequent. The arbitration assumes wormhole routing. NoC links are modelled as segmented wires with drivers and evenly inserted repeaters [1]. Data links are 32 bits wide and 2 mm long [2], and dedicated monitoring link is 8-bit wide and equally long. Figure 5.8 shows an example of how platform agent, cluster agents and cells (with cell agents) can be placed in a NoC based system where all clusters are of rectangular shapes and the platform agent has the same distance to the cluster agents. The whole NoC system is assumed to be mesochronous with network frequency as 1GHz and the supply voltage as 1V. When applying the agent based architecture to NoCs, it is important to notice that the number of clusters and placements of agents are design phase considerations and thus should not be restricted to the configuration in this example.

We estimate the area and energy overhead of switches by simulating with Orion [102]. The switch latency is estimated based on the results from [80]. The wires are modelled and simulated by Cadence. The Orion simulator does not produce result for 65nm technology directly, thus we apply scaling factors (based on [33]) to the result of 70nm technology simulation using Orion. The scaling factors for energy, area, and latency are 0.86, 0.86 and 0.93 respectively. The energy of wires are simulated by Cadence. The latency in the switch buffer assumes an average 50% occupancy ratio.

Latency

The latency is calculated in cycles considering the longest distances between the platform agent and a cluster agent and between a cluster agent to one of its cell agent. From Figure 5.8, we see that both distances are at maximum 4 hop counts with minimal routing. The wire latency is simulated to be 198ps, and each pipeline stage latency in switches is estimated to be lower than 300ps ([80], assuming an FO4 inverter delay to be 15ps in 65nm

[1] wire width: 210nm; spacing: 210nm; repeater interval: 0.25mm; repeater size: 10x minimal inverter size; driver size: 12x.

[2] TeraFLOPS uses 2mm $\times$ 1.5 mm tiles, while we simplify the tiles to be 2 mm $\times$ 2 mm squares

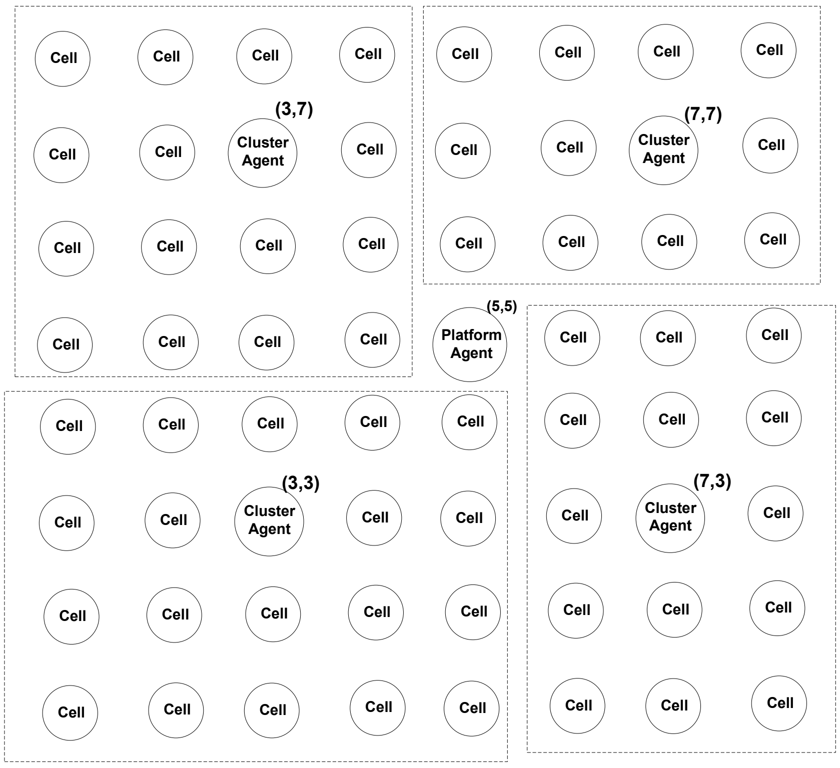


Figure 5.8: Locations of platform, cluster and cell agents in the experimental platform (with initial cluster boundary labelled)

technology). With 1GHz frequency, each link and one router pipeline stage (virtual channel allocation, routing and decoding, crossbar traversal) take 1 cycle delay. Table 5.1 summarizes the latency comparison for monitoring communication in each interconnect architecture.

Table 5.1: Latency comparison of three monitoring interconnect architectures (network working at 1GHz)

Interconnect Architecture	Delay (cluster ↔ cell agents)	Delay (platform ↔ cluster agents)
TDM-based	24 cycles	24 cycles
Unified Dedicated Network	16 cycles	16 cycles
Separate Dedicated Networks	16 cycles	8 cycles

Energy Consumption

The energy is calculated by the amount of energy consumed by a 8-bit flit (as we assume dedicated monitoring networks are 8-bit wide) traversing on the longest paths between the platform agent and a cluster agent, and between a cluster agent to one of its cell agent (4 hop counts as in Figure 5.8 with no misrouting). Table 5.2 summarizes the energy consumption for monitoring communication in each interconnect architecture.

Table 5.2: One-flit monitoring communication energy of three monitoring interconnect architectures (network working at 1GHz)

Interconnect Architecture	Energy (cluster ↔ cell agents)	Energy (platform ↔ cluster agents)
TDM-based	12.92 pJ	12.92 pJ
Unified Dedicated Network	5.40 pJ	5.40 pJ
Separate Dedicated Networks	5.40 pJ	2.31 pJ

Area

We analysed the total wiring and switch area for each interconnect architecture with 65 nm technology. Having more dedicated links involved, it is to our expectation that the separate dedicated network will consume more chip area. To illustrate the geographical impact of these interconnects, the area costs are then calculated as percentages of a TeraFLOPS chip ($275mm^2$). Table 5.3 shows the results of analyses.

Table 5.3: Area overhead of three monitoring interconnect architectures

Interconnect Architecture	Area (mm^2)	Percentage (of a chip area)
TDM-based	7.44	2.71
Unified Dedicated Network	8.95	3.26
Separate Dedicated Networks	9.11	3.32

5.3.3 Design Trade-offs

The figures show that separate dedicated monitoring networks are the most energy-efficient and low-latency interconnection for monitoring communication. Compared to TDM-based interconnection, it reduces the latency by 66.7% and energy consumption 82.1% for the communication between the platform and cluster agents, while achieving the same latency and energy efficiency as unified dedicated network for the communication between the cluster and cell agents. However there is area penalty involved: the area overhead is increased from 2.71% to 3.32%. Nonetheless the wiring area overhead has become less of a design constraint as multi-layer fabrication process provides quite abundant wiring potential for on-chip systems [106]; TILE64 processors incorporate 5 physically separate networks, each of them being 64-bit wide). As transistor feature sizes continue to decrease in the foreseeable future, the separate monitoring networks will provide the most optimal trade-off exploiting the on-chip wiring resources while minimizing the more critical power consumption and global interconnect latency.

5.4 Fault and Variation Tolerance

To reduce the energy consumption of electronic systems, techniques such as shrinking of the feature size and decreasing of the supply voltage are often used which, however at the same time, affect the reliability of NoC based systems by exposing them to different faults of permanent, transient or intermittent nature. Short circuit, for example, is more commonly found in circuits with small feature size, which can cause not only performance degradation but also excessive power consumption.

Among the failure mechanisms, we can enumerate factors such as crosstalk, electro migration, electromagnetic interference, alpha particle hits, and cosmic radiation [31]. These phenomena and system variations can change the timing and functionalities of the NoC fabrics and thus degrade their QoS or, eventually, lead to failures of the whole NoC-based system. Providing resilience from such faults and other PVT (process, voltage, temperature) variations is mandatory for the NoCs.

5.4.1 Overview

The proposed monitoring agent based architecture can provide the fault or variation tolerance for NoC based systems, by the joint effort of all levels of agents.

This hierarchical approach provides efficient and flexible diagnostic services against most of the failures and errors. For permanent faults such as malfunctioning processors or failing links, the corresponding agent will report the failure together with its type to the upper level agent. The latter will then allocate new resources or reconfigure the network to the best of its authority. If an agent itself is faulty, the fault will be detected when it does not send any performance or fault report to its higher level agent within a predefined due time. Transient failures, as aforementioned, will be handled by local agent with best efforts. An example of how the proposed approach helps NoC based systems to recovers from faults is shown in Section 5.4.2.

Besides failures, performance degradations can also be detected and handled by the proposed agent architecture. Each level of agent traces the performance and reports it to the higher level agent. Corresponding actions will be taken when the monitored performance is not satisfactory or beyond the predefined range. Typical actions include replacing the poorly functioning processors, adding more processing elements, speeding up the circuits by increasing supply voltage or frequency, etc. In an extreme situation where the whole system cannot any more meet the performance requirement by all means, the application agent will then interact with external system or operator to acquire more resources or even modify the application specification.

5.4.2 Case Study: 64-point FFT/IFFT

The Fourier Transform is one of the most important mathematical transforms with many applications for system analysis, communication technologies, signal processing, etc. It expresses a mathematical function of time as a function of frequency, known as its frequency spectrum. The most common and efficient Fourier Transform algorithm for digital signals is Fast Fourier Transform (FFT) which is extensively used in multimedia and wireless communication chips. The Inverse Fast Fourier Transform (IFFT), on the other hand, inverts the FFT process and turns the frequency back to time representative of signals. Both FFT and IFFT can exploit multiple levels of parallelism and pipelining, and thus are suitable for NoC based multi-core systems.

Figure 5.9 shows a case study where we focus on the functions of agent hierarchy to flexibly provide the trade-off on a fault-tolerant NoC platform with a pool of DSP processors divided into clusters. A simplified IEEE 802.11a MIMO-OFDM system FFT-component is mapped with different FFT architectures to illustrate how faults are tolerated and performance requirements can be met. The number of functioning cells and mapping are application dependent choices. In order to replace the faulty nodes, redundant hardware (in this example, 30% of the cells) is initially left as spare cells on the NoC based system. Reconfiguring faulty NoC simply needs to move processes from faulty cells to functioning spare ones. The spare cell pattern in this study is designed for the easy reconfiguration, as every functioning cell has a spare cell in its neighbourhood.

We assume that all processing elements in the platform are DSPs operating at 600 MHz clock frequency. The complex multiplications of the 64-point FFT/IFFT computation are assumed to take 6 clock cycles. Another assumption is that the on-chip communication is not limited by link bandwidth during the run-time. The data samples in this case study are 16 bit wide. Two architectural alternatives are used in mapping the FFT/IFFT. The first architecture is "A novel pipelined FFT architecture for double-rate FFT" illustrated in [57], which requires less computational resources. The other alternative is "R22SDF" proposed in [35] which explores more parallelism (thus finishing faster) while using more processors. The case study is simulated by Matlab/SimuLink.

As shown in Figure 5.9, there are three final phases of configuration. The top most Figure 5.9 (a) shows that when the system is configured with a double-rate 64-point FFT/IFFT architecture ([57]), the computation takes 6 processors and 8 μs. The platform agent can (re)configure the computational resources when needed. If some components fail, the platform agent will replace them with spare processors and reconfigure the network. As shown in Figure 5.9 (b), two faulty nodes have been detected and a new

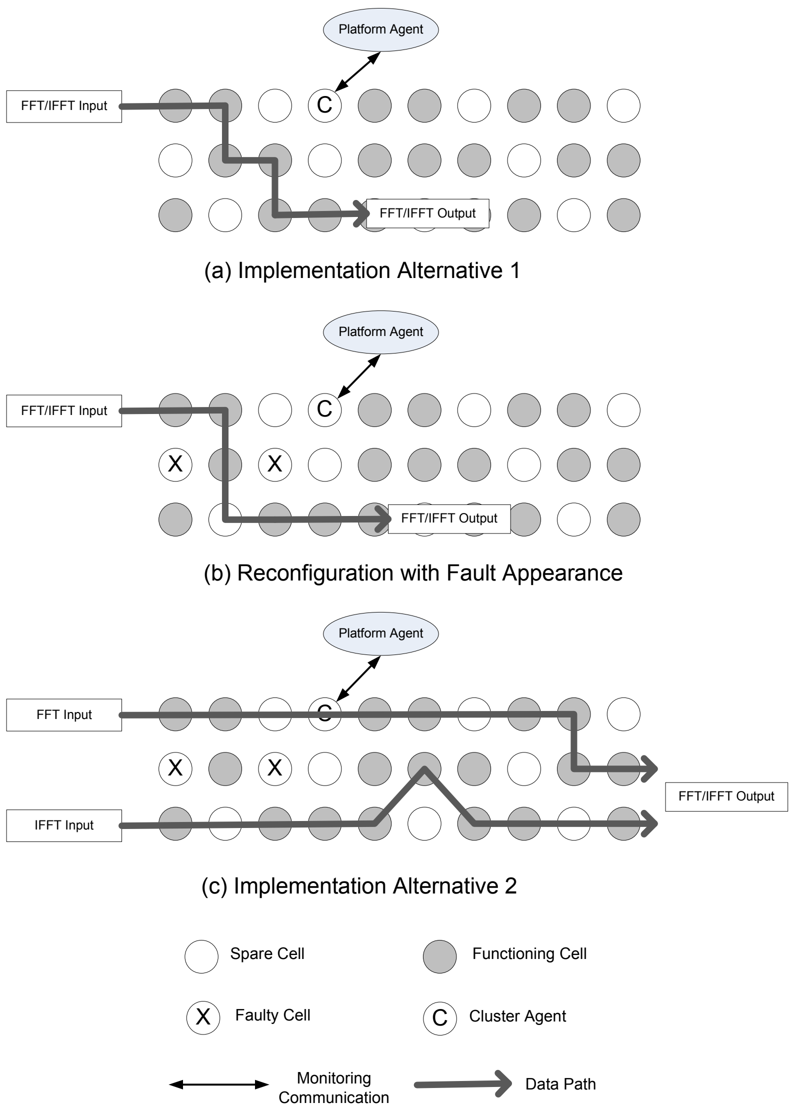


Figure 5.9: Study case on fault/variation tolerance with hierarchical agents

data path is then configured to bypass them by utilizing one of the spare nodes. Another case is that when the application agent specifies tougher timing constraints, the platform agent may utilize more available resources to achieve another performance/cost trade-off. Figure 5.9 (c) describes the performance enhancement after network remapping with architecture alternative as in [35]. The platform agent now allocates another 10 processors and in return, the computation time is reduced to from 8 to 3 μs (spare ones in the data flow are only bypassed, not used in computation).

5.5 Chapter Summary

In order to achieve high energy efficiency, a hierarchical agent monitoring architecture with desirable scalability and design flexibility for future large scale NoC systems is proposed in this chapter. With an extra monitoring layer comprised of four levels of agents, the system is potentially able to achieve maximized efficiency with on-line monitoring services. This chapter also elaborately explains the hierarchical monitoring approaches enabled by the interactions of all levels of agents, and examines the design alternatives for energy optimization of different granularities as an example of flexible functional partitions among agent levels. Quantitative analysis for agent interconnection alternatives suggests reasonable trade-offs between area, energy and latency overhead, and motivates separate dedicated monitoring networks for inter-agent communication.

Furthermore, by using a case study of 64-bit FFT/IFFT implementation, we have shown that the proposed architecture is capable of providing power and variation tolerance which not only enhance the system reliability, but also improve the energy efficiency. This work demonstrates the potential and feasibility of multi-level on-line monitoring layer upon the overwhelming amount of on-chip resources, which provides a great diversity of design options in a scalable manner.

Chapter 6

Dynamic Voltage and Frequency Scaling

To achieve maximum energy efficiency, run-time power management with various Dynamic Voltage and Frequency Scaling (DVFS) techniques is analysed on the hierarchical agent based NoC platform. Conventional power monitoring techniques can be flexibly incorporated into the functions of specific levels of agents. Network condition is observed by the agents at the run-time, and the power supplies are adjusted at different granularity accordingly. This chapter presents the system architectures of two adaptive schemes, together with efficient and feasible algorithms.

6.1 Run-time Power Optimization

As presented in the previous chapter, the three phases in the hierarchical agent based NoC systems are also three energy consumers, namely, computation, communication and agent optimization. The energy overhead of the agents can be minimized by using the proposed communication and implementation architecture. Although both computation and communication energy can be reduced with the agent based design approach, in this thesis, we focus mainly on the communication architecture due to its rising importance.

Considering the dynamic nature of the network environment, run-time power optimization techniques are most favourable in optimizing the energy consumption. For energy constrained systems, a wide range of efforts have been made to minimize the energy consumption under different QoS requirements, with average packet latency as one of the essential metrics for best effort service. DVFS is one of the most widely used methods. The key idea behind DVFS technique is to dynamically scale the supply voltage and frequency of the processing cores to provide just-enough system performance to process the system workload while meeting the time and throughput constrains, and thereby, reducing the power consumption [17].

In the proposed design approach, power monitoring schemes utilize dynamic energy-performance trade-off, based on the observation and adjustment of network load, quantified by the percentage of buffers occupied in interconnections. Initially proposed in [56], the buffer load of each node is collected by controllers, which configure proper level of voltage and frequency based on the network condition. Based on the assumptions of using 65 nm technology and each direction has two-it (each as 64-bit wide) input buers for each virtual network, Figure 6.1 illustrates the relation between latency and buffer load in an 8×8 network with store-and-forward switching and deterministic routing. Packets are injected into the network using uniform random pattern. With lower frequency and voltage, the latency gradually increases until the network saturates. After that, the latency becomes unboudedly high due to the accumulation of queuing time in the buffers. Therefore, although the supply voltage is lower, the total energy consumption will increase dramatically because all the resources are fully loaded with network traffic. Based on this observation, the system can be configured to adaptively trade-off performance and energy by setting a desirable traffic load.

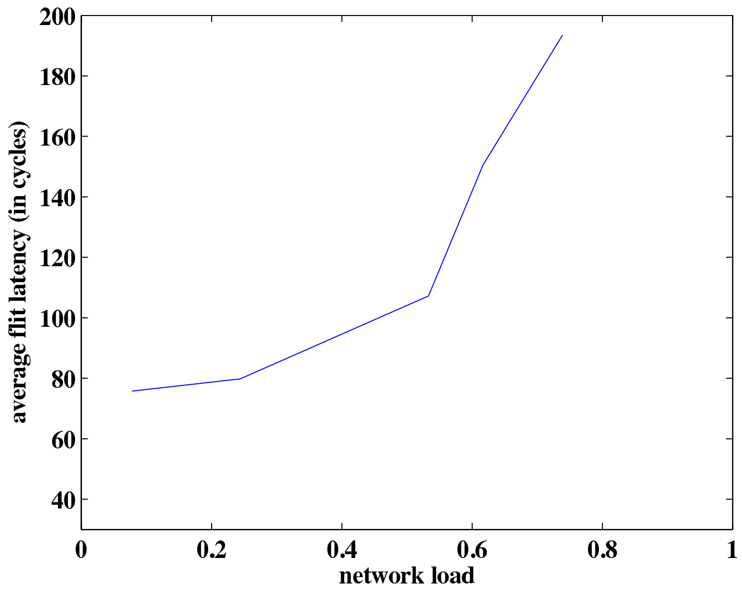


Figure 6.1: Adjustment of transmission latency by observing traffic load (8×8 mesh network, uniform traffic)

In hierarchical agent-based system, specific level of agents will take the responsibility to perform the energy-performance trade-off. The exact algorithm with the desirable traffic load is determined by the designer and programmed into the function of corresponding agents. Such settings can be reconfigured as required by the application or platform agent.

6.2 Island based DVFS

Voltage and frequency island based approach is a coarse-grained architectural method to improve energy efficiency by exploiting the spatial variation of workloads. Combined with DVFS, island-based architecture addresses both the spatial and temporal variations of data communication in on-chip networks.

In hierarchical agent-based NoC, island-based DVFS can be integrated into the functional responsibility of cluster agents (Figure 6.2). The entire network is partitioned into a number of islands (clusters) and each of which works under its own voltage and frequency.

At run-time, a cluster agent collects the average network load in the cluster and adjust the cluster voltage and frequency accordingly. Each cluster is configured as a supply island, equipped with a DC converter and a PLL. The cluster-level DVFS is a scalable trade-off between single domain and per-core supply scaling mechanisms. Only one DC converter and one PLL are required for a whole cluster. FIFOs are implemented to interface different voltage and frequency domains between two neighbouring islands.

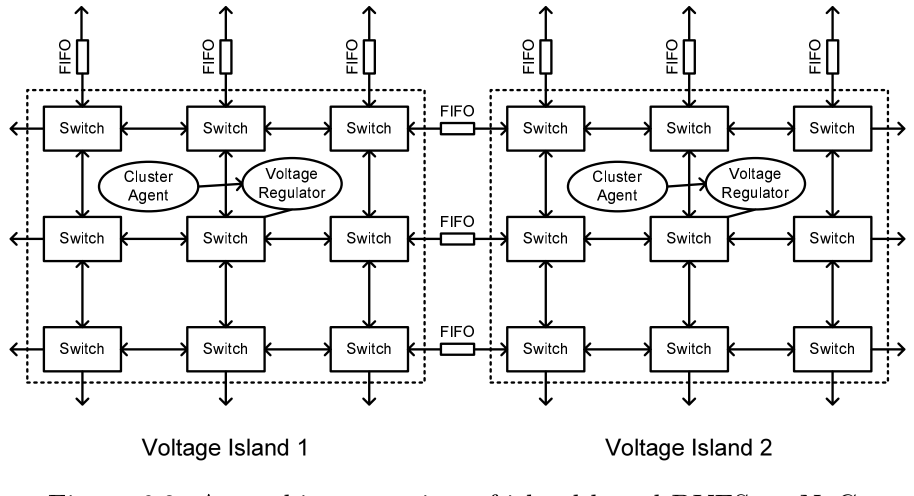


Figure 6.2: An architecture view of island based DVFS on NoCs

6.2.1 Synchronizing FIFO

FIFOs are widely used in multi-clock systems for signal synchronization. Each domain is synchronous to its own clock signal but can be asynchronous with others in either clock frequency or phase [40]. The challenges of designing FIFOs include the enhancement of reliability, reducing latency as well as power and area costs.

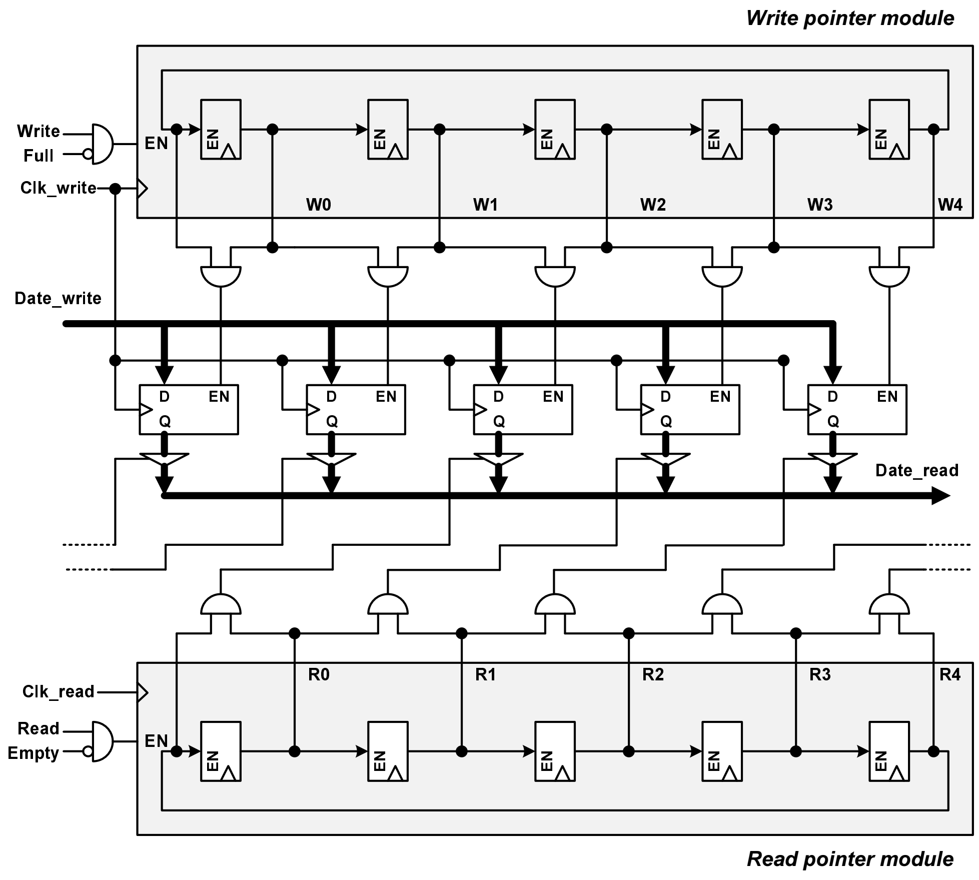


Figure 6.3: Implementations of write pointer, read pointer and data buffer [40]

In this work, the FIFO presented in [40] is used because it is synthesizable in synchronous standard cells and has scalable architecture. Moreover, this bisynchronous FIFO is capable to interface systems working with different clock signals (frequency and/or phase). Robustness against metastability is improved by a proposed mesochronous adaptation which allows switching between mesochronous and asynchronous modes. A token ring based novel encoding algorithm is utilized combined with an astute definition of the FIFO pointers that avoids the utilization of status registers. The implementations of write pointer, read pointer and data buffer are shown in Figure 6.3. Table 6.1 lists the input and output signals as well as their functionalities. Since most of the FIFO's power consumption is due to data buffers, power consumed in the other components is ignored with acceptable accuracy. The FIFO's power consumption is simulated with Orion simulator.

Table 6.1: Description of FIFO signals

Signal	Description
Write	Write request
Data_write	Data to be written in the FIFO
Full	Signal to indicate the FIFO is full and no more data can be received
Clk_write	Clock signal in the write domain
Read	Read request
Data_read	Data to be read from the FIFO
Empty	Signal to indicate the FIFO is empty and hence no data can be read
Clk_read	Clock signal in the read domain

6.2.2 Island based DVFS Algorithm

For island-based DVFS approaches, each cluster agent observes buffer loads of all cells in the cluster, which can be calculated as the percentage of buffers occupied in the switches and the relevant outgoing FIFOs (averaged in an history window of 100 cycles). If the cluster-level load is above a threshold, the higher level of voltage and frequency will be configured to the cluster, otherwise the lower level is applied. Algorithm 1 illustrates this adaptation process. As the voltage and frequency of the whole cluster has to be configured simultaneously, any congestion in a local cell will require the high voltage and frequency to be applied. The threshold is determined at design-time, based on the network property and desirable latency overhead. The number of voltages and voltage levels are application dependent choices. In wireless sensor networks, for example, sub-threshold voltage levels are needed in order to provide optimal energy consumption. According to Figure 6.1, the threshold is set at 0.3 and 3 levels of voltage are applied in our experiments, with a reasonable margin before the latency starts increasing rapidly.

Algorithm 1: Adaptive island-based voltage and frequency scaling for on-chip communication

```
foreach Cluster do
    next_frequency= Low_frequency;
    next_voltage=Low_voltage;
    foreach Cell do
        if $Cluster\_load \geq Threshold$ then
            next_voltage = High_voltage;
            next_frequency=High_frequency;
            break;
        end
    end
end
```

6.3 Per-core based DVFS

Coarse-granular DVFS technique such as centralized or island-based approaches are not efficient in exploiting the local traffic pattern and flexibly group any node into a voltage domain. Therefore, per-core level DVFS architecture and implementation are studied in this thesis. Comparison of these approaches are presented in Section 6.4.

6.3.1 Constraints of DVFS using Voltage Regulators

The mainstream approach to achieve on-chip DVFS is to adjust the voltage and frequency via off-chip or on-chip regulators. Off-chip voltage regulator is the most commonly used technique ever since DVFS was proposed in the 1990s. However, since off-chip voltage regulators are low-level regulation circuitries which are implemented on the printed circuit board (PCB), the voltage changes are performed with high time overhead which is on the order of several microseconds [50]. Moreover, off-chip regulators are area consuming. As reported in [52], the area cost of Linear Technology's LTC3713 low input DC-DC converter which is a high-efficiency power regulation circuit is approximately 718 mm^2. Hence, due to the time and area constraints, it is infeasible to implement multiple power domains using off-chip voltage regulators. To alleviate these challenges, on-chip voltage regulator technique is proposed in the last decade. However, the implementation of DVFS using regulators in general still suffers from two main constraints.

The first constraint comes from the overhead. The voltage regulator, even the on-chip regulator, results in not only area but also power overhead. An efficient implementation of on-chip voltage converter in 180 nm technology is presented in [1]. When scaled into 65 nm technology, assuming that per-core DVFS is used, the area cost of the voltage regulators will be over 25% of the total chip size according to TeraFLOPS 80-core NoC [99]. Besides, regulators also consume power when changing between different voltage levels. When voltage switchings are performed frequently, the power consumed by the switchings may compensate a considerable amount of, if not exceed, the power saved by using DVFS. Moreover, the number of cores on NoCs increases rapidly with the development of technology. In 2008, a NoC based system with 167 cores has been proposed and implemented as shown in [98]. In systems with a large number of cores, DVFS has to be applied at finer granularities to optimize the power consumption. But the area and power overhead restricts the number of regulators in the system and thus prevents the fine-grained control.

The other main constraint is that DVFS using regulators requires additional design effort to be robust. Although the voltage transition time in the state-of-the-art on-chip regulator has been reduced greatly with respect to the off-chip alternative, it still takes tens to a hundred of nanosecond for each voltage transition [50]. During this time, the voltage is gradually scaled to a new level. It is a critical requirement that the system ensures each relevant circuit works correctly under the varying supply voltages.

6.3.2 DVFS Using Multiple Voltage Supply Networks

To overcome the aforementioned constrains and provide fine-grained optimization, we propose a multiple power network architecture to be implemented on every router in the system. Figure 6.4 shows an paradigm of the proposed architecture where three supply voltages are used. The basic unit in the architecture, as shown in Figure 6.4, contains a router, its attached communication links and the voltage supply circuitry. A router is connected to three voltage supply wires via *supply selecting transistors*. The three supply selecting transistors, N_L, N_N and N_H, are controlled by their own controlling signal C_L, C_N and C_H, respectively. The three voltage levels are low voltage Vdd_L, normal voltage Vdd_N and high voltage Vdd_H. The default supply voltages for all units are Vdd_N. When the workload of a particular router increases, its supply voltage can be adapted to a higher level, and vice versa.

Comparing to the DVFS using voltage regulators, the multiple power network architecture provides less flexibility in the supply voltages. This is because of the design complexity of implementing a large number of on-chip power networks. However, the advantages of the proposed approach are obvious. Firstly, comparing with the NoCs using voltage regulators, the area overhead in our approach is significantly lower. The only area costs are caused by the additional power supply wires and supply selecting transistors. Secondly, the voltage transition time is orders of magnitude smaller by using

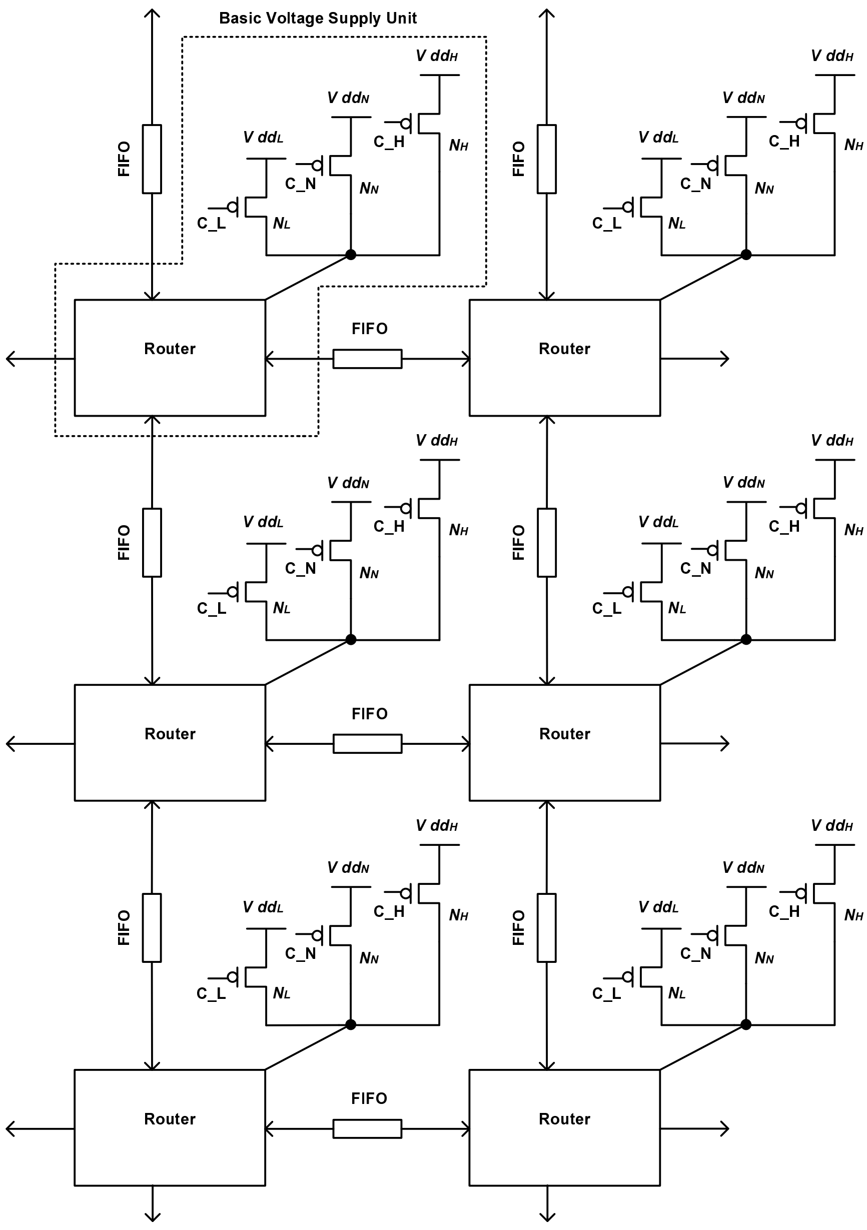


Figure 6.4: Architecture of NoC based two voltage supplies network

the proposed approach. When a router needs to adapt its supply voltage, it firstly change the selecting vector values which change the connection to different power networks and thus voltage shifts can be achieved more quickly. Furthermore, it is much easier and more energy efficient to adapt the supply voltages for the routers in our approach than in those with voltage regulators.

In our approach, a bit vector is used to select the voltage supply for a router. The size of the vector is determined by the number of supply voltages. Assuming there are N different supply voltages, the minimal size of the selection vector should be $\lceil \log_2 N \rceil$. If a system has only two different supply voltages, only one bit is needed for the voltage selection. Table 6.2 shows a paradigm where there are three levels of voltage supplies (like the architecture in Figure 6.4). Different combinations of the bits in the vector stand for different voltage levels of the routers. The adaptation phase can be done either in hardware or in software. Each router is controlled independently by its own adaptation bits and thus very fine grained optimization can be achieved.

Table 6.2: Controlling the voltage supplies

Selection Vector	C_L	C_N	C_H	Voltage Supply
"00"	0	1	1	V_L
"01"	1	0	1	V_N
"10"	1	1	0	V_H
"11"	1	1	1	*Power Gating*

6.3.3 System Architecture of Per-Core DVFS

The architecture of the proposed per-core based DVFS NoC is shown in Figure 6.4. Since the routers are able to run in different frequencies, inter-router instead of inter-island synchronization is needed for the communication in this scenario. Therefore, a FIFO with application dependent pre-defined size is implemented on every communication link among the routers in the entire NoC.

6.3.4 Per-core level DVFS Algorithm

In per-core DVFS approaches, each cell agent observes the local buffer load, which are quantified in the same way as that in island-based DVFS. As shown in Algorithm 2, if the local load is above a threshold, the higher level of voltage and frequency will be configured to the cell, otherwise the lower level is applied. The threshold buffer load is still set at 0.3.

6.4 Quantitative Evaluation

In this section, we evaluate the effectiveness of applying island-based and per-core DVFS on power management of on-chip communications. For reference purpose, we also examine the basic NoC architecture without DVFS

Algorithm 2: Adaptive per-core voltage and frequency scaling for on-chip communication

```
foreach Cell do
    Local_load = average(switch_buffer_load, northFIFO_load,
    southFIFO_load, westFIFO_load, eastFIFO_load) ;
    if $Local\_load \geq Threshold$ then
        next_voltage = High_voltage;
        next_frequency=High_frequency;
    else
        next_frequency= Low_frequency;
        next_voltage=Low_voltage;
    end
end
```

support, where the voltage and frequency are set to the maximum available, to ensure the proper transmission in any situation. Average energy consumption and latency of transmission using various traffic patterns are analysed, in order to show the benefits and trade-off of such techniques applied at different granularity and handled by different levels of agents.

6.4.1 Experiment Setup

Network Model

The experiments are simulated on an 8×8 2-D Mesh direct network in 65 *nm* technology. The switches are connected with two uni-directional 1mm-long wires (Table 6.3), as suggested in [32]. The packet is routed with a minimal-path adaptive routing method with four disjoint virtual networks. We assume simple single-flit store-and-forward switching , and each direction has two-flit input buffering for each virtual network (Figure 6.5).

Table 6.3: Parameters used in wire models (65nm technology)

parameter	value
length	1mm
width	210nm
spacing	210nm
repeater interval	0.25mm
repeater size	10x
driver size	12x

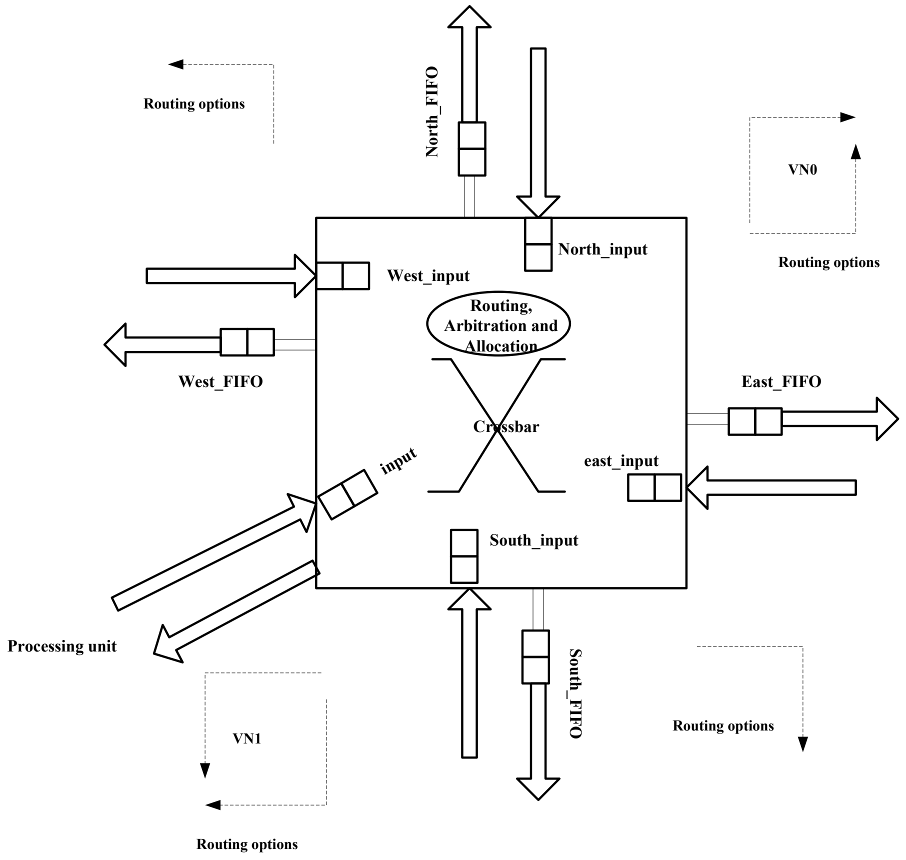


Figure 6.5: Internal structure of a switch

Energy Modelling

The voltage and frequency scaling are calculated following the alpha-power model (Equation 6.1) [85]

$$f \infty K * \frac{(V_{DD} - V_{th})^{\alpha}}{V_{DD}} \tag{6.1}$$

where f is the switching frequency, K is a fitting parameter and α is the velocity saturation index dependent on the technology. Before transistor-level implementation is available, we assume the base voltage and frequency are (0.6 V, 600MHz) as consistent with [50]. Based on Equation 6.1, the following pairs of voltage and frequency are used: (1.3V, 1.2GHz), (0.75V, 0.8GHz), (0.6V, 0.6GHz).

The switch energy and area overhead are obtained from Orion simulator specifying the switching voltage and frequency, as well as the switch structure. The link energy and overhead are simulated with Cadence as the transistor-level repeater insertion and driver sizing influence the results greatly in sub-100nm technology.

The energy consumed by voltage scaling follows Equation 6.2

$$Energy_{overhead} = C * (1 - \mu) * \left| V_{dd2}^2 - V_{dd1}^2 \right| \tag{6.2}$$

where C is the decoupling capacitance of the power-supply regulator on the circuit, μ is the conversion efficiency, V_{dd1} and V_{dd2} are the voltages before and after the switching. We assume C as 40 nf [50], and 80% as the conversion efficiency [1].

Traffic Model

Synthetic traces characterize communication traffic patterns, and are helpful in early-stage evaluation of architectural design [62].

We categorize the experimental traffics based on three features: the injection temporal rate from each processing element, the spatial variation of injection rates between processing elements, and the destination pattern of communication. In terms of injection temporal rates, we consider uniform traffic and temporally varying traffic. For the latter case, the injection rate increases during the simulation. In terms of injection spatial variation, uniform and hotspot injection patterns are considered. Uniform pattern assumes all the processing elements have the same injection rate. Hotspot injection pattern chooses a few processing elements with higher injection rate than others. In terms of packet destination pattern, locality and uniform traffics are simulated. Locality traffic [62] makes a reasonable assumption that the closer destinations have higher probability:

$$Prob(d) = \frac{1}{A(D) * 2^d} \tag{6.3}$$

where A(D) is the normalizing factor that makes sure the probabilities sum up to 1 and D is the maximum distance in the network. Locality traffic simulates a situation in which the resource mapping algorithm assigns commonly communicating processors as adjacent neighbours. Uniform destination traffic assigns each node in the network with an equal probability as the destination. The combination of different settings of the three features leads to 8 types of traffic traces (Table 6.4).

[1] the efficiency is dependent on the output voltage and the regulator's activity factor. 80% is an average figure reported in [50]

Table 6.4: Synthetic network traffic traces for experiments

Index	Type
1	temporally uniform, spatially uniform, destination uniform
2	temporally uniform, spatially uniform, destination locality
3	temporally uniform, spatially hotspot, destination uniform
4	temporally uniform, spatially hotspot, destination locality
5	temporally varying, spatially uniform, destination uniform
6	temporally varying, spatially uniform, destination locality
7	temporally varying, spatially hotspot, destination uniform
8	temporally varying, spatially hotspot, destination locality

6.4.2 Simulation Results

Average Energy Consumption

We measured the average energy consumption per flit (64-bit) of the eight traces (Table 6.4) running on the aforementioned three architectures with settings described in Section 6.4.1, and the results are summarized in Table 6.5 (normalized with the energy of the basic architecture without any power management scheme).

Table 6.5: Normalized average energy consumption in three architectures

Trace	Basic	Island DVFS	Per-core DVFS
1	100%	81.8%	74.7%
2	100%	57.5%	50.5%
3	100%	55.2%	48.9%
4	100%	65.4%	56.9%
5	100%	70.4%	64.1%
6	100%	51.4%	54.9%
7	100%	60.2%	53.1%
8	100%	38.3%	35.7%

One major power overhead of using DVFS schemes is the FIFO access between voltage and frequency domains, as summarized in Table 6.6.

Table 6.6: Energy overhead due to FIFO access

Trace	Island DVFS	Per-core DVFS
1	3.4%	17.1%
2	2.5%	16.6%
3	3.3%	16.2%
4	2.7%	16.5%
5	3.2%	16.9%
6	2.4%	16.7%
7	3.3%	16.5%
8	2.3%	15.5%

Average Transmission Latency

Contributed by the frequency scaling and FIFO overhead, the average transmission latency (per-flit) is moderately increased using the two DVFS schemes, comparing with the basic architecture with all-time high frequency (Table 6.7).

Table 6.7: Normalized average transmission latency in three architectures

Trace	Basic	Island DVFS	Per-core DVFS
1	1	1.42	1.65
2	1	1.61	1.78
3	1	1.72	1.9
4	1	1.55	1.73
5	1	1.53	1.76
6	1	1.68	1.71
7	1	1.61	1.85
8	1	1.81	1.89

6.4.3 Experiment Analysis

The quantitative evaluations demonstrate that applying power management on different levels of granularity effectively reduces power consumption, and the trade-off varies depending on the specific level of monitoring.

Comparing with the basic architecture, both DVFS schemes achieve considerable energy benefit (25.3%-64.3% lower), especially in traffic traces with temporal and spatial variations (Traces 2-8). Island-based DVFS utilizes a coarse-granularity of monitoring, and FIFO overhead is minimal (2.3%-3.4%). With a finer-granularity, per-core DVFS achieves higher energy efficiency as its adjustment can be targeted on specific cells. Especially for traffics with high spatial variations (for instance Traces 2,4,7), per-core DVFS consumes much less energy compared with the island-based scheme. As there exist FIFOs on each link for per-core DVFS, a higher percentage of FIFO overhead is incurred (15.5%-17.1%).

As a natural consequence of frequency scaling, both DVFS schemes witness latency penalty, which is also contributed by FIFO delay. However, such a penalty is predictable since the frequency levels are determined at the design time. The applied trade-off scheme avoids local congestion by putting an upper bound on the traffic load, which ensures that no long time queuing occurs (given fair arbitration of flits). As a result, the transmission latency overhead is bounded and moderate. Per-core DVFS leads to longer average latency because of the fine-grained FIFO insertion, as a penalty for its higher energy efficiency compared with island-based DVFS.

The trade-off prioritizing energy efficiency with bounded performance penalty is targeted at energy-critical platforms. And setting a proper buffer load boundary at the design time can fine-tune such trade-off, addressing specific transmission requirements.

6.5 Routing Algorithms

In NoC based systems which support DVFS and power gating technologies, conventional routing algorithms such as XY routing may result in severe performance degradation or even faults. Take the per-core level DVFS shown in Figure 6.6 as an example. Assuming data packets are sent from Router 1 to Router 4 using XY routing, packet delivery will be delayed when Router 2 is switched to a lower voltage and frequency. Worse still, the packets might even be lost when Router 2 is power gated.

6.5.1 Q-Learning based Routing Algorithm

To solve this problem, we propose a reinforcement routing algorithm based on Q-learning technique. Let $Q_x(d, y)$ be the time that a node x estimates it takes to deliver a packet P to the destination node d via x's neighbour node y, including the time that P would spend in node x's queue. Upon sending P to y, x immediately receives estimation from y for the time remaining in the trip, namely

$$t_y = Q_y(d, minQ_y(d)) \tag{6.4}$$

Each node x in the system maintains a two dimensional table storing the

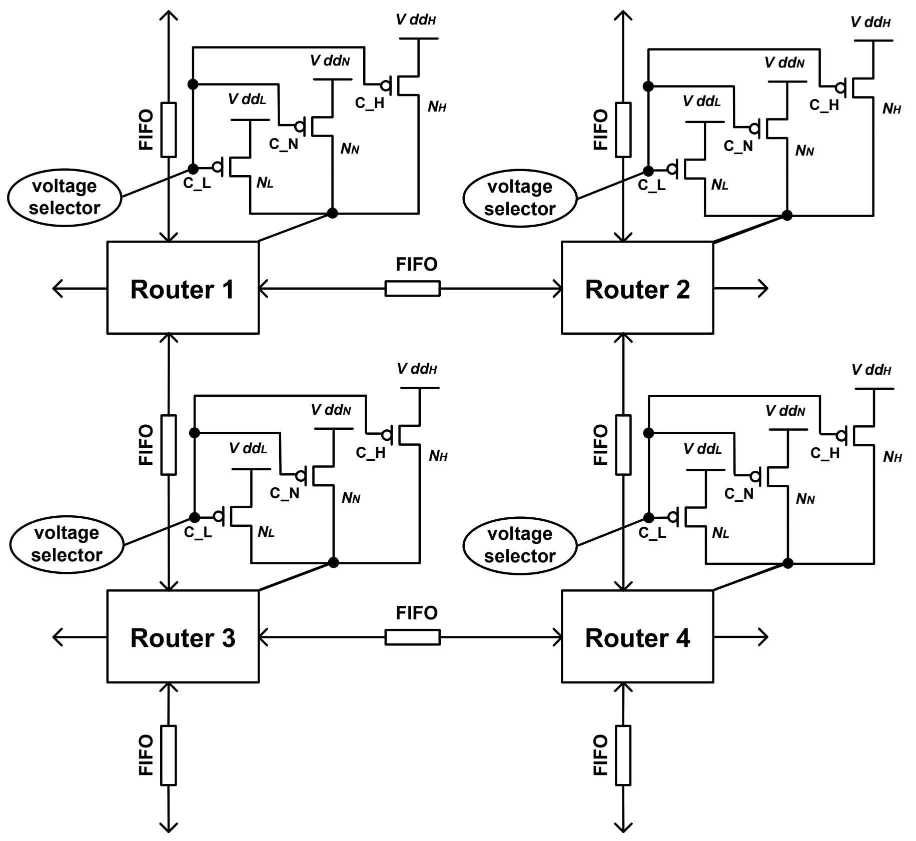


Figure 6.6: Routing example for per-core level DVFS

value of $Qx(d, y)$ for all the possible destination nodes via its neighbouring nodes. When delivering packet P to the destination d, node x firstly checks the Q-values via its neighbours and then route P to the direction of the smallest Q-value.

The proposed reinforcement routing policy for per-core granular DVFS is shown in Algorithm 3. When node y receives the power gating signal, it immediately sends notification to all its neighbouring nodes, setting their Q-values via node y to be infinity. Therefore, delivery failures can be avoided by not routing via the power gated node y. Similarly, when a power gated node y receives the power resuming signal, it again sends notification to all its neighbouring nodes, setting the corresponding Q-values to the initial state. In DVFS scenarios, node y notifies its neighbours about the potential changes of the queue time based on its most recent record q_y.

Algorithm 3 can be modified slightly to support island-based DVFS techniques. Instead of sending notifications to all neighbouring nodes, node y will only notify its neighbours which are not in the same voltage island. By doing so, less notification signals will be sent and thus power and traffic overhead due to voltage switching can be reduced.

For regular communication scenarios without power gating or DVFS, the proposed routing algorithm can also be adapted to update the Q-table, in order to have more accurate network status and efficient routing, as shown in Algorithm 4. However, this topic is beyond the discussion of this thesis.

6.5.2 Deadlock Handling Mechanism

NoC based systems are susceptible to deadlocks, which often lead to performance degradation or even system failure. Therefore, it is very important to resolve the deadlocks as soon as possible. According to [25], there are two strategies to deal with deadlocks in NoCs, namely, deadlock avoidance and deadlock recovery.

In the deadlock avoidance approach, communication resources such as routers and links are allocated according to certain rules so that the whole network is deadlock free. This is usually achieved by either prohibiting some of the turns (turn model based algorithms) or strictly ordering of the virtual channels [111]. Thus, deadlock avoidance techniques generally incur restricted routing functions or additional network resources. Furthermore, for complicated network scenarios, such as NoCs with DVFS and power-gating support, the complexity of deadlock avoidance techniques increase rapidly.

Deadlock recovery, instead of avoiding deadlocks from happening, allows deadlock to occur after which the system will be autonomously recovered without external intervene. Deadlock detection, in this approach, plays an important role determining how accurate and fast can the deadlocks be discovered and treated. However, efficient deadlock detection is challenging due to the distributed nature of deadlocks.

The most commonly used deadlock detection techniques are the heuristic approaches such as time-out mechanisms. These approaches monitor the activities on each link for deadlock speculations, but suffer from the accuracy challenges since it is difficult to determine the best threshold value. A deadlock detection method is proposed in [5] that utilizes run-time Transitive Closure (TC) computation to discover the existence of deadlock-equivalence sets, which imply loops of requests in NoC based systems using fully adaptive routing algorithms. This approach is employed in this thesis for the following reasons.

Firstly, as elaborated in the paper, by using a run-time TC computation scheme, the existing deadlock-equivalence sets can be discovered accurately and efficiently in NoC based systems. Comparing with many other approaches, this mechanism guarantees the true deadlock detection without false alarms.

Algorithm 3: Reinforcement Routing Algorithm for NoC with Per-core DVFS

```
Variables and functions:
$Q_x(d, y)$ - estimated delivery time from node $x$ to node $d$ via node $y$
$q_x(dirIn)$ - the queue time of the most recent packet received from port $dirIn$ then sent from node $x$
determinstic $minQ_x$:
$dir\ minQ_x(d, \{directions\})$ - a function to select the routing direction $dir$ with minimum Q-value from $\{directions\}$
non-determinstic $minQ_x$:
$\{dirs\}\ minQ_x(d, \{directions\})$ - a function to select the routing directions $\{dirs\}$ with minimum Q-value from $\{directions\}$
$t_y = (d == y)?q_y : Q_y(d,$ deterministic $minQ_y(d, \{all\ directions\})$

Update the Q-table for power gating and DVFS:
$if$ (node $y$ is power gated) $then$
  node $y$ sends signal to all neighboring nodes
  $\forall x \in \{neighbors\ of\ y\},\ \forall d \in NoC \wedge d \neq x,\ Q_x(d, y) = \infty$
$else\ if$ (node $y$ is power resumed) $then$
  node $y$ sends signal to all neighboring nodes
  $\forall x \in \{neighbors\ of\ y\},\ \forall d \in NoC \wedge d \neq x,\ Q_x(d, y) = initial\ value$
$else\ if$ (frequency of node $y$ is scaled from $f$ to $f'$) $then$
  node $y$ sends signal to all neighboring nodes
  $\forall x \in \{neighbors\ of\ y\},\ \forall d \in NoC \wedge d \neq x,\ Q_x(d, y) = Q_x(d, y) + q_y(f'^{-1} - f^{-1})$
$end\ if$

Routing polocy:
$for$ each node $x$ to route a packet $p$ with destination node $d$
  $if$ $(x == d)$ $then$
    route $p$ to local PE
  $if$ ($x$'s neighbor $y == d$) $then$
    route $p$ to $y$
  $else$
    route $p$ to (either determinstic or non-determinstic)$minQ_x(d, \{\forall dir \in directions \wedge dir \neq input\})$
```

Secondly, this approach requires less area and power overhead for the NoCs. According to the simulation results shown in [5], the TC circuits consume 0.76% area overhead to the total router area and 0.08% power overhead to the total router power, whereas in the time-out implementations, these numbers are 2.9% and 0.11% respectively.

Algorithm 4: Update the Q-table for regular packet delivery

```
when node $x$ sends a packet $p$ to its neighbor $y$
  if p was routed to the $port \neq local \neq neighbor \neq dirIn$ (meaning
routing was "normal") then
      save the queue time of $p$ to $q_x$
      send q-table value query request to all neighbor nodes
$\forall y \in \{neighbors\ of\ x\}$
      wait for q-table value feedback signal from all neighbor nodes
$\forall y \in \{neighbors\ of\ x\}$
    $\forall y \in \{neighbors\ of\ x\}$
      $(\forall d \in NoC) \wedge (d \neq x)$
         update Q-table with $Q'_x(d,y) = Q_x(d,y) + \Delta Q_x(d,y)$
          where $\Delta Q_x(d,y) = \eta(q_x + 1 + t_y - Q_x(d,y))$
  end if
when node $y$ receives q-table value query request from its neighbor $x$
  send feedback signal with entire q-table to node $x$
```

Thirdly, this approach can be realized using a distributed architecture, which speeds up the necessary computation and avoids introducing traffic overhead in the communication network. Moreover, as shown in Figure 6.7, the TC-unit and TC-interconnect can be easily integrated into our proposed agent based NoC architecture by utilizing a part of the agent resources.

After a deadlock is detected, the system will try to recover either regressively or progressively. A regressive recovery is based on packet dropping and retransmitting. While a progressive recovery utilizes additional hardware to bypass the packet to its destination [7]. Both approaches have their pros and cons depending on the application specification, which need to be evaluated according to the application specifications.

6.5.3 Case Studies

To demonstrate how the proposed Q-routing algorithm works under different network scenarios, in this section, we will show 4 different test cases simulated by our modified version of Noxim simulation environment. In each case, only one pair of source and destination nodes are used for the clearer demonstration purpose.

For all these cases, the simulations last for 10000 clock cycles with the initial 1000 cycles as system warming up period. Clock dividers are added to the routers for DVFS purpose and power gating feature is also enabled. The packet injection rate is set at 0.01 by default, meaning that the source node sends packets to the destination at the probability of 1%. Due to the random

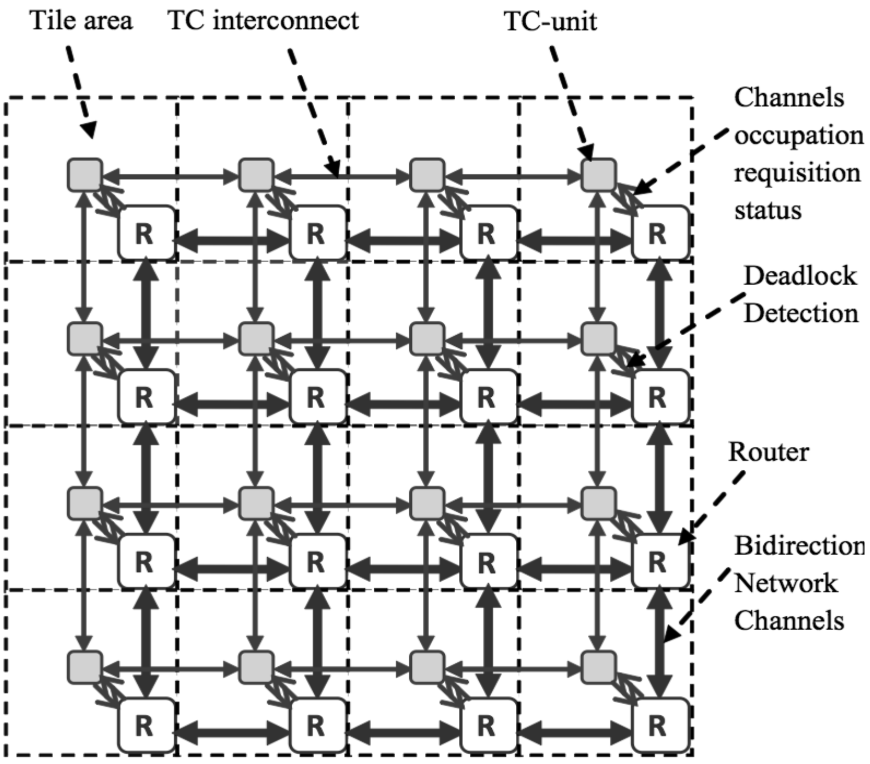

Tile area
TC interconnect
TC-unit
Channels occupation requisition status
Deadlock Detection
Router
Bidirection Network Channels
R

Figure 6.7: A TC-network coupled to a mesh network [5]

nature, the simulation results varies slightly for each set of simulation. Thus, the average numbers of 10 sets of simulations are used in these case studies.

Case 1

In this case study, node 4 constantly sends packets to node 7 after the system is warmed up. The operating frequency of router 5 and 6 is scaled down to 1/10 of its origin from clock cycle 2500. In situations where several routing paths between the current router and the destination node share the same Hamming distance, the packet will be routed in a randomly selected direction with the shortest distance.

As shown in Figure 6.8, before cycle #2500, 81 packets [1] have been routed along the shortest path "4 $\rightarrow$ 5 $\rightarrow$ 6 $\rightarrow$ 7". However, after the frequencies of router 5 and 6 are scaled down, this path becomes no longer the fastest routing alternative. Therefore, packets are delivered along either "4 $\rightarrow$ 0 $\rightarrow$ 1$\rightarrow$ 2 $\rightarrow$ 3 $\rightarrow$ 7" or "4 $\rightarrow$ 8 $\rightarrow$ 9 $\rightarrow$ 10 $\rightarrow$ 11 $\rightarrow$ 7".

[1]The number on the top-left corner of each router represents the number of packets delivered via this router.

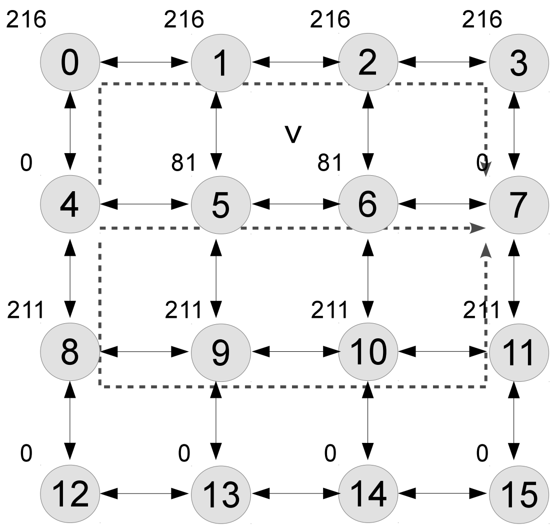


Figure 6.8: Q-routing: case study 1

Case 2

The traffic and DVFS configurations in this case study remain the same as they were in Case 1. However, this time we use deterministic routing selection when there are more than one paths with the same Hamming distance to the destination node.

Figure 6.9 shows the packet delivery scenario for this case. Instead of choosing two alternatives when the frequencies of router 5 and 6 are scaled down, the deterministic algorithm uses only "$4 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 7$" since the north direction has higher priority than south in the routing list.

Case 3

In this case study, the traffic configuration remains the same as in the previous cases. However, only the operating frequency of Router 5 is scaled down since cycle #2500, while for Router 6 since cycle #5000. The choice of routing paths with the same distance to the destination node is based on random selection.

Therefore, as can be seen in Figure 6.10, within the time period between the power scaling of Router 5 and 6, paths "$4 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 6 \rightarrow 7$" or "$4 \rightarrow 8 \rightarrow 9 \rightarrow 10 \rightarrow 6 \rightarrow 7$" are also utilized since they have the same Hamming distance for paths "$2 \leftrightarrow 7$" and "$10 \leftrightarrow 7$".

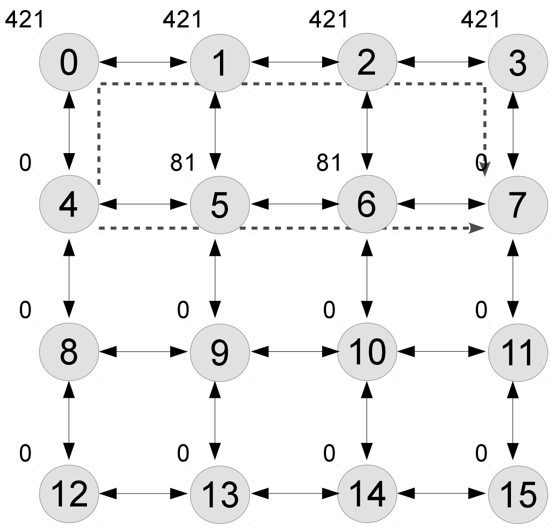


Figure 6.9: Q-routing: case study 2

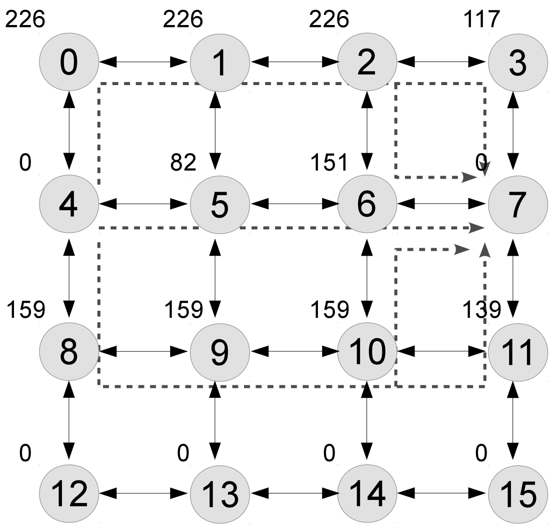


Figure 6.10: Q-routing: case study 3

Case 4

In this case study, we demonstrate how the proposed Q-routing algorithm can handle both power gating and power resuming. The data traffic is still from node 4 to node 7. Router 5 and 6 are power gated from clock cycle #5000 and #2500 respectively, while they are both power resumed since cycle #7500.

The traffic flows are shown in Figure 6.11. Between the warming up of system and cycle #2500, path "4 → 5 → 6 → 7" is used since it suffers from the shortest communication delay. However, when Router 6 is power gated at cycle #2500, the routing paths are changed to ”4 → 5 → 1 → 2 → 3 → 7” and ”4 → 5 → 9 → 10 → 11 → 7”, and later on ”4 → 0 → 1 → 2 → 3 → 7” and ”4 → 8 → 9 → 10 → 11 → 7” are also used after the power gating status of Router 6 is propagated to Router 5. Since cycle #5000 when Router 5 is power gated, the paths via Router 5 are disabled immediately and all packets are then de-routed. Finally when Router 5 and 6 are power consumed, path ”4 → 5 → 6 → 7” again becomes the communication route due to its shortest distance.

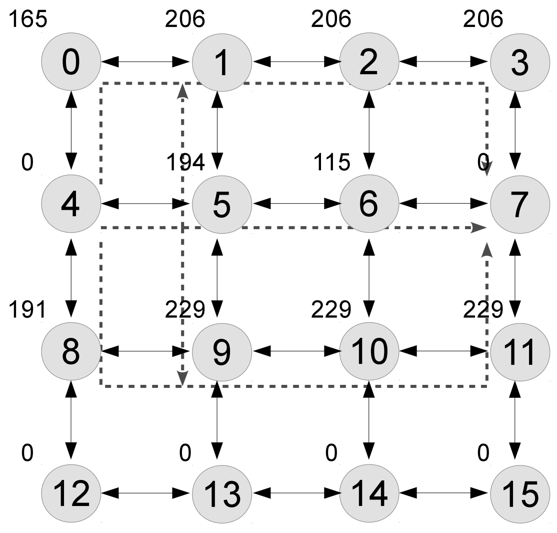


Figure 6.11: Q-routing: case study 4

6.6 Chapter Summary

Run-time energy optimization takes into account the dynamic nature of NoC based systems and thus can effectively improve the energy efficiency. In this chapter, Dynamic Voltage and Frequency Scaling (DVFS) and power gating techniques are studied together with their implementation approaches on agent based NoCs. By the joint efforts of agents at all hierarchies, network operations can be autonomously adjusted to reduce the total energy consumption. With different monitoring and controlling granularity, island based and per-core based DVFS architectures are studied and compared. According to our simulation, by using the proposed method, the total energy consumption for on-chip communication is significantly reduced while having little circuit area and communication delay overhead.

Due to the ever-changing network condition, deterministic routing algorithms such as XY routing can lead to severe performance degradation or even system failures. Therefore, we have introduced a fully adaptive reinforcement routing algorithm based on Q-learning technique. Instead of ensuring deadlock freedom, in our research, deadlock recovery strategy is chosen for the proposed architecture. A run-time Transitive Closure (TC) approach is employed due to its high accuracy, low overhead and easiness to be integrated in the agent based network architecture. With four different case studies, we have shown its capability of handling both DVFS and power gating incidents.

Chapter 7

Honeycomb NoC

Topology is one of the most important characteristics of NoC architectures. For the same application, even with same configurations, different topologies will result in significantly different system performances as well as energy consumption. During the last decade, 2D Mesh and Torus are the dominating topologies for NoC based systems due to their simplicities and symmetries. In this chapter, we propose a Honeycomb based NoC architecture which provides more efficient energy consumption thanks to better network performances in terms of network cost, power and area consumption as well as communication delay.

7.1 Honeycomb Topology Overview

To evaluate the implementation feasibilities of different NoC topologies, network cost, as shown in Equation 7.1, is introduced in [30] and [23]. In this equation, degree refers to the number of channels entering and leaving each node in the network, while diameter is the largest, minimal hop count over all pairs of terminal nodes. Thus, in designing a topology, there is always a trade-off between degree, which relates to the hardware cost, and diameter, which relates to the message transmission time. The elaboration of these parameters is presented in Section 7.2.

$$NetworkCost = Degree * Diameter \tag{7.1}$$

In this chapter, we examine Honeycomb topology and its implementation feasibility in NoC based systems. By adopting Honeycomb topology, a significant proportion of network cost in NoC systems can be reduced, while maintaining the regularity, symmetry and scalability. Furthermore, energy consumption is more efficient in Honeycomb networks thanks to the lower router power consumption, simpler router architecture and higher communication efficiency. We also propose a turn-model based deadlock free routing algorithm for Honeycomb topologies.

As shown in Chapter 2, Mesh (Figure 7.1(a)) and Torus (Figure 7.1(b)) are the dominating topologies for NoC based systems. Similarly, for Honeycomb topologies, we can also define the Mesh (Figure 7.1(c)) and Torus (Figure 7.1(d)) alternatives. A Honeycomb topology is composed of a number of hexagons. The size of a Honeycomb is defined as the number of rings of hexagons. For example, the size of a single hexagon is 1 for a Honeycomb Mesh topology, which is denoted as HM_1. The HM_2 of size 2 is obtained by adding 6 hexagons on the boundary of HM_1. Inductively, HM_t is obtained by adding a ring of hexagons on the boundary of HM_{t-1}. Like in rectangular topologies, by adding a number of wrap-around links on the boundary, a Honeycomb Mesh can be turned into a Honeycomb Torus topology.

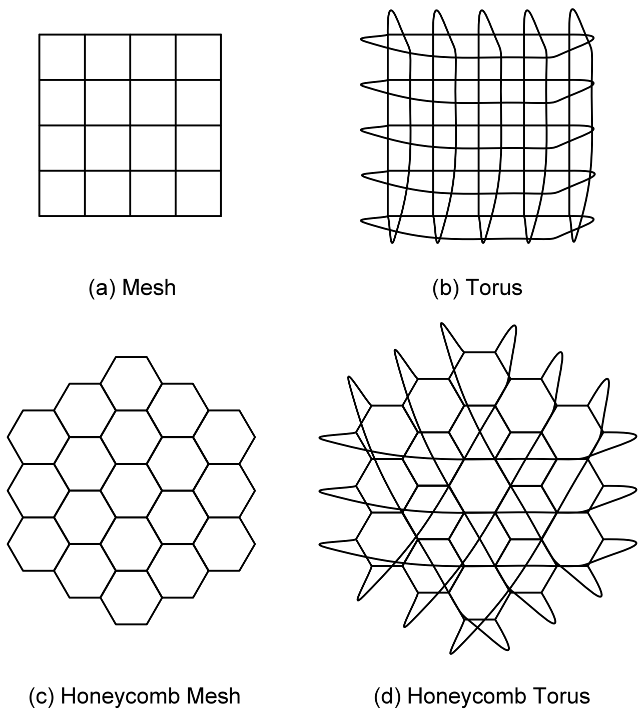


Figure 7.1: Rectangular and Honeycomb topologies

7.2 Topology Properties

The coordinate system used in this chapter was proposed in [92]. Figure 7.2 shows the coordination axes on Honeycomb Mesh topology. The X, Y and Z axes start from the center of the Honeycomb and evenly divide the topology into three regions. The nodes with the same coordinate number in X, Y or Z direction form a zigzag chain which is vertical to the axis. Examples are shown in Figure 7.2 where all nodes on each of the bold zigzag chains have the same Z coordinate, e.g., z=3. Let the two central zigzag chains have the Z coordinate of 0 and 1 respectively, the six chains with respect to Z axis are z = {-2, -1, 0, 1, 2, 3}. The Z coordinate can be changed only by moving along the links which are parallel to the Z axis.

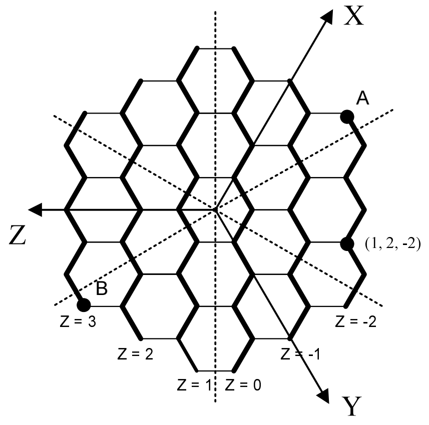


Figure 7.2: Coordinate and coding of Honeycomb networks

To quantify the benefits of Honeycomb topologies over regular 2D Mesh and Torus, we explore several key topological factors such as hop count distance, network diameter, number of terminal nodes and eventually network cost. The exploration is illustrated by the following lemmas and theorems.

LEMMA 1. Nodes of HM_t can be coded by integer triples (x, y, z) such that $-t+1 \leq x, y, z \leq t$, and $1 \leq x+y+z \leq 2$ [92].

Proof for this lemma can be found in [92].

LEMMA 2. The distance between a pair of nodes (x, y, z) and (x', y', z') is $|x-x'|+|y-y'|+|z-z'|$.

According to Lemma 1, the Honeycomb network is coded in three axes, X, Y and Z. Hops on each axis can affect the coordinate only on this axis. Therefore, the Hamming distance is suitable to be used to define distance between two nodes in the Honeycomb Mesh topology.

LEMMA 3. The diameter of HM_t is 4t-1.

According to the definition, diameter is the largest, minimal hop count over all pairs of terminal nodes in the network. In a synchronous on-chip interconnect, the distance between two nodes determines the minimal hop count. Thus, it can be observed from Figure 7.2 that the pairs of nodes which are on the opposite outermost border of the Honeycomb, e.g. A and B, have the largest minimal hop count. The generalized coordinates of nodes A and B are $(t, 0, -t+1)$ *and* $(-t+1, 1, t)$ *respectively. Therefore, based on lemma 2, the diameter* H_{max} *is* $|t-(-t+1)|+|0-1|+|(-t+1)-t| = 4t-1$.

LEMMA 4. The number of terminal nodes in HM_t $N = 6t^2$. [92]

As is shown in Figure 7.2, a Honeycomb Mesh topology HM_t can be partitioned into six regions by the dotted lines based on the edges and vertexes of HM_1. It can be observed that in each region, the outermost border of HM_t has 2t-1 vertexes. Therefore, the total number of terminal nodes in HM_t is $\sum_{n=1}^{t} 6 \times (2t-1) = 6t^2$.

THEOREM 1. The network cost of Honeycomb Mesh topology which has N terminal nodes is $12\sqrt{\frac{N}{6}} - 3$.

As is mentioned previously, the network cost is calculated as the product of network degree and diameter. In the Honeycomb Mesh topology, the network degree is 3. From lemma 3 and 4, the diameter is $4\sqrt{\frac{N}{6}} - 1$.

Similarly, the topology properties of Honeycomb Torus network are studied and analysed.

THEOREM 2. The diameter of Honeycomb Torus topology HT_t which has N terminal nodes is $2t = 2\sqrt{\frac{N}{6}}$ and the network cost is $6\sqrt{\frac{N}{6}}$.

THEOREM 3. The diameters of of rectangular 2D Mesh and Torus topologies are $2\sqrt{N}-2$ and $\sqrt{N}$, and thus network costs $8\sqrt{N}-8$ and $4\sqrt{N}$ respectively [23].

The diameter of a network is the maximum inter-node distance i.e. the maximum number of hop counts that must be traversed when sending a packet to any node along a shortest path. The lower the diameter of a network the shorter the time to send a packet from one node to the node furthest away from it. On the contrary, a network with a long diameter faces many problems, such as long communication delay, coupling problems and the existence of uncontrolled noise (distorted part of a signal). Therefore, having a large number of nodes and a small diameter is one of the most desirable features in network design.

The degree of a node is the number of connections attached to the it, and the maximum degree over all the nodes in the network is the maximum degree of the network. The degree of a node is restricted by hardware constraints. In large interconnection networks, higher amount of wiring is required, and thus greater part of the system performance is spent on operating the wiring. Furthermore, the more wiring is implemented, the more costly the system will become. Therefore, it is desirable to keep a small maximum degree of the network while its size increases.

As shown in Equation 7.1, network cost is the product of diameter and degree. Therefore, it refers to the balance between cost and performance. From the graphical impression shown in Figure 7.3, it is obvious that by using the Honeycomb topologies, the network costs can be reduced by approximately 40%.

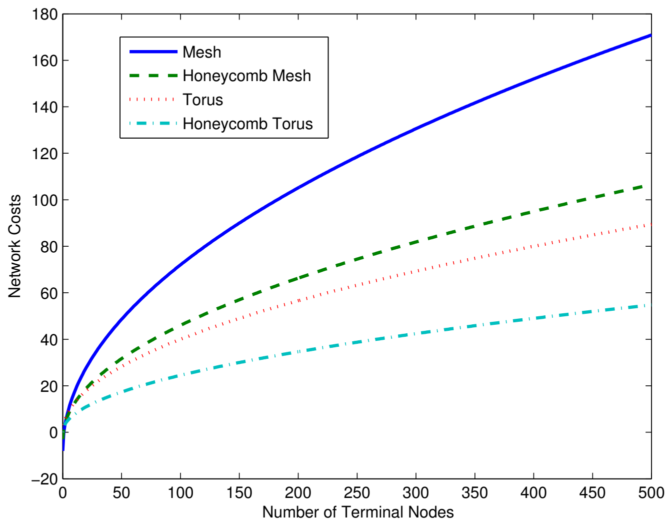


Figure 7.3: Comparison of the network costs between rectangular and Honeycomb topologies

7.3 Implementation Feasibility

In the modern IC technology, devices of rectangular shapes are dominating the fabrication industry. Therefore, NoCs which are of Honeycomb shapes may not be the most efficient interconnect implementation in terms of placement, wire routing, layout and fabrication.

To solve this challenge, we exploit the transformation of both Honeycomb Mesh and Torus topologies into rectangular shapes. By horizontally squeezing Honeycomb topologies towards the center and vertically drawing the zigzag chains into straight lines, Honeycomb Mesh HM_3 (Figure 7.1 (c)) and Honeycomb Torus HT_3 (Figure 7.1 (d)) are transformed into the rectangular brick shapes which are shown in Figure 7.4 (a) and (b) respectively.

Comparing with the hexagon Honeycomb Mesh topology, the "brick" implementation keeps the same coordinate system, link lengths and routing algorithms. Moreover, due to its rectangular shape, routers and PEs such as processors, memory banks and RF transmitters enjoy the higher feasibilities for placement, layout and fabrication. Similarly, the "brick" implementation of Honeycomb Torus is also more advantageous than the hexagon alternative.

To summarize, these brick shapes not only have the implementation feasibility, but also keep all the topological properties and advantages of the Honeycomb architectures. Therefore, this study provides a topological innovation and serves as a general guideline for Honeycomb NoC architectures.

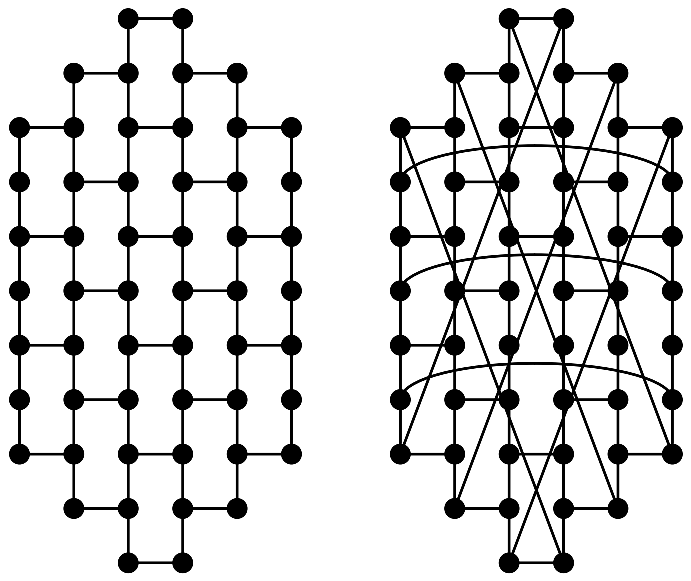


Figure 7.4: Brick transformations of Honeycomb Mesh and Torus

7.4 Router Designs

As shown in the previous sections, from the architectural point of view, the main difference between rectangular Mesh (hereinafter "Mesh") and Honeycomb Mesh (hereinafter "Honeycomb") routers is the number of income and outgoing ports. In a Mesh network, besides the port connection to the local PE, there are four other directions to be connected, namely, North, East, South and West. While in a Honeycomb network, the number of non-local directions is reduce to three, namely, X, Y and Z, either in positive or negative direction.

Figure 7.5 and 7.6 illustrate the router architectures of Mesh and Honeycomb networks. In a Mesh router, the local port is connected to the Network Interface (NI) attached to the local PE while the other four ports are connected to the neighbouring routers via inter-module links. At the incoming side of each port, there are input buffers of predefined sizes where incoming packets are stored before being sent forward. In NoCs where Virtual Channels (VCs) are needed, the incoming buffer should then be split to several logical buffers according to the number and specifications of VCs. The VC allocator decides from which VC buffer should a packet be taken and sent forward, according to the VC allocation algorithm. Similarly, based on the routing algorithm, a switch allocator decides from which input buffer a packet should be taken and to which direction it shall be sent forward via the crossbar inside the router.

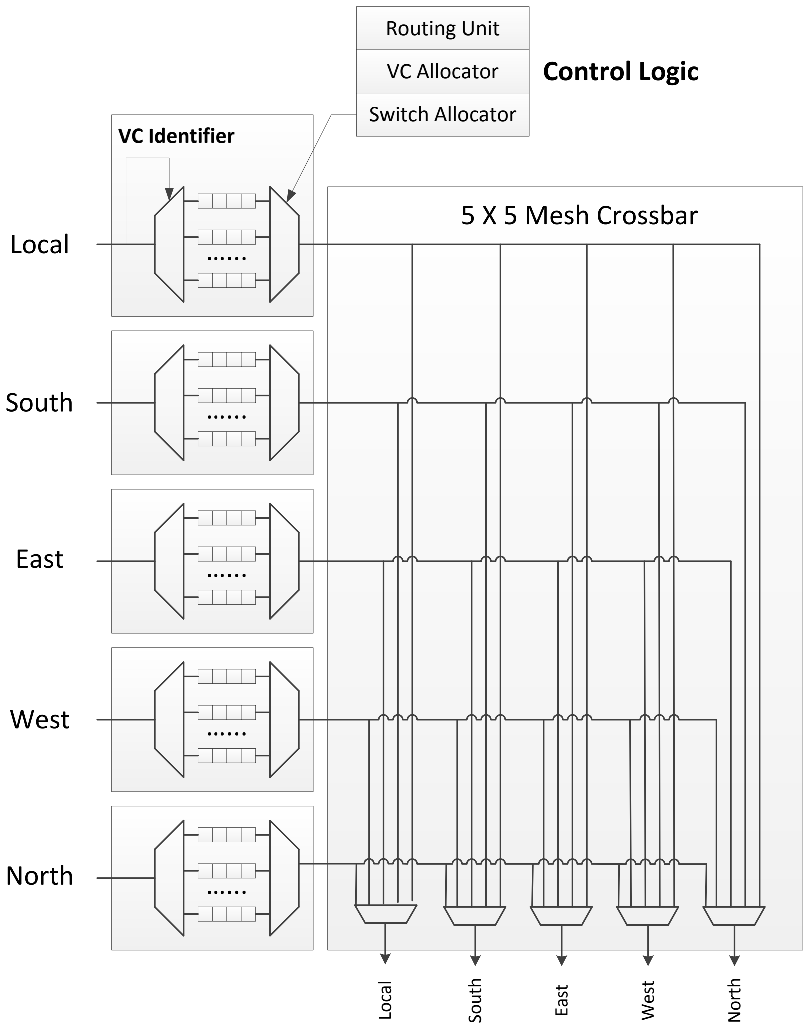


Figure 7.5: Architecture of a Mesh router

The architecture of Honeycomb routers is similar to the Mesh counterparts. However, since there are only three output directions besides the local PE, namely, X, Y and Z, either in the positive or negative side, a Honeycomb router has simpler architecture and thus higher reliability and lower costs.

We compare the Mesh and Honeycomb routers in terms of area cost and power consumption. The simulation is performed using a widely known and accepted NoC simulator called Orion 2.0 [102].

In our simulation, the supply voltage of each router is set at 1.0 V and operating frequency at 800 MHz. The process technologies used to compare Mesh and Honeycomb routers are 90 nm, 65 nm, 45 nm and 32 nm. We

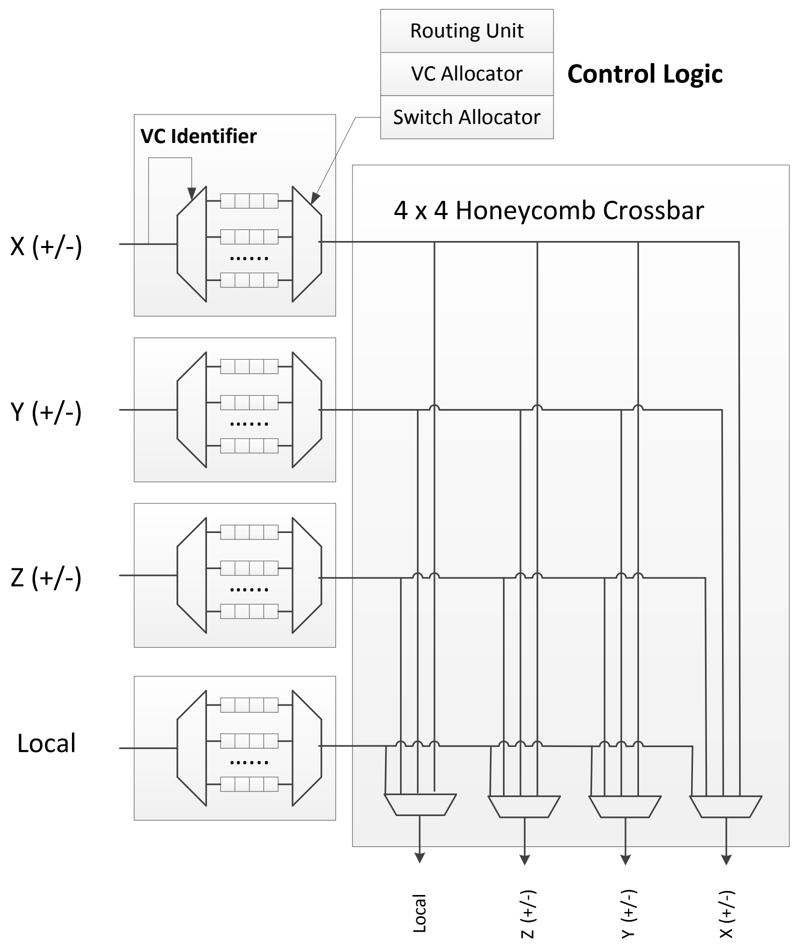


Figure 7.6: Architecture of a Honeycomb router

assume that each flit is 32 bits long and 4-stage pipeline is used. For each incoming port, 4 VCs are applied which share 8 flit input buffer. No buffer is allocated to the output ports. The simulation results are shown in Figure 7.8 and 7.7. It is obvious that both area and power consumptions are significantly reduced in Honeycomb routers.

7.5 Routing Algorithms

The turn model is one of the most common bases for deadlock free routing algorithms in rectangular network topologies. During our research, we have found that the turn model can be extended also to Honeycomb topologies by eliminating one of the six turns in each cyclical dependency circle.

As is illustrated in Figure 7.9 (a), there are two cyclical dependency

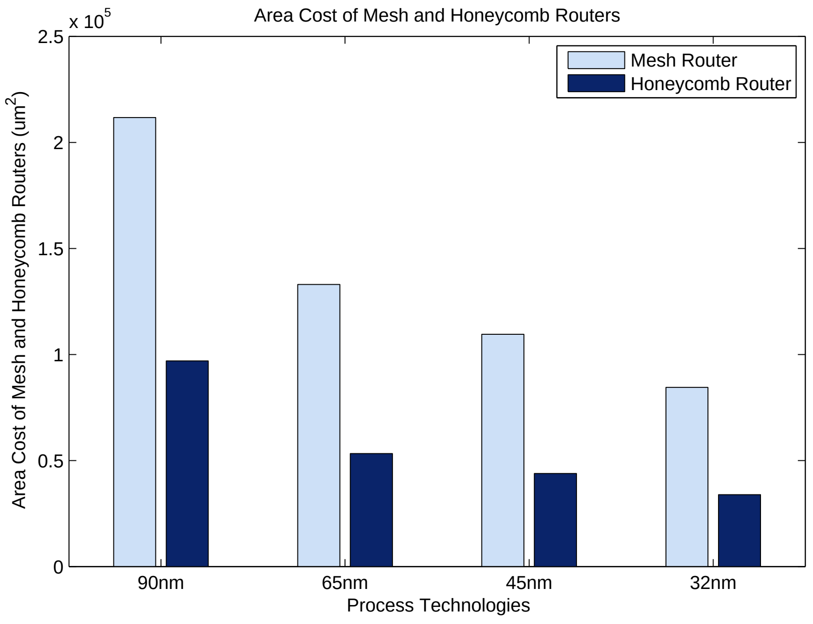


Figure 7.7: Area cost of Mesh and Honeycomb routers under different technologies

circles in a Honeycomb Mesh network; one is counter-clockwise and the other is clockwise. Each circle is formed by six turns. The two sides of each turn are in the directions of X+, X-, Y+, Y-, Z+, Z- and the measure of the included angle is 120°. According to the turn model theory, deadlock free routing can be achieved if there is no cyclical dependency circle in the routing graph. Thus, one of the six turns in each circle need to be prohibited to break the cyclical dependency. Figure 7.9 (b) shows one of the prohibition possibilities in the counter-clockwise circle. The "ring" formed by the other five turns is named as a "+Z -Y" turn since it is equivalent to a turn which starts from the +Z direction and ends in the -Y direction. Figure 7.9 (c) presents the six possible turn prohibitions in the clockwise circle. However, if we take the "+Z -Y" turn in Figure 7.9 (b) and "-Y +Z" turn in Figure 7.9 (c), as is shown in Figure 7.10, a new cyclical dependent circle which is of "∞" shape will be formed. But all other five possibilities in Figure 7.9 (c) can be combined with the one in Figure 7.9 (b) to avoid the deadlock.

To illustrate the routing algorithm, we take the "+Z -Y" and "-X +Y" turns which are shown in Figure 7.11. This routing algorithm is named as -X +Z first routing. In a wormhole routing scheme, a flit firstly checks if the destination is on the -X or +Z side of its current position. If so, it is routed in the -X +Z zigzag chain until there is no more -X and +Z. Then it

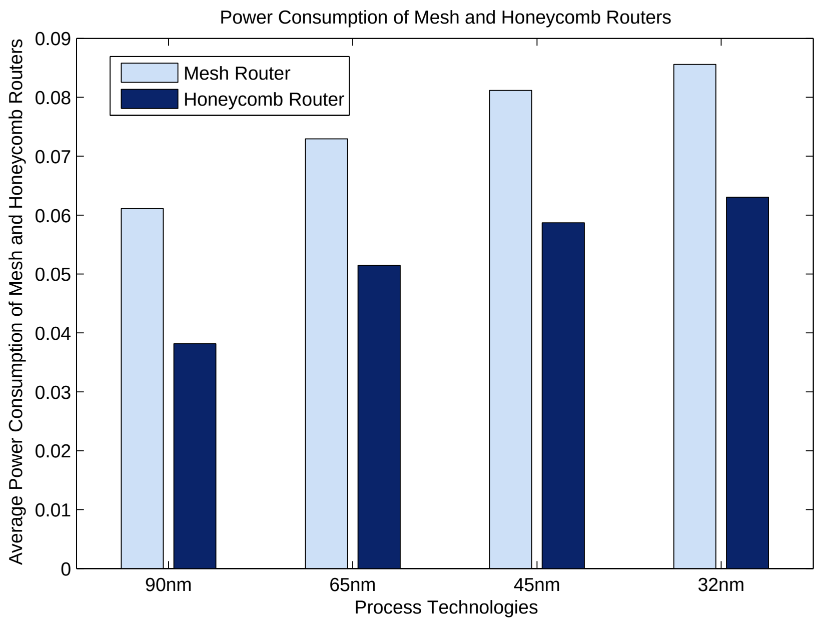


Figure 7.8: Power consumption of Mesh and Honeycomb routers under different technologies

is delivered according to the minimal routing algorithm until it reaches the destination. In case a flit reaches the border before finishing all the -X and +Z, it will be routed in the Y dimension for 1 hop so that the routing in -X and +Z can be continued. If the destination is neither on the -X nor +Z side of its source position, the flit is routed by minimal routing algorithm from the beginning. When the flit reaches the destination, it is routed to the network interface (NI) and then forwarded to the processing element. Three example paths for the -X +Z first routing algorithm are shown in Figure 7.12 where the white squares represent source nodes, black squares represent destination nodes and the arrowed bold lines represent the routing paths.

Algorithm 5 shows the proposed deadlock free "-X +Z" first routing in Honeycomb Mesh topology. The variables and functions used in the algorithm are listed in Table 7.1.

7.6 Simulation Results

In this subsection, we simulate the proposed -X+Z routing algorithm in Honeycomb NoCs using a modified Noxim simulation environment. The network architecture and coordination system are implemented as previ-

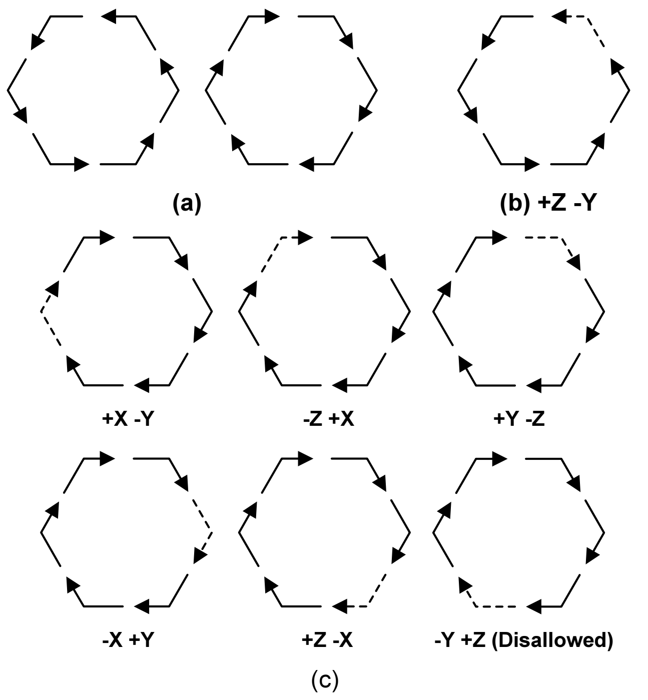


Figure 7.9: Turn model based routing algorithm on Honeycomb Mesh topology. (a) two cyclical dependency circles. (b) eliminate one turn in the counter-clockwise cycle. (c) possible turn eliminations in the clockwise cycle.

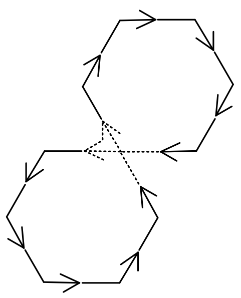


Figure 7.10: The combinatorial cyclical dependent circle formed by two non-cyclical dependent rings

ously described in Figure 7.2. Unlike in Mesh topology where the network formation starts from one of the four corners, in Honeycomb NoC, the formation starts from the inner most ring of nodes. Thereafter, accordingly to the size of Honeycomb, the entire architecture is built ring by ring from the network center.

To illustrate the advantages of Honeycomb topology, Mesh topologies

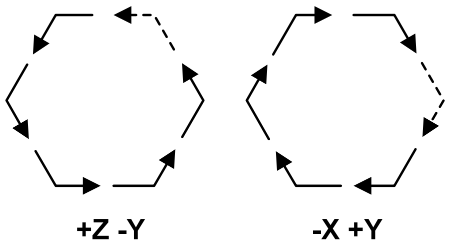


Figure 7.11: The +Z -Y and -X +Y turns

Table 7.1: List of variables and function

Variable/Function	Description
f.pos	current position of a flit
f.dst	destination address
f.dir	next routing direction
minRoute(f)	send f to the next node according to minimal routing algorithm

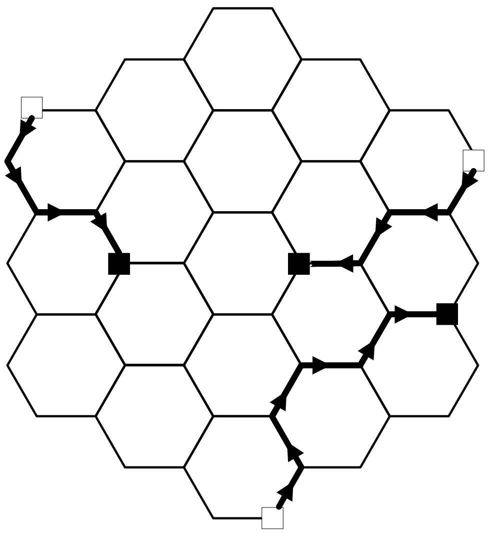


Figure 7.12: Examples of the -X +Z first routing in a Honeycomb Mesh network

with XY routing algorithm are used as comparisons. As proved in Lemma 4, the number of nodes in a Honeycomb network $N = 6t^2$ where t is the size of Honeycomb. In reality, the number of PEs can be different from $6t^2$ with slight adjustments in routing algorithms. However in this simulation, for

Algorithm 5: -X +Z first routing in Honeycomb Mesh

```
procedure mXpZ(Mesh, flit):
for each flit f in Honeycomb Mesh:
    for(f.pos.X>f.dst.X or f.pos.Z<f.dst.Z)
        f.dir := (-X,+Z)
        do send f to next node
            if f reaches Honeycomb border
                f.dir := Y
        until no (-X,+Z)
    minRoute(f)
    if f.pos == f.dst
        f.dir := NI
```

the sake of simplicity, complete Honeycomb networks with size 1 and 2 are utilized where there are 6 and 24 PEs in each network respectively. To have the consistent number of PEs, the Mesh architectures in our simulation are 2×3 and 4×6 networks.

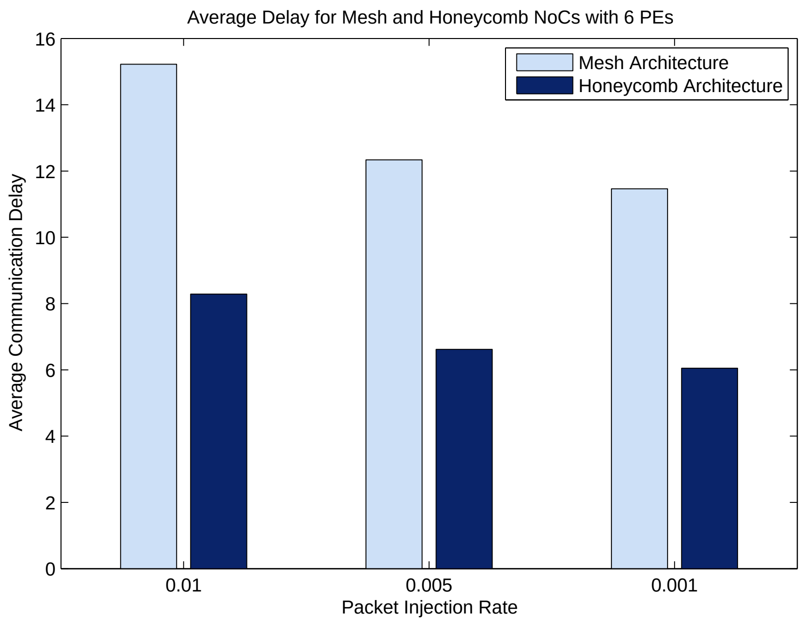


Figure 7.13: Average delay for Mesh and Honeycomb architectures with 6 PEs

Moreover, to test the network performance under different loads, we use three different packet injection rates (0.001, 0.005 and 0.01) for each topology and the time distribution of traffic is based on memory-less Poisson

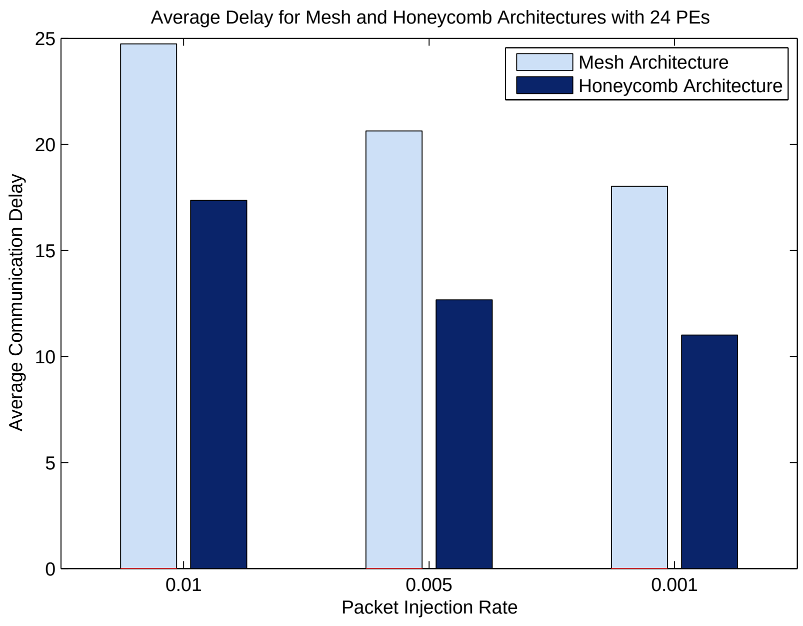


Figure 7.14: Average delay for Mesh and Honeycomb architectures with 24 PEs

distribution. The simulations of Mesh topology are performed using Noxim simulator.

Figure 7.13 shows the comparisons of average packet delivery delays between Honeycomb and Mesh networks with 6 PEs. Comparing with Mesh topology, the delays in Honeycomb are 45.5%, 46.4% and 47.2% shorter respectively. The comparisons between networks with 24 PEs are shown in Figure 7.14 where the average network delays are reduced by 29.8%, 38.6% and 39% respectively.

As explained in Chapter 4, communication has become the bottleneck for both system and power performances and consumes an increasingly large proportion of energy. Therefore, the proposed Honeycomb topology with significantly reduced network delay enables higher energy efficiency for NoC based systems thanks to the alleviation of communication bottleneck.

From the simulation, we also notice that with the increase of network size and the number of PEs, the improvement of network performance becomes less significant. This is due to the fact that comparing with Mesh topology, Honeycomb network often have more traffic among the central rings of routers. Thus, packets will be stored for a longer time in the relevant FIFOs of the routers.

7.7 Chapter Summary

In this chapter, we present a Honeycomb architecture for NoC based systems. The chapter starts with the illustration of network cost which is the product of degree and diameter of a topology. It is often used to evaluate the implementation feasibility of different NoC topologies and represents the trade-off between hardware cost and system performance. In our research, Honeycomb architecture is introduced as an alternative NoC topology with better energy performance, comparing with the most commonly used 2D Mesh and Torus topologies. From the architectural analysis shown in Section 7.2, Honeycomb NoC has the benefits of lower degree and diameter, and thus approximately 40% lower network cost.

As one of the key properties of a NoC topology, deadlock-free routing algorithms are presented in this chapter. By using Orion, Mesh and Honeycomb Noxim simulation environments, we have shown that Honeycomb topology can significantly alleviate the communication bottleneck for NoC based systems, and therefore provides more optimized energy efficiency.

Chapter 8

Conclusion

Energy efficiency is one of the most critical challenges for modern computing platforms such as portable devices, work stations, data centres, etc. The main concerns include thermal effect, heat dissipation, rising cost of power and environmental considerations. During the past decades, the number of integrated on-chip transistors has doubled approximately every 18 months which leads to significantly improved system performance. However, power consumption of microprocessors, at the same time, has gone up from under 1 Watt to over 130 Watts.

With the increasing number of integrated on-chip components, communication has become a major bottleneck for large scale Integrated Circuits (ICs). In order to solve this issue, at the beginning of this century, researchers have proposed a Network-on-Chip (NoC) architecture based on which our research was conducted.

8.1 Summary of Contributions

NoC based systems are more energy efficient than bus-based SoCs since heterogeneous and well optimized Intellectual Property (IP) cores can be easily integrated and with larger amount. In this thesis, we focused on the architectural level design in order to further optimize the energy efficiency for NoC based systems. Our research has been conducted mainly in the following aspects.

Three Dimensional Architecture - Three dimensional architecture has been proposed already since 1980s in the form of System-in-Package (SiP) where chips are stacked vertically. Nevertheless, communication among layers is achieved still via off-chip links in SiPs. During the recent years, with more advanced technologies such as TSVs, the real 3D NoCs have become feasible where all the layers are within the same chip and thus off-chip communication and global wirings are reduced significantly. As shown in

Chapter 4, 3D NoCs provide higher energy efficiency than traditional 2D systems.

Nevertheless, the benefits of 3D NoCs do not come for free. In this thesis, we proposed a method of change function which can quantitatively measure the pros and cons of using 3D architecture for NoC based systems. A number of key benefits and limitations of 3D NoCs have been studied and compared with the 2D network. We claimed that 3D NoCs should only be adopted when the change function value is larger than the expected threshold.

Considering the thermal and heat dissipation challenges with the current technology, 3D NoCs with processing units over all layers would be neither economically feasible nor operationally reliable. However, 3D integration with on-chip DRAM is a promising solution by having all processors to be placed on the top layer which is close to the heat sink. A case study showed the performance and energy benefits alongside an optimized TSV insertion architecture.

Agent-based Monitoring - In large scale NoCs with dynamic nature of network traffic, in order to achieve high energy efficiency while having reliable system performance, a hierarchical agent based monitoring design approach was proposed in this thesis. The agents collect the real-time information and adjust system operations at different hierarchies, i.e., cell level, cluster level, platform level and eventually application level. The joint effort of all the agents enables NoC based systems with run-time self-optimization capability for higher energy efficiency as well as fault and variation tolerance.

The implementation of the proposed hierarchical agent based approach has also been discussed in Chapter 5. Different alternatives were studied and compared in terms of power consumption and communication latency. A case study of 64-bit FFT/IFFT implementation showed the validity of the proposed method.

DVFS and Power Gating - Dynamic Voltage and Frequency Scaling (DVFS) and power gating are two effective techniques to reduce system power consumption. The former reduces dynamic power consumption by providing "just enough" voltage supplies and operating frequencies, while the latter focuses on leakage power consumption which becomes more significant with feature sizes shrink. We have studied the implementations of both techniques for NoC based systems by using the hierarchical agent based design approach. Simulations in different scenarios showed that the choice between island based and per-core based implementation should depend on the target traffic patterns.

Due to the changing network condition and configuration as the result of DVFS and power gating support, deterministic routing algorithms such as XY routing may lead to severe performance degradation or even system failures. Therefore, we have proposed a reinforcement routing algorithm based on Q-learning technique. With different test cases, we have shown how the DVFS and power gating incidents are handled by the proposed algorithm.

Topological Exploration - Another energy optimization approach studied in this thesis was to exploit different network topologies. Honeycomb NoC topology was proposed in Chapter 7 with its design and implementation feasibility studies. By surveying and comparing with different topologies, especially Mesh networks, we have proved that Honeycomb NoC has the benefits of not only lower implementation cost but also lower area and routing power consumption. Moreover, with the turn model based deadlock free routing algorithm, simulations showed that Honeycomb NoC can reduce the communication delay by a large proportion, which further optimizes the total energy consumption at the application level.

8.2 Future Direction

The improvement of energy efficiency will be a continuous research topic for modern computing platforms. From the architecture design perspective, the following two issues will be of great interests in the near future:

- The joining forces of 3D design and topological exploration will further improve the energy efficiency. Currently, the 3D designs are mainly based on Mesh topology while topological exploration on 2D systems. The future development of integrated circuit technology will certainly remove the barriers and allow more flexible topologies to be implemented in a multi dimensional manner. In this case, network communication will no longer be an obstacle and thus consume less power as well as time.

- Autonomous system optimization will be achieved with techniques including but not limited to DVFS and power gating. With the development of machine learning methodology, computing platforms will become more autonomously intelligent. Future on-chip communication techniques will enable the system with more accurate prediction as well as faster and more effective reaction. Energy efficiency will then be taken to another level, if this is the case.

Bibliography

[1] S. Abedinpour, Semicond Freescale, and AZ Tempe. A multi-stage interleaved synchronous buck converter with integrated output filter in a 0.18/spl mu/ sige process. In *IEEE International Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers*, pages 1398 – 1407, 2006.

[2] Adrijean Adriahantenaina, Herve Charlery, Alain Greiner, Laurent Mortiez, and Cesar Albenes Zeferino. Spin: A scalable, packet switched, on-chip micro-network. In *Proceedings of the conference on Design, Automation and Test in Europe*, pages 70–73, 2003.

[3] U.S. Environmental Protection Agency. Epa report on server and data center energy efficiency, 2007.

[4] Y. Akasaka. Three-dimensional ic trends. 74(12):1703–1714, 1986.

[5] R. Al-Dujaily, T. Mak, Fei Xia, A. Yakovlev, and M. Palesi. Run-time deadlock detection in networks-on-chip using coupled transitive closure networks. In *Proc. Design, Automation & Test in Europe Conf. & Exhibition (DATE)*, pages 1–6, 2011.

[6] AMD. The amd opteron 6000 series platform. http://www.amd.com/us/products/server/processors/6000-series-platform/pages/6000-series-platform.aspx, May 2010.

[7] K. V. Anjan and T. M. Pinkston. Disha: a deadlock recovery scheme for fully adaptive routing. In *Proc. th Int Parallel Processing Symp.*, pages 537–543, 1995.

[8] S. I. Association. The international technology roadmap for semiconductors (itrs). http://www.itrs.net/Links/2007ITRS/Home2007.htm, 2007.

[9] Semiconductor Industry Association. Interbational technology roadmap for semiconductors. Technical report, World Semiconductor Council, 1999.

[10] Bevan M. Baas. *An Approach to Low-Power, High-Performance, Fast Fourier Transform Processor Design.* PhD thesis, Stanford University, 1999.

[11] C. Bienia, S. Kumar, and Kai Li. Parsec vs. splash-2: A quantitative comparison of two multithreaded benchmark suites on chip-multiprocessors. In *Proc. IEEE Int. Symp. Workload Characterization IISWC 2008*, pages 47–56, 2008.

[12] M. Bohr. The new era of scaling in an soc world. In *Proc. IEEE Int. Solid-State Circuits Conf. - Digest of Technical Papers ISSCC 2009*, pages 23–28, 2009.

[13] R. Casas and O. Casas. Battery sensing for energy-aware system design. *Computer*, 38(11):48–54, 2005.

[14] Chuan-Hua Chang. *Performance Optimization of Pipeline Circuits With Latches and Wave Pipelining.* PhD thesis, University of Michigan, 1996.

[15] Xuning Chen and Li-Shiuan Peh. Leakage power modeling and optimization in interconnection networks. In *Proc. Int. Symp. Low Power Electronics and Design ISLPED '03*, pages 90–95, 2003.

[16] Yibo Chen, Dimin Niu, Yuan Xie, and K. Chakrabarty. Cost-effective integration of three-dimensional (3d) ics emphasizing testing cost analysis. In *Proc. IEEE/ACM Int Computer-Aided Design (ICCAD) Conf*, pages 471–476, 2010.

[17] Kihwan Choi, R. Soma, and M. Pedram. Fine-grained dynamic voltage and frequency scaling for precise energy and performance tradeoff based on the ratio of off-chip access to on-chip computation times. *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems*, 24(1):18–28, January 2005.

[18] P. Christie and D. Stroobandt. The interpretation and application of rent's rule. 8(6):639–648, 2000.

[19] C. Ciordas, K. Goossens, T. Basten, A. Radulescu, and A. Boon. Transaction monitoring in networks on chip: The on-chip run-time perspective. In *Proc. International Symposium on Industrial Embedded Systems IES 06*, pages 1–10, 2006.

[20] Calin Ciordas. *Monitoring-Aware Network-on-Chip Design.* PhD thesis, Eindhoven University of Technology, 2008.

[21] Pip Coburn. *The Change Function: Why Some Technologies Take Off and Others Crash and Burn.* Portfolio, 2006.

[22] William James Dally. Future directions for on-chip interconnection networks. OCIN Workshop, December 2006.

[23] William James Dally and Brian Patrick Towles. *Chapter 1, Principles and Practices of Interconnection Networks.* Morgan Kaufmann, 2004.

[24] William James Dally and Brian Patrick Towles. *Chapter 12, Principles and Practices of Interconnection Networks.* Morgan Kaufmann, 2004.

[25] William James Dally and Brian Patrick Towles. *Chapter 14, Principles and Practices of Interconnection Networks.* Morgan Kaufmann, 2004.

[26] William James Dally and Brian Patrick Towles. *Chapter 15, Principles and Practices of Interconnection Networks.* Morgan Kaufmann, 2004.

[27] V. De and S. Borkar. Technology and design challenges for low power and high performance [microprocessors]. In *Proc. Int Low Power Electronics and Design Symp*, pages 163–168, 1999.

[28] W. Donath. Placement and average interconnection lengths of computer logic. 26(4):272–277, 1979.

[29] Xiangyu Dong and Yuan Xie. System-level cost analysis and design exploration for three-dimensional integrated circuits (3d ics). In *Proc. Asia and South Pacific Design Automation Conf. ASP-DAC 2009*, pages 234–241, 2009.

[30] Kemal Efe. A variation on the hypercube with lower diameter. *IEEE Transactions on Computers*, 40 (11):1312–1316, 1991.

[31] C. Grecu, L. Anghel, P.P. Pande, A. Ivanov, and R. Saleh. Essential fault-tolerance metrics for noc infrastructures. In *Proc. 13th IEEE International On-Line Testing Sym- posium IOLTS 07*, pages 37–42, 2007.

[32] L. Guang, E. Nigussie, L. Koskinen, and H. Tenhunen. Autonomous dvfs on supply islands for energy-constrained noc communication. In *Proc. 22nd International Conference on Architecture of Computing Systems ARCS '09*, pages 183–194, 2009.

[33] W. Haensch, E. J. Nowak, R. H. Dennard, P. M. Solomon, A. Bryant, O. H. Dokumaci, A. Kumar, X. Wang, J. B. Johnson, and M. V. Fischetti. Silicon cmos devices beyond scaling. *IBM Journal of Research and Devel- opment*, 50:339–361, 2006.

[34] Paul J.M. Havinga. *Mobile Multimedia Systems.* PhD thesis, University of Twente, 2000.

[35] Shousheng He and M. Torkelson. A new approach to pipeline fft processor. In *Proc. 10th International Parallel Processing Symposium, IPPS 96*, pages 766–770, 1996.

[36] Ahmed Hemani, Axel Jantsch, Shashi Kumar, Adam Postula, Johnny berg, Mikael Millberg, and Dan Lindqvist. Network on chip: An architecture for billion transistor era. In *Proc. IEEE NorChip Conference*, 2000.

[37] R. Ho, K. W. Mai, and M. A. Horowitz. The future of wires. *Proceedings of the IEEE*, 89(4):490–504, 2001.

[38] Ang-Chih Hsieh, TingTing Hwang, Ming-Tung Chang, Min-Hsiu Tsai, Chih-Mou Tseng, and H.-C. Li. Tsv redundancy: Architecture and design issues in 3d ic. In *Proc. Design, Automation & Test in Europe Conf. & Exhibition (DATE)*, pages 166–171, 2010.

[39] Jingcao Hu and R. Marculescu. Energy and performance- aware mapping for regular noc architectures. *IEEE Transactions on COMPUTER-AIDED DESIGN of Inte- grated Circuits and Systems*, 24(4):551–562, 2005.

[40] Miro Panades. I. and A. Greiner. Bi-synchronous fifo for synchronous circuit communication well suited for network-on-chip in gals architectures. In *Proc. First International Symposium on Networks-on-Chip NOCS 2007*, pages 83–94, 2007.

[41] IBM. Ibm power 7 processor. In *Hot Chips 2009*, 2009.

[42] Sungjun Im and K. Banerjee. Full chip thermal analysis of planar (2-d) and vertically integrated (3-d) high performance ics. In *Proc. IEDM Technical Digest Electron Devices Meeting Int*, pages 727–730, 2000.

[43] Intel. Intel core i7-980x processor extreme edition. http://ark.intel.com/Product.aspx?id=47932, May 2010.

[44] Axel Jantsch and Hannu Tenhunen, editors. *Chapter 1, Networks on Chip.* Springer, 2003.

[45] Axel Jantsch and Hannu Tenhunen, editors. *Chapter 5, Networks on Chip.* Springer, 2003.

[46] James T. Kao, Masayuki Miyazaki, and Anantha P. Chandrakasan. A 175-mv multiply-accumulate unit using an adaptive supply voltage and body bias architecture. *IEEE Journal of Solid-State Circuits*, 37:1545–1554, 2002.

[47] Faraydon Karim, Anh Nguyen, and Sujit Dey. An interconnect architecture for networking systems on chips. *IEEE Micro*, 22 (5):36–45, 2002.

[48] C. Kim, D. Burger, and S. W. Keckler. An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches. *ACM SIGPLAN*, 36, 2002.

[49] Hyungjun Kim, P. Ghoshal, B. Grot, P. V. Gratz, and D. A. Jimenez. Reducing network-on-chip energy consumption through spatial locality speculation. In *Proc. Fifth IEEE/ACM Int Networks Chip (NoCS) Symp*, pages 233–240, 2011.

[50] Hyunjin Kim, Hyejeong Hong, Hong-Sik Kim, Jin-Ho Ahn, and Sungho Kang. Total energy minimization of real-time tasks in an on-chip multiprocessor using dynamic voltage scaling efficiency metric. *IEEE Trans- actions on CAD*, 27(11):20882092, November 2008.

[51] J. S. Kim, M. B. Taylor, J. Miller, and D. Wentzlaff. Energy characterization of a tiled architecture processor with on-chip networks. In *Proc. Int. Symp. Low Power Electronics and Design ISLPED '03*, pages 424–427, 2003.

[52] Seunghoon Kim, Robert Dick, and Russ Joseph. Power deregulation: Eliminating off-chip voltage regulation cir- cuitry from embedded systems. In *Proc. of International Conference on Hardware/Software Codesign and System Synthesis (CODES-ISSS 2007)*, 2007.

[53] M. B. Kleiner, S. A. Kuhn, P. Ramm, and W. Weber. Thermal analysis of vertically integrated circuits. In *Proc. Int. Electron Devices Meeting*, pages 487–490, 1995.

[54] Teijo Lehtonen, Pasi Liljeberg, and Juha Plosila. Online reconfigurable self-timed links for fault tolerant noc. *VLSI Design*, 13, 2007.

[55] Feihui Li, C. Nicopoulos, T. Richardson, Yuan Xie, V. Narayanan, and M. Kandemir. Design and management of 3d chip multiprocessors using network-in-memory. In *Proc. 33rd Int. Symp. Computer Architecture ISCA '06*, pages 130–141, 2006.

[56] G. Liang and A. Jantsch. Adaptive power management for the on-chip communication network. In *Proc. 9th EUROMICRO DSD Conference*, pages 649–656, 2006.

[57] Hsin-Lei Lin, Hongchin Lin, Yu-Chuan Chen, and R.C. Chang. A novel pipelined fast fourier transform archi- tecture for double rate ofdm systems. In *Proc. IEEE Workshop on Signal Processing Systems SIPS 2004*, pages 7–11, 2004.

[58] Dake Liu and C. Svensson. Power consumption estimation in cmos vlsi chips. 29(6):663–670, 1994.

[59] G. H. Loh. 3d-stacked memory architectures for multi-core processors. In *Proc. 35th Int. Symp. Computer Architecture ISCA '08*, pages 453–464, 2008.

[60] G. L. Loi, B. Agrawal, N. Srivastava, Sheng-Chih Lin, T. Sherwood, and K. Banerjee. A thermally-aware performance analysis of vertically integrated (3-d) processor-memory hierarchy. In *Proc. 43rd ACM/IEEE Design Automation Conf*, pages 991–996, 2006.

[61] J.-Q. Lu, Y. Kwon, R. P. Kraft, R. J. Gutmann, J. F. McDonald, and T. S. Gale. Stacked chip-to-chip interconnections using wafer bonding technology with dielectric bonding glues. In *Proc. IEEE 2001 Int. Interconnect Technology Conf*, pages 219–221, 2001.

[62] Z. Lu, A. Jantsch, E. Salminen, and C. Grecu. Network- on-chip benchmarking specification part 2: Microbench- mark specification version 1.0. Technical report, OCP International Partnership Association, Inc., 2008.

[63] Zhonghai Lu. *Design and Analysis of On-Chip Communication for Network-on-Chip Platforms.* PhD thesis, Royal Institute of Technology, 2007.

[64] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hallberg, J. Hogberg, F. Larsson, A. Moestedt, and B. Werner. Simics: A full system simulation platform. *Computer*, 35(2):50–58, 2002.

[65] E. J. Marinissen and Y. Zorian. Testing 3d chips containing through-silicon vias. In *Proc. Int. Test Conf. ITC 2009*, pages 1–11, 2009.

[66] A. Markopoulou, Y. Li, C. Chan, and N. Bambos. Energy-efficient communication in battery-constrained portable devices. In *Proc. 2nd Int. Conf. Broadband Networks BroadNets 2005*, pages 408–417, 2005.

[67] S.M. Martin, K. Flautner, T. Mudge, and D. Blaauw. Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads. In *Proc. IEEE/ACM International Conference on Computer Aided Design ICCAD 2002*, page 721 725, 2002.

[68] J. D. Meindl. Low power microelectronics: retrospect and prospect. 83(4):619–635, 1995.

[69] P. Mercier, S. R. Singh, K. Iniewski, B. Moore, and P. O'Shea. Yield and cost modeling for 3d chip stack technologies. In *Proc. IEEE Custom Integrated Circuits Conf. CICC '06*, pages 357–360, 2006.

[70] T. L. Michalka, R. C. Varshney, and J. D. Meindl. A discussion of yield modeling with defect clustering, circuit repair, and circuit redundancy. 3(3):116–127, 1990.

[71] B. T. Murphy. Cost-size optima of monolithic integrated circuits. 52(12):1537–1545, 1964.

[72] P. K. Nag, A. Gattiker, Sichao Wei, R. D. Blanton, and W. Maly. Modeling the economics of testing: a dft perspective. *IEEE Design & Test of Computers*, 19(1):29–41, 2002.

[73] Kshirasagar Naik. A survey of software based energy saving methodologies for handheld wireless communication devices. Technical report, University of Waterloo, 2010.

[74] University of Catania. Noxim - the noc simulator. http://noxim.sourceforge.net/, 2010.

[75] A. Papanikolaou, D. Soudris, and R. Radojcic, editors. *Three Dimensional System Integration: IC Stacking Process and Design.* Springer Publishers, 2011.

[76] Dongkook Park, S. Eachempati, R. Das, A. K. Mishra, Yuan Xie, N. Vijaykrishnan, and C. R. Das. Mira: A multi-layered on-chip interconnect router architecture. In *Proc. 35th Int. Symp. Computer Architecture ISCA '08*, pages 251–261, 2008.

[77] A. Patel and K. Ghose. Energy-efficient mesi cache coherence with pro-active snoop filtering for multicore microprocessors. In *Proc. ACM/IEEE Int Low Power Electronics and Design (ISLPED) Symp*, pages 247–252, 2008.

[78] Michael K. Patterson and Dave Fenwick. White paper: The state of data center cooling. Technical report, Intel Corporation, Digital Enterprise Group, 2008.

[79] V. F. Pavlidis and E. G. Friedman. 3-d topologies for networks-on-chip. In *Proc. IEEE Int. SOC Conf*, pages 285–288, 2006.

[80] L.-S. Peh and W.J. Dally. A delay model and speculative architecture for pipelined routers. In *Proc. of The Seventh International Symposium on High-Performance Computer Architecture*, pages 255–266, 2001.

[81] V. Raghunathan, M. B. Srivastava, and R. K. Gupta. A survey of techniques for energy efficient on-chip communication. In *Proc. Design Automation Conf*, pages 900–905, 2003.

[82] George A. Riley. Nailing ics together. Tutorial, flipchips.com, February 2007.

[83] Brian M. Rogers, Anil Krishna, Gordon B. Bell, Ken Vu, Xiaowei Jiang, and Yan Solihin. Scaling the bandwidth wall: challenges in and avenues for cmp scaling. In *Proceedings of the 36th annual international symposium on Computer architecture*, 2009.

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

[85] T. Sakurai and A.R. Newton. Alpha-power law mosfet model and its applications to cmos inverter delay and other formulas. *IEEE Journal of solid-state circuits*, 25(2):584594, 1990.

[86] R. Saleh, S. Wilton, S. Mirabbasi, A. Hu, M. Greenstreet, G. Lemieux, P. P. Pande, C. Grecu, and A. Ivanov. System-on-chip: Reuse and integration. 94(6):1050–1069, 2006.

[87] Li Shang, Li-Shiuan Peh, and N. K. Jha. Dynamic voltage scaling with links for power optimization of interconnection networks. In *Proc. Ninth Int. Symp. High-Performance Computer Architecture HPCA-9 2003*, pages 91–102, 2003.

[88] Abbas Sheibanyrad, Frdric Ptrot, and Axel Jantsch, editors. *3D Integration for NoC-based SoC Architectures (Integrated Circuits and Systems)*. Springer, 2010.

[89] T. Shyamkumar, M. Naveen, A. J. Ho, and J. N. P. Cacti 5.1. Technical report, HP Labs, 2008.

[90] Amit Sinha. *Energy Efficient Software and Operating Systems*. PhD thesis, Massachusetts Institute of Technology, 2001.

[91] SPEC. Specjbb 2000. http://www.spec.org/jbb2000/.

[92] Ivan Stojmenovic. Honeycomb networks: Topological properties and communication algorithms. *IEEE Transactions on Parallel and Distributed Systems*, 8 (10):1036–1042, 1997.

[93] C. Svensson and A. Alvandpour. Low power and low voltage cmos digital circuit techniques. In *Proc. Int Low Power Electronics and Design Symp*, pages 7–10, 1998.

[94] D. Sylvester, D. Blaauw, and E. Karl. Elastic: An adap- tive self-healing architecture for unpredictable silicon. *IEEE Design & Test of Computers*, 23(6):484–490, 2006.

[95] Craig Michael Teuscher. *Low Power Receiver Design for Portable RF Applications: Design and Implementation of an Adaptive Multiuser Detector for an Indoor, Wideband CDMA Application.* PhD thesis, UNIVERSITY OF CALIFORNIA, BERKELEY, 1998.

[96] TPC. Tpc-h decision support benchmark. http://www.tpc.org/tpch/.

[97] M. Tremblay and S. Chaudhry. A third-generation 65nm 16-core 32-thread plus 32-scout-thread cmt sparc processor. In *Proc. Digest of Technical Papers. IEEE Int. Solid-State Circuits Conf. ISSCC 2008*, pages 82–83, 2008.

[98] D. Truong, W. Cheng, T. Mohsenin, Zhiyi Yu, T. Jacobson, G. Landge, M. Meeuwsen, C. Watnik, P. Mejia, Anh Tran, J. Webb, E. Work, Zhibin Xiao, and B. Baas. A 167-processor 65 nm computational platform with per- processor dynamic supply voltage and dynamic clock frequency scaling. In *Proc. IEEE Symposium on VLSI Circuits*, pages 22–23, 2008.

[99] S.R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erra-guntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar. An 80-tile sub-100-w teraflops processor in 65-nm cmos. *IEEE Journal of Solid-State Circuits*, 43(1):29–41, 2008.

[100] H. J. M. Veendrick. Short-circuit dissipation of static cmos circuitry and its impact on the design of buffer circuits. 19(4):468–473, 1984.

[101] D. Velenis, M. Stucchi, E. J. Marinissen, B. Swinnen, and E. Beyne. Impact of 3d design choices on manufacturing cost. In *Proc. IEEE Int. Conf. 3D System Integration 3DIC 2009*, pages 1–5, 2009.

[102] Hang-Sheng Wang, Xinping Zhu, Li-Shiuan Peh, and S. Malik. Orion: a power-performance simulator for interconnection networks. In *Proc. 35th Annual IEEE/ACM Int. Symp. (MICRO-35) Microarchitecture*, pages 294–305, 2002.

[103] Hangsheng Wang. A detailed architectural-level power model for router buffers, crossbars and arbiters. Technical report, Department of Electrical Engineering, Princeton University, 2004.

[104] R. Weerasekera, Li-Rong Zheng, D. Pamunuwa, and H. Tenhunen. Extending systems-on-chip to the third dimension: performance, cost and technological tradeoffs. In *Proc. IEEE/ACM Int. Conf. Computer-Aided Design ICCAD 2007*, pages 212–219, 2007.

[105] A. Y. Weldezion, Zhonghai Lu, R. Weerasekera, and H. Tenhunen. 3-d memory organization and performance analysis for multi-processor network-on-chip architecture. In *Proc. IEEE Int. Conf. 3D System Integration 3DIC 2009*, pages 1–7, 2009.

[106] D. Wentzlaff, P. Griffin, H. Hoffmann, Liewei Bao, B. Edwards, C. Ramey, M. Mattina, Chyi-Chang Miao, J.F. Brown, and A. Agarwal. On-chip interconnection ar- chitecture of the tile processor. *IEEE MICRO*, 27:15–31, 2007.

[107] T. W. Williams and N. C. Brown. Defect level as a function of fault coverage. *IEEE Transactions on Computers*, 12:987–988, 1981.

[108] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta. The splash-2 programs: characterization and methodological considerations. In *Proc. Symp. nd Annual Int Computer Architecture*, pages 24–36, 1995.

[109] T. C. Xu, P. Liljeberg, and H. Tenhunen. A study of through silicon via impact to 3d network-on-chip design. In *Proc. Int Electronics and Information Engineering (ICEIE) Conf. On*, volume 1, 2010.

[110] T. C. Xu, A. W. Yin, P. Liljeberg, and H. Tenhunen. A study of 3d network-on-chip design for data parallel h.264 coding. In *Proc. NORCHIP*, pages 1–6, 2009.

[111] Alexander Yin. Generalization of slot table size for virtual circuits on nostrum networks on chip. Master's thesis, The Royal Institute of Technology (KTH), Sweden, 2008.

[112] Rongtian Zhang, Kaushik Roy, Cheng-Kok Koh, and D. B. Janes. Power trends and performance characterization of 3-dimensional integration for future technology generations. In *Proc. Int Quality Electronic Design Symp*, pages 217–222, 2001.

[113] Jishen Zhao, Xiangyu Dong, and Yuan Xie. Cost-aware three-dimensional (3d) many-core multiprocessor design. In *Proc. 47th ACM/IEEE Design Automation Conf. (DAC)*, pages 126–131, 2010.

Turku Centre for Computer Science
TUCS Dissertations

ISBN 978-952-12-2918-3
ISSN 1239-1883

Alexander Wei Yin

On Energy Efficient Computing Platforms

Alexander Wei Yin

On Energy Efficient Computing Platforms

Alexander Wei Yin

On Energy Efficient Computing Platforms