On the power efficiency, low latency, and quality of service in network-on-chip

Wang, P.

On the Power Efficiency, Low Latency, and Quality of Service in Network-on-Chip

Peng Wang

On the Power Efficiency, Low Latency, and Quality of Service in Network-on-Chip

PROEFSCHRIFT

ter verkrijging van
de graad van Doctor aan de Universiteit Leiden,
op gezag van Rector Magnificus Prof.mr. C.J.J.M. Stolker,
volgens besluit van het College voor Promoties
te verdedigen op woensdag 12 februari 2020
klokke 11:15 uur

door

Peng Wang
geboren te Liaoning, China
in 1990

Promotor:	Dr. Todor P. Stefanov	Universiteit Leiden
Second-Promotor:	Prof. Dr. Joost N.Kok	Universiteit Leiden
Promotion Committee:	Prof. Dr.-ing Diana Göhringer	Technische Universität Dresden
	Prof. Dr. Georgi Gaydadjiev	Rijksuniversiteit Groningen
	Prof. Dr. Kees G.W. Goossens	Technische Universiteit Eindhoven
	Prof. Dr. Aske Plaat	Universiteit Leiden
	Prof. Dr. Harry A.G. Wijshoff	Universiteit Leiden
	Prof. Dr. Fons J. Verbeek	Universiteit Leiden

On the Power Efficiency, Low Latency, and Quality of Service in Network-on-Chip
Peng Wang. -
Dissertation Universiteit Leiden. - With ref. - With summary in Dutch.

This dissertation was typeset using LaTeX in Linux.
Cover designed by Jian Zhang

ISBN: 978-90-9032677-1
Printed in the Netherlands.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

MOORE's Law[1] has a profound impact on the semiconductor industry and the processor development. The downscaling of the manufacturing size of transistors has resulted in faster, smaller and more efficient transistors, which has created continuing increase of transistor budgets for designers to realize higher performance processors. In the past, taking the advantage of the new generation of advanced transistors, the designers had been respecting the Dennard's scaling [DGR+74][2] to improve the performance of uni-processors (single-thread) by increasing the clock frequency to speed up uni-processors' computations and by taking more transistors to realize more efficient micro-architectures, such as superscalar, superpipeline, branch prediction and so on. However, this situation came to an end around 2005-2007 [Boh07] because the further increase of the clock frequency caused exponential power consumption increase, while at the same time, the cooling technology did not scale exponentially to handle the temperature rise on the chip. As a consequence, the further increase of the clock frequency causes severe power dissipation problems and becomes impractical on a chip. Furthermore, the more advanced, but more complex micro-architectures will consume more power, which makes it difficult to further improve the performance of the uni-processors. As a result, system designers transfer their attention to build multi/many-core System-on-Chips (SoCs). Instead of using a huge and complex uni-processor, designers use a multi/many-core processor containing multiple, relatively simple processing elements (processing cores) to increase the performance with reasonable power consumption. In such a system, the performance almost linearly scales with the number of processing elements [Bor07]. Thus, without the need for scaling

[1] Moore's law is the observation that the number of transistors in a dense integrated circuit doubles about every two years (or 18 months).

[2] Dennard Scaling postulated that as transistors get smaller their power density stays constant, so that the power use stays in proportion with area. This allowed CPU manufacturers to raise clock frequencies from one generation to the next without significantly increasing the overall circuit power consumption.

up the frequency, performance improvements in many-core processors can be gained by simply increasing the number of processing elements.

At the beginning of the multi/many-core systems' development, bus-based communication infrastructures [HHS+00] ware used to support the communication between a few processing elements. However, as many-core systems constantly increase the number of processing elements to gain higher performance, the traditional bus-based communication infrastructure cannot meet the communication demands of large many-core systems. First, as a single centralized and shared infrastructure, a bus does not support parallel communication between multiple processing elements. The processing elements have to access the bus one at a time, which results in a very limited communication bandwidth. Furthermore, with the downscaling of the manufacturing size of transistors, the parasitic capacitance and signal propagation delay on the wires become significant, which results in that the wires delay becomes higher than the gate delay. Thus, the global wires on a bus have to operate at a much lower frequency than the processing elements. As a consequence, the communication over a bus has a significant delay. Finally, there is a significant amount of capacitance on the global wires, which causes high power consumption. The aforementioned drawbacks of low bandwidth, high wire delay, and high power consumption result in poor scalability of bus-based communication infrastructures. As a consequence, the bus-based communication infrastructures cannot efficiently support the communication between the processing elements of large-scale many-core systems.

In order to overcome the aforementioned drawbacks, researchers [DT01, HJK+00, BDM02] inspired by the concepts of off-chip networks have proposed the idea of Network-on-Chip (NoCs). In contrast to the conventional bus, NoCs utilize a distributed communication infrastructure. Routers as intermediate nodes are arranged on a particular topology and connected by short wires. The processing elements access the NoC though their own network interfaces without the need of any global controller. Thus, the processing elements can independently access the NoC. In other words, the NoCs support parallel communication, which not only enormously increases the communication bandwidth, but also significantly reduces the communication latency. In addition, without long global wires, the shorter wires in a NoC cause much less power consumption and much lower delay. Thus, it is possible to achieve a high frequency communication operation in a NoC. Moreover, a processing element uses the network interface to access the NoC, which completely separates the communication on the NoC and the computation on the processing elements. Thus, when designers develop a new chip, they could simply replace the processing elements and do not need to redesign the NoC. Such high reusability of NoCs is an attractive property for the SoC industry, which experiences a high Time-to-Market pressure. Based on the aforementioned advantages, NoCs become a promising communication fabric for future

Chip	NoC Topology	Technology	Frequency	VC setting	Power Consumption
Intel 80-tile [HVS+07]	8×10 Mesh	65 nm	$1.7 \sim 5$ GHz	2 16-flit VCs	28% (4 GHz)
SCCC [HDH+10, SJJ+11]	4×6 Mesh	45 nm	2 GHz	8 4-flit VCs	10% (1 GHz), 5% (250MHz)
TRIPS [GKS+07]	4×10, 5×5 Mesh	130nm	336 MHz	4 2-flit VCs, 1 4-flit VC	-
SCORPIO [DCS+14]	6×6 Mesh	45 nm	833 MHz	4 1-flit VCs, 2 3-flit VCs	19%
Title-64 [BEA+08]	8×8	90 nm	750 MHz	-	-
RAW [TKM+02, KTMW03]	4×4 Mesh	0.15um	420 MHz	4 4-flit VCs	36%

Table 1.1: NoCs on real chips.

large-scale many-core systems.

1.1 Design Trends in Network-on-Chip

Even though the idea of NoCs comes from the off-chip networks, the design constraints on the chip are different from the off-chip. For example, NoCs are power and hardware limited while networks off chip are often pin-bandwidth limited. Such different design constraints result in the NoCs forming their own design trends.

1.1.1 Low Power Consumption

The power consumption is a critical design constraint on a chip. Even though NoCs consume relatively lower power than the conventional bus-based communication infrastructures in large-scale many-core systems, the NoCs power consumption is still the major factor limiting the design and utilization of NoCs. Let us consider some NoCs implemented on real chips and shown in Table 1.1. Most of the NoCs consume significant power on the chip. For example, the NoC in the Intel's 80-tile chip [HVS+07] consumes 28% of the total power of the whole chip under 65 nm technology. Even, with the more power-efficient technology of 45 nm, the NoCs in SCCC [HDH+10, SJJ+11] and SCORPIO [DCS+14] consume 10% and 19% of the total power of the chip, respectively. Furthermore, with the NoC size increasing to connect more processing elements, the NoCs will consume much more power. According to the prediction in [Bor07], the power consumption of a NoC will reach 100 watts when the NoC size further increases to connect 1000 cores. Such high power consumption is unacceptable for the future many-core systems and will become the major factor limiting the utilization of a NoC. Therefore, it is critical and necessary to reduce the power consumption of NoCs.

The power/energy consumption of NoCs can be reduced in different ways. A straightforward way is to simplify the NoCs structure. For example, the complex crossbar in a router can be simplified into a few gates [BKA10, ZML+16] or the buffers in routers can be eliminated [MM09, LSMJ16]. By simplifying the NoCs structure, the NoCs need less hardware and consume less power. However, simpli-

fying the NoCs structure may cause other problems, such as only very limited power reduction [BKA10], poor throughput [MM09], losing packets [LSMJ16], etc.

Applying low power technologies on a NoC is another, more effective way to reduce the NoCs' power consumption. In a NoC with dynamic voltage and frequency scaling (DVFS), when the workload is low, the voltage-frequency (V/F) regulators let the routers work at a low voltage to reduce the power consumption. As the V/F regulators consume extra power, it is necessary to trade off between the power reduction and the extra power consumption caused by the V/F regulators. In practice, to use fewer V/F regulators to avoid the extra power consumption of the V/F regulators, a group of routers [YL16, LY18] or all the routers [BJS+14] share one V/F regulators. Compared with the aforementioned DVFS technique, the implementation of power gating on a NoC is much simpler and low cost power reduction technique. Power gating can be more flexibly applied on a NoC at different granularities. For example, in fine-grained power gating approaches, each component of a router [MKI+10] can be separately powered off. In course-grained power gating approaches, an entire router [WDLW17] or a group of routers [DNSD13] can be powered off by the same power gating controller. Furthermore, as NoCs have the characteristics of a distributed structure, a naturally unbalanced traffic workload, and a low average injection traffic rate, the static power consumption takes a high proportion of the total power consumption. Thus, power gating is an applicable and effective way to reduce the static power consumption of a NoC.

1.1.2 Low Latency

The network latency of a NoC has a significant impact on the system performance. For example, NoCs on general purpose processors should have a low network latency to realize the cache coherence protocol, which needs a low latency communication to synchronize the data located at different places in a very short time.

Many approaches are widely used to reduce the network latency. One approach is to reduce the blocking/stall or congestion, by using multiple virtual channels to reduce the Head-of-Line blocking [DT04] or by using efficient flow control to reduce the traffic congestion in a NoC [OM06, vdBCGB07]. As the packets need to go through multiple routers in order to reach their destinations, the pipeline stages of routers have a significant impact on the network latency. Thus, reducing the pipeline stages of routers in a NoC is necessary and is also widely used to reduce the network latency, such as in the look-ahead routing [Gal97] and in the speculative routing [PD01]. In some state-of-the-art router designs [KKS+07, MKAY09], the router pipeline can be aggressively reduced to only one stage.

Bypass-based approaches are also widely used to achieve a low network latency. The bypass paths in these approaches can be physical or virtual. The implementation

of physical bypass paths [MPK07, HY13] can be more efficient to reduce the network latency, but it needs more wires and complex routers, which causes more area and power overhead. In contrast, virtual bypass paths in NoCs, such as the express virtual channel [KPKJ07, KKC+08], do not need extra physical wires, so they do not cause such high area and power overhead, but the efficiency is lower. Recently, some researches show that, by adding asynchronous repeaters to reduce the wire delay, the wires on the chip can transfer packets to go through a long distance in a short time (under 45 nm manufacturing size, the packet can go up to 16mm in 1ns). Taking this advantage of such high speed wires, some bypass paths [CPK+13, KCKP13, CJ16] can be dynamically built, without the need of extra physical or virtual bypass paths, by reserving the existing wires between routers for a short time to dynamically build bypass paths.

1.1.3 Advanced Quality of Service

Quality-of-Service (QoS) is a strategy for allocating resources according to some service policies or purposes. Future NoCs will be expected to provide advanced QoS due to the growing popularity of service consolidation and real-time demands of SoCs.

At the system level, SoCs grow in complexity with an increasing variety of applications integrated on a single chip. One important branch of these applications is the hard real-time applications, in which the tasks must be finished before their deadline. Before executing a hard real-time application in a system, the functional and temporal behavior of such application should be verified. This verification process cannot be done on the application in isolation, because when multiple applications are simultaneously executed on a NoC-based many-core system, the communication interference on the NoC may influence the temporal behavior of the applications. In order to accurately verify the temporal behavior, the communication interference on the NoC must be considered. However, there are many possible combinations of simultaneously running applications and the communication interference is quit complex, which makes the verification process challenging. In order to facilitate this verification process, the NoCs should support composability, in which the communication (packet transmission) of different applications is completely isolated and cannot affect each other. Thus, the temporal behavior of applications will not be influenced by the communication interference on NoCs.

The circuit switching[3] is an effective way to achieve the aforementioned composability in a NoC. By reserving a routing path between the source router and the destination router, the packet transmission is completely isolated and there is no in-

[3]Circuit switching is a connection-oriented network switching technique. Here, a dedicated route is established between the source and the destination and the entire message/packet is transferred through it.

terference or contention between packets. Thus, there is no communication interference between different applications. In NoCs, the circuit switching can be implemented in different ways, such as time-division-multiplexed (TDM) circuit switching [GDR05, SMG14, GH10] or virtual TDM circuit switching [MNTJ04] (there are multiple virtual channels in the routers). The circuit switching avoids the packet interference but results in low bandwidth utilization and poor scalability. Compared with the circuit switching, a confined-interference communication [WGO+13, P+16] can more efficiently utilize the bandwidth. In the confined-interference communication, the packets of different applications are grouped into different domains and interference can occur only in the same domain, while there is no interference between domains. Thus, the packets in one domain have no influence on the transmission time of the packets in other domains. By supporting confined-interference communication, some composability support in a NoC can be realized in relatively simpler and easier way.

1.2 Contradictions between Design Trends

We have briefly introduced some NoC design trends in Section 1.1. However, there are a few notable contradictions between the design trend for low power consumption and the design trends for the low network latency and advanced QoS.

- **Low power consumption VS low network latency**

  To achieve low communication latency, NoCs need a certain amount of hardware to implement highly efficient communication infrastructure, such as multiple virtual channels to reduce the Head-of-Line blocking, a deep virtual channel to hide the credit-round trip delay, or the physical/virtual bypass paths to reduce the network latency. However, this hardware consumes significant power. If aggressively using a complex hardware structure to reduce the network latency, the power consumption of NoCs may sharply increase. On the other hand, if aggressively simplifying the NoC hardware structure to reduce the power consumption, the network latency may significantly increase. Moreover, in order to reduce the power consumption, low power techniques are applied on NoCs, such as DVFS and power gating. These low power techniques are effective in reducing the power consumption of NoCs, but this reduction is at the cost of increasing the network latency. Thus, to achieve low power consumption, the efficiency of the NoC may be undermined, which results in the network latency increase.

- **Low power consumption VS advanced QoS**

Typically, in order to achieve a certain QoS on NoCs, more hardware is required, which further increases the power consumption of a NoC. For example, in order to achieve a confined-interference communication [WGO+13], a NoC needs a large number of virtual channles in routers. This high hardware overhead in a NoC further increases the power consumption of the NoC. Furthermore, recent research [ZCPP15] shows that the routers in a NoC are potential hotspots on the chip. Thus, it becomes impractical to further increase the hardware overhead of routers to support QoS.

The aforementioned contradictions severely prevent the utilization of NoCs in future large-scale many-core systems. Thus, it becomes crucial to resolve these contradictions in a way which enables efficient untilization of NoCs in such future many-core systems.

1.3 Problem Statement

As discussed in Section 1.2, the high power consumption of a NoC constraints the utilization of NoCs in future large-scale many-core systems. Meanwhile, with more advanced semiconductor technologies, chips can work at lower voltages, which is helpful to achieve more energy-efficient many-core systems, but this also results in the static power consumption taking larger proportion of the total power consumption [BS00]. Thus, in this thesis, we focus our attention on reducing the static power consumption of NoCs in two directions: applying efficient power gating on NoCs to reduce the static power consumption and realizing a confined-interference communication on a simplified NoC infrastructure to achieve energy-efficient packet transmission.

1.3.1 Problem 1

Power gating is a promising low power technique to reduce the high static power consumption of the NoCs by powering off idle components that are not used. In a NoC, power gating can be flexibly applied at different granularities. In fine-grained power gating schemes, power gating can be separately applied on the components of a router, such as input ports, the crossbar, and the allocation unit. In course-grained power gating schemes, power gating can be applied on the whole router or a group of routers. When the components/routers are not used, the power of these components/routers is switched off to reduce the static power consumption. When the powered-off components/routers will be used to transfer packets, the power of these powered-off components/routers is switched on, but these components/routers cannot be immediately used because they have to be fully charged. This charging process is called the wakeup process. The extra clock cycles needed to charge the powered-off components is called

wakeup delay, which is about 6-12 clock cycles [CZPP15] in practice. The wakeup process blocks a routing path for a while. The packets have to be blocked until the powered-off components/routers are fully charged, which causes some packet latency increase. In addition, under a low traffic workload, a packet has high probability to experience multiple wakeup processes and accumulate large wakeup delay. As a consequence, the power gating causes significant packet latency increase. Furthermore, the power gating process itself consumes extra power. As a consequence, frequent power gating or power gating in a short time may cause more power consumption or inefficient power consumption reduction. Therefore, the challenge for applying power gating on NoCs is:

Problem 1: How to reduce the packet latency increase caused by power gating and achieve significant reduction of the power consumption?

1.3.2 Problem 2

As discussed in Section 1.1.3, a confined-interference communication is a promising QoS to meet real-time demands of SoCs. In a confined-interference communication, the communication interference is limited to the same domain, and there is no communication interference between different domains. By supporting a confined-interference communication, the NoCs support composability to facilitate the temporal verification of hard real-time applications. Furthermore, compared with the circuit switching, the confined-interference communication can better utilize the bandwidth of a NoC. However, realizing a confined-interference communication on a conventional (virtual channel/buffer based) NoC [WGO+13, P+16] requires a large number of virtual channels, which causes sharp power consumption increase. Therefore, there is an urgent need for realizing the confined-interference communication on a more power-efficient NoC architecture.

Simplified NoC architectures can be effective in reducing the power consumption. Bufferless NoCs [MM09, FCM11, LSMJ16] are one of the most power-efficient NoCs. By eliminating virtual channels/buffers in routers, the bufferless NoCs consume much less power than the conventional NoCs. However, as there are no buffers in bufferless NoCs to temporarily hold packets, packets have to keep moving, which makes it more difficult to control the interference between packets. As a consequence, current bufferless NoCs do not support a confined-interference communication. Therefore, it is attractive and promising, to exploit the advantage of the low power consumption of bufferless NoCs and to try to realize a confined-interference communication. So, the challenge is:

Problem 2: How to realize a confined-interference communication on a bufferless NoC?

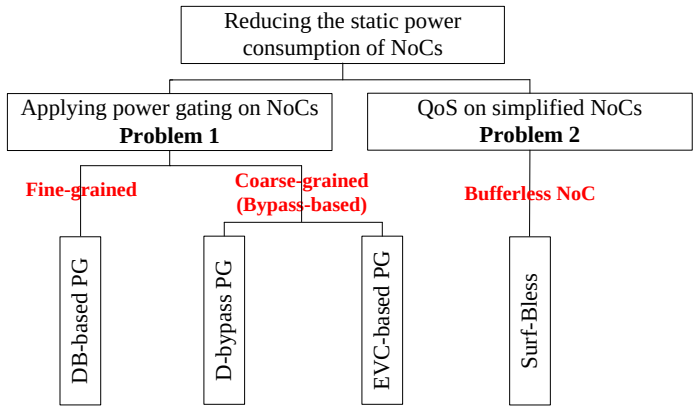


Figure 1.1: Contributions outline.

1.4 Contributions of The Thesis

By addressing the research problems formulated in Section 1.3, in this section, we summarize the research contributions as shown in Figure 1.1. To address Problem 1, described in Section 1.3.1, we propose three novel power gating approaches: duty buffer based (DB-based) power gating, dynamic bypass (D-bypass) power gating, and express virtual channel based (EVC-based) power gating. These three power gating approaches can significantly reduce the power consumption of NoCs and are effective in reducing the packet latency increase caused by power gating. In addition, with different properties, they have their own advantages. As a fine-grained power gating approach, our DB-based power gating approach can fully utilize the idle time of each input port in a router to reduce the static power consumption. Thus, our DB-based power gating approach is effective in reducing the power consumption of a NoC in a wider range of traffic workloads. The D-bypass power gating approach and the EVC-based power gating approach allow packets to bypass the powered-off routers without the need of experiencing the wakeup processes introduced in Section 1.3.1. Thus, they are more effective in reducing the packet latency increase caused by power gating and have less performance penalty. In addition, the EVC-based power gating approach allows packets to bypass powered-on routers as well to reduce the dynamic power consumption. Thus, at high workloads, the EVC-based power gating approach consumes less power than the D-bypass power gating approach. However, the EVC-based power gating approach may cause unfair allocation of the network resources, which may result in more performance penalty.

To address Problem 2, described in Section 1.3.2, we propose a novel routing

approach called Surfing on a Bufferless NoC (Surf-Bless), which achieves an energy-efficient confined-interference communication by taking advantage of the low power consumption of a bufferless NoC.

The specific novel contributions of the thesis are the following:

Contribution 1: Duty buffer based (DB-based) power gating approach

In this contribution, presented in Chapter 3, first, we propose a novel hardware structure, called duty buffer (DB), which can be used to replace any virtual channel in an input port. Then, taking advantage of our novel duty buffer, we propose a novel fine-grained power gating approach, called DB-based power gating approach. In our DB-based power gating approach, each input port of a router can be independently powered off to reduce the static power consumption. As the virtual channels in input ports are the main consumer of the static power in a NoC, our DB-based power gating approach is very effective in reducing the power consumption. Moreover, by using the duty buffer to replace the powered-off virtual channels, even when the input ports are powered-off, a router still keeps certain transmission ability to transfer packets. Thus, the packet can avoid as much as possible to be blocked by the powered-off input ports, and the packet latency increase caused by the power gating is reduced. Finally, by experiments in comparison with the related works [MKWA08, ZOG+15], our DB-based power gating approach achieves comparable power reduction. Moreover, our approach is more effective in reducing the packet latency increase caused by the power gating.

However, being a fine-grained power gating approach, our DB-based power gating needs to separately switch the power of each input port in a router. Thus, some times packets may experience more power gating processes, which may result in a lot extra packet latency increase.

Contribution 2: Dynamic bypass (D-bypass) power gating approach

In order to overcome the aforementioned drawback of the DB-based power gating, we propose the Dynamic bypass (D-bypass) power gating approach, presented in Chapter 4, which is a course-grained power gating approach. First, we apply power gating on each router and add one special hardware bypass structure in each router, which allows a packet to bypass a powered-off router from any input port to any output port at a time. Then, we propose a reservation mechanism to dynamically allocate the use of the special hardware bypass structure in a router. Thus, by dynamically reserving this hardware bypass structure in the powered-off router, packets can bypass the powered-off router more flexibly than in the related approaches [CP12, FTKH16, BHW+17, ZL18]. In our D-bypass power gating approach, as the packet can go through the powered-off routers, some packets do not need to experience the wakeup process and wait for the powered-off routers to be fully charged. Thus, the packet latency increase caused by the power gating is reduced. Furthermore, without

frequent interruption of powering on the powered-off routers, routers have more idle time to stay powered-off to reduce the static power consumption and have less power consumption overhead caused by the power gating. Finally, by experiments, we show that our D-bypass approach can efficiently reduce the power consumption. Moreover, compared with other related approaches [MKWA08, CP12, WNWS17, ZL18], our D-bypass approach has less performance penalty than the related approaches and the DB-based power gating approach.

However, just like most of the course-grained power gating approaches, our D-bypass power gating approach cannot fully utilize the idle time of each component in a router. When the traffic workload is high, most of the routers in a NoC become busy and cannot be powered off to reduce the static power consumption. As a consequence, our D-bypass power gating approach is effective in reducing the power consumption only at low traffic workloads.

Contribution 3: Express virtual channel based (EVC-based) power gating approach

To overcome the aforementioned drawback of the D-bypass power gating approach, we propose the express virtual channel based (EVC-based) power gating approach, presented in Chapter 5. First, we extend the router micro-architecture and apply power gating on each router. Then, we pre-define multiple virtual bypass paths between different routers. Packets can take these virtual bypass paths to bypass intermediate routers that can be powered-on or powered-off. Thus, our EVC-based power gating approach not only reduces the packet latency increase and extra power consumption caused by power gating, but also reduces the dynamic power consumption by transferring packets to bypass the powered-on routers. Thus, even at high traffic workloads when all of the routers are powered-on and power gating is ineffective in reducing the power consumption, our EVC-based power gating approach consumes less power than other related approaches [MKWA08, ZL18]. Moreover, the pre-defined virtual bypass paths are more efficient in transferring packets. Thus, our EVC-based power gating approach can achieve lower packet latency than the related approaches [MKWA08, WNWS17, ZL18]. However, compared with the D-bypass power gating approach, the EVC-based power gating approach may cause unfair allocation of the NoC resources thereby causing, in some cases, higher performance penalty in spite of the fact that the EVC-based power gating approach has lower packet latency.

Contribution 4: Surfing on a Bufferless (Surf-Bless) NoC routing approach

To address Problem 2 in Section 1.3.2, we propose a novel routing approach, called Surfing on a Bufferless NoC (Surf-Bless), presented in Chapter 6, which is based on a specific assignment and scheduling of the resources in a bufferless NoC. This specific assignment and scheduling can be visualized as multiple “waves" which move

in space and time over the NoC in a specially designed repetitive pattern. The specially designed repetitive pattern for the waves guarantees that packets "surfing" on a wave can keep moving, which is critical to correctly use a bufferless NoC to transfer packets. It is because, in a bufferless NoC, there are no buffers and packets have to keep moving. Furthermore, the specially designed repetitive pattern also guarantees that there is no interference between different waves. Thus, by assigning different domains on different waves, there is no interference between domains and a confined-interference communication is achieved. By experiments, we show that our approach is effective to support such confined-interference communication and consumes much lower energy/power than the related approach [WGO+13].

1.5 Thesis Outline

Below, we give an outline of this thesis, which summarizes the content of the following chapters.

Chapter 2 provides basic information about NoCs to make it easier to understand the contributions of this thesis.

Chapter 3 to Chapter 6 contain the main contributions of this thesis. Each chapter is organized in a self-contained way, meaning that each chapter contains more specific introduction to the problem addressed, background information, related works, the proposed solution approach, an experimental evaluation, and a concluding discussion. Chapter 3, Chapter 4, and Chapter 5 are about applying power gating on a NoC to reduce the static power consumption and address the problem of the packet latency increase caused by the power gating. Chapter 6 is about realizing confined-interference communication on a bufferless NoC.

Chapter 3 presents our novel duty buffer structure and our DB-based power gating approach. This chapter is based on our publication [WNWS17].

Chapter 4 introduces our D-bypass power gating approach. This chapter is based on our publication [WNM+19a].

Chapter 5 elaborates our EVC-based power gating approach. This chapter is based on our publication [WNM+19b].

Chapter 6 presents our novel Surf-Bless routing to realize a confined-interference communication on a bufferless NoC. This chapter is based on our publication [WNM+19c].

Chapter 7 concludes this thesis.

Chapter 2

Background

In this chapter, to better understand the contributions of this thesis, we provide some basic information about NoCs, such as network topologies, routing approaches, flow control approaches, and the basic router architecture. In order to confirm and show the major power consumer in a NoC, we also briefly analyze the power consumption on a NoC. Finally, we introduce the conventional power gating technology used in a NoC.

2.1 Network-on-Chip

In the beginning of this chapter, we start with an example of a NoC in a many-core system. As shown in Figure 2.1, this many-core system has 16 processing elements (PEs) connected to a 4×4 2D NoC. These PEs can be of different types, such as processors, digital signal processors (DSPs), peripheral controllers, memory subsystems, etc. Each PE is connected to one network interface (NI) and this NI is connected to a router in the NoC. The routers are arranged in a 2D mesh topology and connected with their neighbor routers.

A PE uses its NI to access the NoC. The NI accepts different messages from the PE and converts the messages into packets that can be transferred over the NoC. Depending on the message type, a message can be converted into one or multiple packets. A packet consists of one or multiple flits. Typically, one packet has one head flit, one or multiple body flits, and one tail flit. Some packets may have only one head-tail flit. The packets carry payload and routing information. The routing information, such as the source PE/router ID, the destination PE/router ID, etc., is stored in the head flit or the head-tail flit.

A NI injects the packets into the NoC. Based on the routing information carried in packets, the routers determine a routing path for the packet, and the packet is trans-

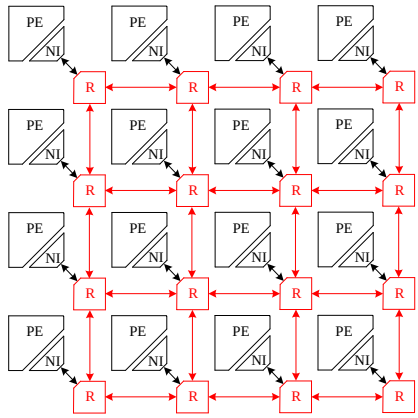


Figure 2.1: An example of a many-core system.

ferred over the routing path to its destination. When the packet reaches it destination router, the destination router ejects this packet into the corresponding NI. The NI collects several packets (if there are multiple packets for a message) and converts these packets into a message that can be processed by the PE.

When designing a NoC, a designer needs to determine the following design factors: the **network topology** (how to connect routers), the **routing approach** (which routing path should be taken to deliver the packets from a source PE/router to a destination PE/router), the **flow control approach** (how to transfer packets between routers), and the **router architecture** (how to implement the routing and the flow control approaches in hardware). Thus, we introduce these design factors in the following subsections.

2.1.1 Network Topologies

The first step in designing a NoC is to choose a proper topology. A given topology arranges the connection of routers and wires in a specific way and has a significant influence on the network performance in terms of network latency and throughput. Currently, most of the real implementations of NoCs utilize simple and regular topologies, such as a ring, a mesh, or a torus.

As shown in Figure 2.2(a), the ring is the simplest topology. The communication on a ring can be easily controlled. However, when the number of routers increases, the network latency of the ring sharply increases, which significantly limits the scalability of the ring topology. Recently, some novel ring-based topologies are proposed. For example, taking the advantage of the simple structure in a ring topology, the hierarchical ring structures [FYNM11, ZML+16] use a ring to increase the communication

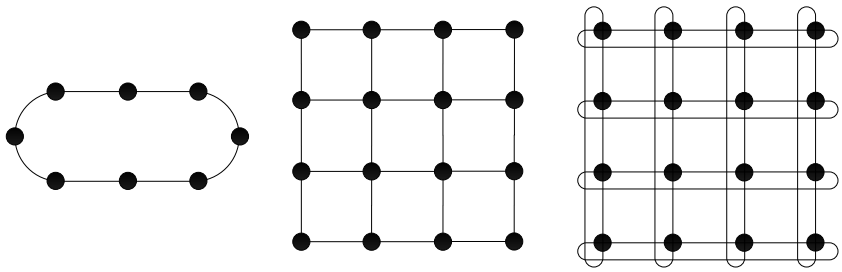


Figure 2.2: Classical network topologies.

bandwidth for the bandwidth-hungry PEs and improve the system performance at a low hardware cost. To improve the poor scalability of a ring, [AABC18, LCL+16] simply use multiple rings to fully connect a large number of routers, which shows excellent performance.

Compared with the ring topology, the mesh topology, shown in Figure 2.2(b), has better scalability and is also widely implemented on real chips. For example, the NoCs in Intel 80-tile [HVS+07], TRIPS [GKS+07], Title-64 [BEA+08], SCORPIO [DCS+14], and SW26010 [FLY+16] adopt such mesh topology. Many novel mesh-based topologys are proposed to reduce the network latency and improve the scalability, such as the flattened butterfly (FBfly) [KBD07], multidrop express channels (MECS) [GHKM09], the concentrated mesh [BD14], the express virtual channel [KPKJ07, KKC+08], the Kilo-NoC [GHKM11], etc. The main drawback of a mesh topology is that most of the traffic workload is concentrated on the routers in the center region of the mesh topology, which results in relatively low throughput of the mesh topology.

The torus topology, shown in Figure 2.2(c), overcomes the aforementioned drawback of the mesh topology by adding long wires to connect the routers on the edges. Thus, the traffic workload can be evenly distributed over the routers. Furthermore, the torus topology has a shorter network diameter than the ring topology and the mesh topology, so the torus has lower network latency. The different length of wires incurs different communication delay between routers, which makes it difficult for controlling the communication between routers. In order to remove the delay gap between the long wires and the short wires in a torus topology, many works utilize the folded torus [DT04, MJW12] topology.

As the mesh topology and the torus topology have better scalability and are widely used in real chips, we implement and evaluate our works using the tours topology in Chapter 3 and the mesh topologies in Chapter 4, Chapter 5, and Chapter 6.

2.1.2 Routing Approaches

A routing approach determines the routing path from a source router to a destination router in a particular topology. Based on the number of the routing paths used form a source router to a destination router, the routing approaches can be classified into:

- Deterministic routing approaches: packets are transferred from a source router to a destination router over exactly the same routing path. In a NoC, there exist multiple feasible routing paths from a source router to a destination router, but deterministic routing approaches always use the same routing path (determined at design-time) from a source router to a destination router, such as X-Y/Y-X dimension-order routing (X-Y/Y-X DoR) [DT01].

- Oblivious routing approaches: packets are transferred from a source router to a destination router over multiple candidate routing paths. These candidate routing paths are randomly selected without considering the state of the network, for example the Valiant's randomized routing [Val82].

- Adaptive routing approaches: based on the network state, the routing path is dynamically selected among multiple candidate routing paths to transfer packets from a source router to a destination router, such as in [FDC+09].

It is easy to implement a deterministic routing approach with a low hardware cost, so the deterministic routing approach is more common in practice, but deterministic routing may cause the problem of an unbalanced traffic workload on different routers, which undermines the resource utilization and degrades the NoC performance. In contrast, the oblivious routing [Val82] and the adaptive routing [FDC+09] are better in balancing the traffic workload on a NoC, but the implementation of oblivious routing approaches and adaptive routing approaches is much more complex than deterministic routing approaches and causes high hardware overhead.

Based on the length of the routing path, the routing approaches can be classified into minimal path routing approaches and non-minimal path routing approaches.

- Minimal path routing approaches: packets are only transferred over the minimal routing paths from a source router to a destination router.

- Non-minimal path routing approaches: packets can be transferred through non-minimal routing paths from a source router to a destination router, such as in [VB81, Val82].

Minimal path routing approaches consume less power because the packets need less hops to reach their destination routers. Thus, most of the NoCs use minimal path routing approaches. However, non-minimal path routing approaches are more promising in achieving better workload balance and realizing fault tolerance in NoCs.

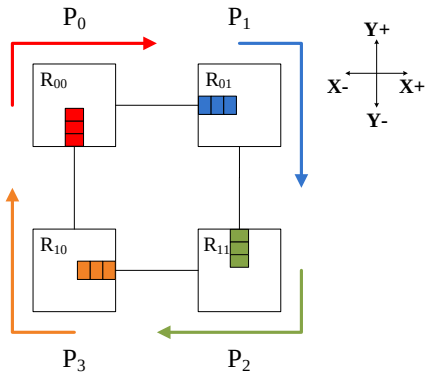


Figure 2.3: Deadlock caused by cyclic dependency of packets.

Avoiding Deadlock

One key role of routing approaches is to avoid deadlock in a NoC. Deadlock occurs when a group of packets are unable to make progress because they are waiting on one another to release resources, usually buffers or virtual channels. Consider the example shown in Figure 2.3. Packet P_0 occupies the buffers in router R_{00} and is going to router R_{01}, but the buffers in router R_{01} are occupied by packet P_1. P_0 has to wait for the buffers in R_{01} to be free. The similar situation happens for packets P_1, P_2, and P_3. As there is a cyclic dependency and all packets are waiting for others to release the buffers, none of the packets can move to the next router and deadlock occurs.

To remove the deadlock in Figure 2.3, the turn model routing algorithm [GN92] is widely used to analyze the dependency between routing paths. According to the turn model routing algorithm, there are eight possible turns in a mesh topology ($X+$ to $Y+$, $X-$ to $Y+$, $X+$ to $Y-$, and so on). These turns can be combined to create two dependent circles (the clockwise circle as shown in Figure 2.3 and the counterclockwise circle), which cause deadlock. By prohibiting some turns to eliminating these dependent circles, the deadlock can be avoid. For example, in the X-Y deterministic order routing (X-Y DoR), packets can be transferred in the $Y+/Y-$ direction only when the packet transmission in $X+/X-$ direction is finished. Thus, the packets cannot turn from $Y+$ to $X+/X-$ or from $Y-$ to $X+/X-$ in a routing path. In this way, all of the dependent circles are eliminated and the deadlock can be avoided.

In Chapters 3, 4, 5, we use the X-Y DoR in the experiments. This is because the X-Y DoR is a deterministic/minimal routing approach, which can be easily implemented and needs less hardware. Furthermore, the X-Y DoR is deadlock free by nature. In

Chapter 6, we implement our approach on a bufferless NoC, which is based on a special adaptive/non-minimal routing approach to guarantee the correctness of the bufferless NoC. This special adaptive/non-minimal routing approach is deadlock free by nature as well.

2.1.3 Flow Control Approaches

After selecting the routing approach, the designers need to choose a flow control approach to transfer packets between routers. The flow control approach determines the packet transmission behavior between routers. Efficient flow control approaches can fully utilize the NoC resources to achieve a low network latency.

Classical Flow Control

The store-and-forward, the virtual cut through, and the wormhole are the most widely used classical flow control approaches in NoCs.

Store-and-forward: A router waits until a packet has been completely received (stored) and then forwards the packet to the downstream router. In addition, the packet being transferred cannot be interrupted by other packets. Store-and-forward incurs high serialization latency [DT04] because routers need to wait for receiving the entire packet.

Virtual cut through : A router can forward a packet as soon as the head flit is received, without waiting for the entire packet to be received, but the packet being transferred cannot be interrupted by other packets. Compared with store-and-forward, virtual cut through removes the serialization latency and is more efficient.

Wormhole: Similar to the virtual cut through flow control approach, wormhole allows routers to transfer packets as soon as the head flit is received. The difference with the virtual cut through is that, in wormhole, the packet transmission can be interrupted by other packets.

Concerning resource allocation, store-and-forward and virtual cut through allocate resources (buffers and wires) at the granularity of packets, while wormhole allocates resources at the much finer granularity of flits. By allocating resources at the granularity of flits, wormhole is beneficial in reducing the amount of required buffers in a router. Furthermore, with more flexible resource allocation, wormhole is able to alleviate the packet blocking and to increase the throughput of a NoC.

Credit-based Flow Control Approaches

The store-and-forward, virtual cut through, and wormhole approaches need buffers in the routers to temporarily store packets. Thus, a means is required to communicate the availability of buffers between an upstream router and a downstream router. Then,

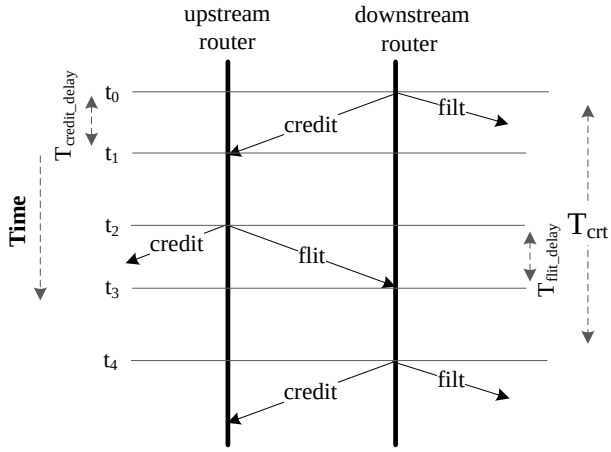


Figure 2.4: A timeline example of a credit-based flow control.

the upstream routers can determine when a buffer is available to hold the next flit (or a packet for store-and-forward and cut-through) to be transmitted. Typically, a credit-based flow control approach is used to monitor the availability of buffers in the routers.

The key idea of the credit-based flow control is that the upstream router keeps a counter of credits, which corresponds to the number of the available buffers in the downstream router. Then, each time the upstream router sends a flit to the downstream router, thereby occupying a buffer in the downstream router, the credit-based flow controller decrements the appropriate credit counter. If the counter reaches zero, all of the downstream buffers are full and no further flits can be forwarded until a buffer becomes available. Once the downstream router forwards a flit and frees the associated buffer, it sends a credit to the upstream router. When the upstream router receives this credit, the upstream router increments the corresponding credit counter to indicate that there is one more free buffer in the downstream router.

An example of the timeline of a credit-based flow control is shown in Figure 2.4. The credit counter in the upstream router is zero and all buffers in the downstream router are occupied. At time t_0, the downstream router sends a flit, thereby freeing a buffer. At the same time, a credit for this buffer is sent to the upstream router. After a short time delay for the credit transmission T_{credit_delay}, at time t_1, the upstream router receives the credit and knows that there is a free buffer in the downstream router. After a short processing delay, at time t_2, the upstream router sends a flit to occupy the free buffer in the downstream router. After a short time delay for the flit transmission T_{flit_delay}, at time t_3, the downstream router receives the flit. After a short period, at time t_4, this flit is forwarded by the downstream router, thereby freeing the buffer again

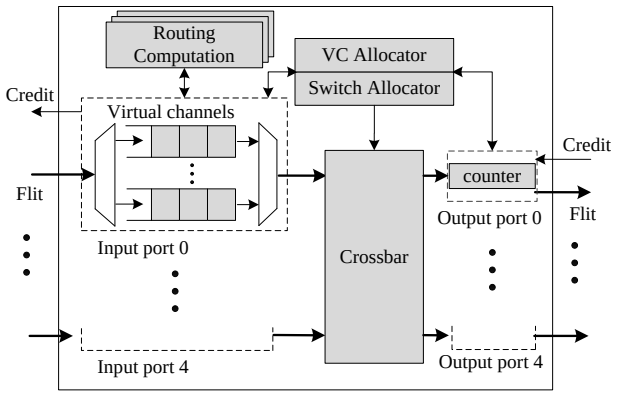


Figure 2.5: Router architecture.

and sending a credit to the upstream router. The time between sending successive credits for the same buffer is the credit round-trip delay T_{crt}. This credit round-trip delay T_{ctr} is the minimal time interval after which the same buffer can be utilized again. In order to reduce the time packets are blocked on waiting for the free buffers, routers need a large number of buffers to hold packets. In this way, the credit round-trip delay can be hidden.

In Chapters 3, 4, 5, we use the wormhole flow control approach and the credit-based flow control approach to control the communication between routers. This is because the wormhole flow control approach needs less buffers than the store-and-forward flow control approach and the virtual cut through flow control approach. Furthermore, the credit-based flow control is the most widely used approach to monitor the availability of buffers in routers. In Chapter 6, we implement the confined-interference communication on a bufferless NoC, in which the flow control is completely different compared to the aforementioned flow control approaches. We will introduce it in Chapter 6.

2.1.4 Router Architecture in NoCs

The router implements the routing approach and the flow control approach in hardware. In this section, we introduce the basic router architecture and the router pipeline.

Router Micro-architecture

As shown in Figure 2.5, a router consists of input ports, routing computation units, a virtual channel allocator unit (VC Allocator), a switch allocator unit, a crossbar, and

output ports.

- **Input ports** are mainly used to temporarily hold packets. In each input port, there are multiple buffer queues (first in first out, FIFO), called virtual channels (VCs). These virtual channels are used for several other purposes, such as avoiding the Head-of-Line blocking [DT01], hiding the credit-round trip delay [DT01], and constructing multiple virtual network, etc. In order to go to the correct VC, each flit of a packet has to carry a VC address, which is used to indicate the VC that the flit should be stored. When a flit of a packet goes into a router, based on the VC address in the flit, the input port stores the flit into the corresponding VC. When a flit of a packet leaves the VC, the input port releases one buffer and sends a credit to inform the upstream router which VC in the downstream router has released one buffer.

- **Routing Computation** unit implements the routing approach for the packets. When there is a head flit of a packet coming, the routing computation unit determines the output port for the packet according to the implemented routing algorithm. In practice, the routing computation unit is separately implemented in each input port to simultaneously compute the output ports for multiple incoming packets.

- **VC Allocator** is a $(N_{upIP} \times N_{upInVC}) \rightarrow (N_{upOP} \times N_{downInVC})$ mapper, where N_{upIP} is the set of input ports in an upstream router; N_{upInVC} is the set of VCs of an input port in the upstream router; N_{upOP} is the set of output ports in the upstream router; $N_{downInVC}$ is the set of VCs of input ports in the downstream routers. When there is a new packet coming, the VC allocator allocates a free VC in the downstream router to the packet. The flits of the packet carry this VC address to the downstream router. When the flits of the packet reach the downstream router, based on this VC address, the downstream router stores the flits of this packet into the corresponding VC. Typically, the VC allocator unit has the most complex hardware logic in a router. The hardware critical path is through the VC allocator. Thus, the complexity of the VC allocator unit limits the operating frequency of a router.

- **Switch Allocator** is a $(N_{upIP} \times N_{upInVC}) \rightarrow N_{upOP}$ mapper. The switch allocator grants the packets to use the crossbar and solves the conflict between multiple packets contending for the same output port. As we use the wormhole flow control, each flit of a packet can go to the downstream router only when the flit gets the grant from the switch allocator.

- **Crossbar** is used to transfer the flits of a packet from an input port to an output port.

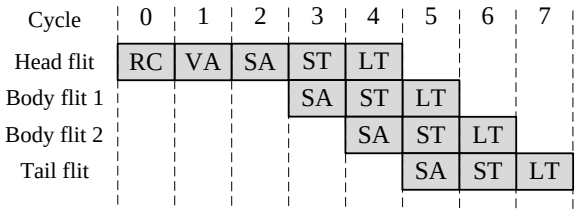


Figure 2.6: Router pipeline.

- **Output ports** contain credit counters, which are used to record the free buffers in each VC in the downstream router. As we have introduced the credit-based flow control in Section 2.1.3, each credit corresponds to a free buffer in the downstream router. When a flit of a packet leaves the router, the credit counter in the output port is decremented to indicate that a buffer space in the downstream router will be occupied. When the output port receives a credit from the downstream router, the output port increments the credit counter, which indicates that there is one buffer released in the downstream router.

In order to transfer packets, a router opperates in several pipeline stages. In a conventional router, there are four pipeline stages: routing computation (RC), virtual-channel allocation (VA), switch allocation (SA), and switch traversal (ST). A packet may experience one more stage called link traversal (LT) in the wires. The pipeline stages in a conventional router are shown in Figure 2.6.

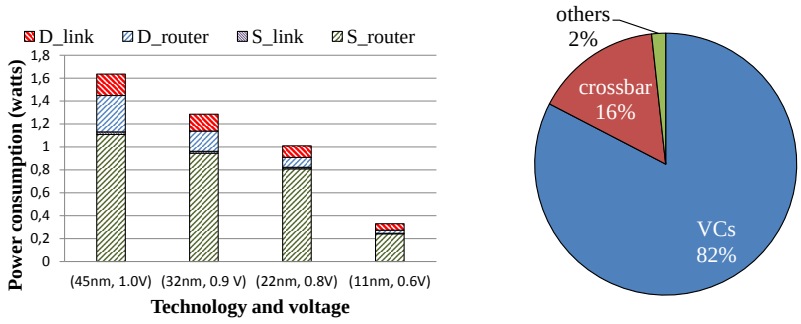
Route computation (RC) stage: The routing computation unit selects a suitable output port according to the routing algorithm. This stage only needs to be performed for the head flit of each packet. The rest of body flits and the tail flit follow the head flit to the same output port.

VC allocation (VA) stage: After the RC stage, the head flit of a packet needs to experience the VA stage to get a free VC in the downstream router. Similar to the RC stage, the VA stage only needs to be performed for head flits, and the rest of the flits in the packet inherit the VC address allocated to the head flit.

Switch allocation (SA) stage: After a packet has been assigned an output port in the current router and a VC in the downstream router, each flit of the packet sequentially requests for permission the switch allocater unit to use the crossbar. The switch allocator unit solves the contention for the same output port between multiple packets.

Switch traversal (ST) stage: After receiving a grant from the switch allocator unit, a flit can traverse the crossbar in the next cycle to reach its destination output port.

Link traversal (LT) stage: Finally, the flit of a packet goes through the links to



(a) Power consumption under different technologies and voltages.

(b) Static power consumption in a router.

Figure 2.7: Power consumption in a 8×8 2D mesh NoC.

the downstream router.

The aforementioned pipeline stages can be hidden or overlapped. For example, based on the Look-ahead routing [Gal97], the RC stage for the downstream router can be executed ahead in the upstream router. When the packet reaches the downstream router, the packet directly goes to the VA stage. Thus, one pipeline stage can be hidden. However, the look-ahead routing is only feasible for the deterministic routing approach, in which the routing path of a packet can be easily determined. Based on the speculative routing [PD01], the VA stage and the SA stage can be performed in parallel. The SA stage for the head flit of a packet is speculative because it depends on the success of the VA stage being performed at the same time. If the VA stage fails, the SA stage will be ignored even if it succeeds.

2.2 Power Consumption Analysis

In order to confirm and show the major power consumer in a NoC, in this section, we briefly analyze the power consumption of each component in a NoC.

We use Dsent [SCK+12] to evaluate the power consumption of a 8×8 2D mesh NoC. Each input port of a router has two 4-flit VCs and the wire bandwidth is 128 Gbits/s. The packet injection rate is 0.1 flits/node/cycle and the flit size is 128 bits. This injection packet rate is much higher than what can be observed in most of the real applications [DMMD09]. This NoC works at the frequency of $1GHz$. We evaluate the power consumption across different technologies and voltages; $(45nm,\ 1.0V)$, $(32nm, 0.9V)$, $(22nm, 0.8V)$, and $(11nm, 0.6V)$.

Figure 2.7(a) shows the power consumption under different pairs of technologies

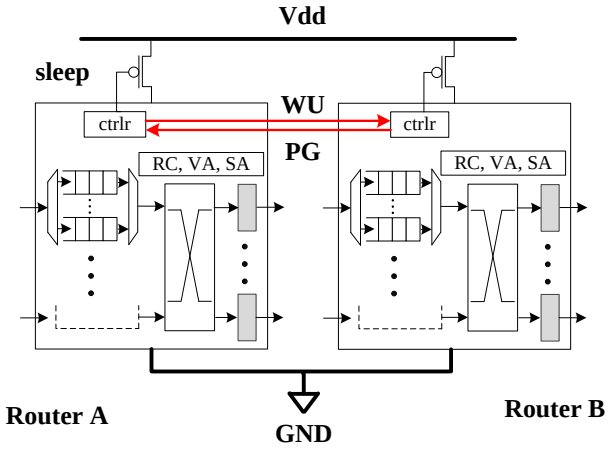


Figure 2.8: Conventional NoC power gating.

and voltages. The total power consumption of a NoC is broken down into four parts: the dynamic power consumption of the link and the routers (D_link and D_router), and the static power consumption of the link and the routers (S_link and S_router). With the downscaling of the technology and the voltage, the total power consumption of a NoC is reduced. However, most of the power consumption is contributed by the static power consumption of routers. For example, the static power consumption of the routers takes 67.78%, 73.55%, 80.17%, and 73.39% of the total power consumption. Thus, in order to reduce the total power consumption in a NoC, the critical point is to reduce the static power consumption of the routers.

In Figure 2.7(b), we show the static power consumption contributed by each component in a router under $(45nm, 1.0V)$. VCs consume the most of the static power. The crossbar and the other components only consume 16% and 2% of the total static power, respectively. Furthermore, in this evaluation, we use two 4-flit VCs in each input port, which is less than what is used in most of the real-word NoCs listed in Table 1.1. So, if the number of VCs further increases, the VCs will consume even more static power. Thus, to reduce the total power consumption, it is crucial to reduce the static power consumption of VCs.

2.3 Conventional Power Gating in A NoC

As we have shown in Section 2.2, the static power consumption takes large portion of the total power consumption. Power gating is an effictive way to reduce the high static power consumption of a NoC. An implementation example of applying conventional power gating on the routers is shown in Figure 2.8. By inserting header transistors

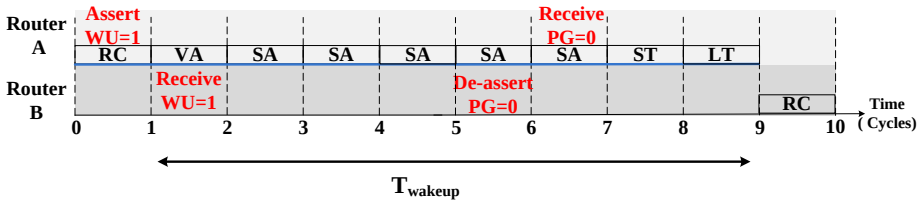


Figure 2.9: Wakeup process.

between the voltage supply and the router, the power controller (the ctrlr unit in Figure 2.8) can cut off the power supply of the router to reduce the power consumption. In order to correctly control the packet transmission, additional handshaking control signals WU (wakeup) and PG (power gating) are added between routers.

When $RouterB$ is idle (there are no flits left in input ports or the crossbar) and the WU signals are clear, the controller in $RouterB$ asserts the sleep signal to cut off the router's power supply (The ctrl unit in routers is always powered-on. Besides, the flow control units that contain the credit counters in output ports are also always powered-on.) and asserts the PG signal to notify its upstream $RouterA$. Once $RouterA$ receives the signal PG, $RouterA$ marks the output port to $RouterB$ as being powered-off. When $RouterA$ needs to send a packet to $RouterB$, $RouterA$ has to assert the WU signal to wake up $RouterB$ and waits for $RouterB$ to be fully charged. Once $RouterB$ is fully charged, the PG signal is de-asserted and $RouterA$ can send the packet to $RouterB$.

An optimized wakeup process [MKWA08, CZPP16] is shown in Figure 2.9. When $RouterA$ executes the RC stage for packets, $RouterA$ determines that there is a packet going to $RouterB$ and asserts the WU signal to wake up $RouterB$. In the following clock cycles, $RouterA$ executes the VA stage and the SA stage, but as $RouterB$ is powered off, the packet has to be blocked in $RouterA$. Once the WU single is received, the ctrlr unit in $RouterB$ clears the sleep signal to charge $RouterB$. After experiencing $T_{wakeup} - MARGIN$ ($MARGIN = 4$ in this example) clock cycles, $RouterB$ de-asserts the PG signal. When $RouterA$ is aware that the PG signal is de-asserted, $RouterA$ allows the packet to go to $RouterB$ and executes the ST stage and the LT stage to transfer the packet. When the packet reaches $RouterB$, $RouterB$ is just fully charged.

Chapter 3

Duty Buffer Based Power Gating Approach

Peng Wang, Sobhan Niknam, Zhiying Wang, Todor Stefanov,
"A Novel Approach to Reduce Packet Latency Increase Caused by Power Gating in Network-on-Chip"
in Proceedings of the 11th IEEE/ACM International Symposium on Networks-on-Chip (NOCS), Seoul, South Korea, 2017, pp. 1–8.

In this chapter, we present in more detail our duty buffer based (DB-based) power gating approach, which corresponds to **Contribution 1** introduced in Section 1.4, to solve the research **Problem 1**, described in Section 1.3.1. This chapter is organized as follows. Section 3.1 further introduces the research problem of the packet latency increase caused by power gating. It is followed by Section 3.2, which gives a summary of the contributions in this chapter. Then, Section 3.3 gives an overview of the related work. Section 3.4 elaborates on our novel duty buffer (DB) structure and the DB-based power gating approach. Finally, Section 3.5 introduces the experimental setup and shows the results, and Section 3.6 concludes and discusses this chapter.

3.1 Problem Statement

As we have shown in Section 1.1.1, the NoC accounts for 10% to 36% of the total power consumption in many-core systems. Due to the low average traffic load of real applications [DMMD09], most of the NoC power consumption is contributed by the static power consumption of idle routers. Even, under a minimal resource configuration, the NoC router static power consumption is still about 64% [CZPP15] of the total NoC power consumption. As a consequence, it is critical to reduce the high static power consumption of the routers.

On the other hand, NoCs have the characteristics of a distributed structure, a naturally unbalanced traffic workload, and a low average injection traffic rate, which make power gating being an applicable and effective way of powering off idle NoC routers to reduce the power consumption. However, there is a notable wakeup delay to power on the powered-off components during the wakeup process. When power gating is applied on a NoC, this wakeup delay, about 6-12 clock cycles [CZPP15], will interrupt the close cooperation between routers and will block the routing path for a while. As a consequence, the packet latency over the whole routing path dramatically increases and the NoC performance is degraded.

Several works [ZOG+15, MKWA08] try to reduce the packet latency increase caused by the power gating. As the drowsy SRAM has only 2 clock cycles wakeup delay, Zhan [ZOG+15] uses the drowsy SRAM to build virtual channels (VCs) in a NoC router and implements a fine-grained power gating on virtual channels. In this way, the wakeup process becomes faster. By sending a wakeup signal ahead of the packet injection, Matsutani [MKWA08] switches on the power of the powered-off routers earlier, thereby hiding part of the wakeup delay. These approaches [ZOG+15, MKWA08] can reduce or hide part of the wakeup delay in a single wakeup process. But a packet may experience multiple wakeup processes and accumulate large delay along the routing path. In fact, as the average traffic load of real applications is low [DMMD09], there is high probability for packets to experience multiple wakeup processes. Furthermore, with more cores integrated on a chip in future many-core systems, this cumulative wakeup delay becomes much higher.

3.2 Contributions

In order to further reduce the packet latency increase caused by the power gating, in this chapter, we propose a novel and flexible hardware structure, called Duty Buffer (DB). Based on the DB structure, we propose a DB-based power gating approach to reduce the static power consumption of the VCs in routers. By using our DB to temporarily replace a powered-off VC and accept packets, an upstream router does not need to block packets while waiting for the VC in a downstream router to completely wake up. Thus, we can efficiently reduce the packet latency increase. Furthermore, as all VCs in the same input port of a router share the same DB, we can keep a minimal number of DBs powered on (on duty) and power off all of the VCs. In this way, we use minimal static power consumption to keep a certain transmission ability, which is helpful to reduce the static power consumption. The specific novel contributions of this chapter are the following:

- We propose a novel Duty Buffer structure, which can be used to replace any VC in a router. Taking the advantage of our novel DB, we propose a novel

DB-based power gating approach on VCs in a router. This approach efficiently reduces the packet latency increase caused by the power gating. By keeping a minimal number of DBs powered on, our DB-based approach also achieves a significant reduction of the static power consumption in a NoC.

- By experiments, we show that our DB-based power gating approach can effectively reduce the packet latency increase caused by the power gating. Taking a conventional router without power gating as the baseline, our DB-based power gating approach, with one flit depth of DB, increases the average packet latency only by 9.67%, which is much less than the 57% latency increase in [MKWA08] and 21.75% in [ZOG+15]. Compared with the based in terms of the power consumption, our DB-based approach can save on average 52.19% of the total power consumption, which is comparable with the 59.39% saving in [MKWA08] and 57.05% in [ZOG+15].

3.3 Related work

In this section, we discuss the related works on reducing the packet latency increase caused by power gating in a NoC.

As VCs are the main source of the static power consumption in a router, Zhan [ZOG+15] applies the power gating on the VCs and uses the drowsy SRAM [FKM+02] to build the VCs because the wakeup delay of the drowsy SRAM is only two clock cycles, thus the wakeup process is much faster. However, this approach cannot completely remove the wakeup delay and packets accumulate large wakeup delay along the whole routing path. In contrast, our approach keeps a certain transmission ability of routers when the VCs are powered off. The packets are not blocked by the powered-off VCs in the routers. Thus, even though a packet experiences multiple wakeup processes along the routing path, the packet will not accumulate large wakeup delay. Therefore, our approach can reduce more efficiently the packet latency increase.

Based on the principle of the look-ahead routing, Matsutani [MKWA08] proposes a runtime power gating approach. By sending a wakeup signal into the next router ahead of the packet injection, this approach hides a few clock cycles in the wakeup process and wakes up the powered-off router earlier. However, the number of hidden clock cycles is determined by the number of the router pipeline stages, and is insufficient to cover the entire wakeup delay. By sending the wakeup signal to the rest of the routers along the routing path, Chen [CZPP15] improves the approach in [MKWA08]. In this way, the powered-off routers can be waked up much earlier and this approach achieves almost non-blocking power gating in deterministic routing. However, this approach highly depends on the correct prediction of the routing path. If the routing path of a packet is adaptively changed, this approach may not find the correct routers

to power on. Compared with [MKWA08, CZPP15], the power gating mechanism in our approach does not need the handshaking control signals (the WU signal and the PG signal in Figure 2.3) between routes to control the packet transmission. By keeping a certain transmission ability of routers when the VCs are powered off, routers in our DB-based approach still can transfer packets during the wakeup process, which can be efficiently used to significantly reduce the overall packet latency increase in both deterministic routing and adaptive routing.

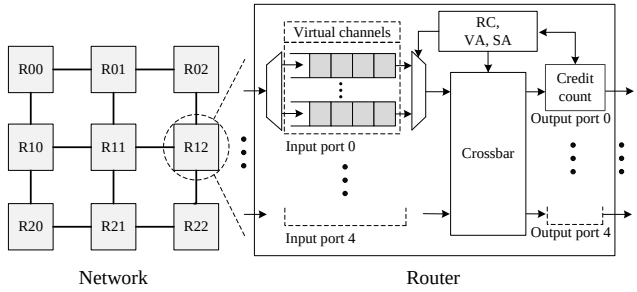
Kim proposed in [KKY11] the Flexibuffer scheme to reduce the static power consumption of the VC buffers. Based on the buffer occupancy rate, the router predictively powers on the sleeping buffers in the VCs in advance. In this way, the negative impact of the power gating on the packet latency increase is reduced. However, the scheme in [KKY11] can reduce the static power consumption, only when the VC size is large enough. This is because each VC in this approach has to keep at least $b_{min} = max(N_{wakeup}, N_{crt})$ (N_{wakeup} and N_{crt} are the number of clock cycles of the wakeup delay and credit round-trip delay, respectively) buffers powered on at run-time, which makes the flexibuffer scheme not very efficient in reducing the static power consumption. Furthermore, considering the high wakeup delay, 6-12 clock cycles [CZPP15], for NoCs with small size of VCs, such as the NoCs in [DCS$^+$14, ZOG$^+$15, CZPP15, YL16, DNSD13], the scheme in [KKY11] cannot save any power consumption. Compared with the scheme in [KKY11], the DB in our approach is shared with all VCs in the same input port, and the minimal DB size is one flit. In this way, our approach can be widely and efficiently used to reduce the static power consumption in NoCs with small and large size of VCs.

3.4 DB-based Approach

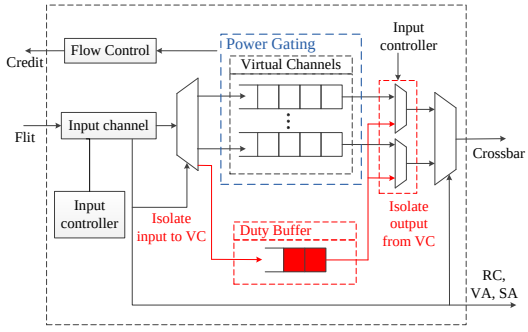
The key idea of our DB-based power gating approach is, by always having a small number of buffers powered on (on duty) in VCs, to keep a certain transmission ability of a router in the wakeup process when power gating is used. In this way, we can reduce the packet latency increase caused by the power gating. In order to achieve this goal, we have to overcome the following challenges.

- **Which VCs should be on duty?** As we have introduced in Section 2.1.4, the input VC address of a packet, which indicates where the packet will be stored, is determined in an upstream router. Therefore, a downstream router does not know when packets will arrive or which VCs will be occupied. As a consequence, it is unknown which VC should be on duty in a downstream router.

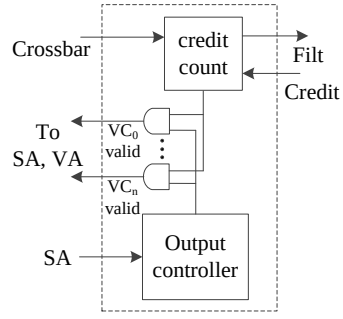
- **How to use as few buffers on duty as possible?** Keeping fewer buffers on duty in a VC is helpful to reduce the static power consumption in a NoC. However,



(a) Architecture of a NoC



(b) Extended input port



(c) Extended output port

Figure 3.1: The NoC structure and extended input port and output port.

the role of the VCs is not the same. Different classes of VCs are used to receive different packets. For example, the MESI [PP84] coherence protocol needs at least two different data VCs and one control VC to prevent a deadlock. If we want to keep some buffers in VCs on duty to reduce the packet latency increase, each VC class should have buffers on duty and ready to receive packets. As a consequence, we cannot efficiently reduce the static power consumption.

3.4.1 Input Port with Duty Buffer

In order to overcome the challenges, mentioned above, we propose the novel Duty Buffer structure shown in Figure 3.1(b) to extend the input ports of a virtual-channeled wormhole router. Compared with the conventional router in Figure 3.1(a), which is a simplified version of Figure 2.5, the following components are added to an input port: Input controller, Duty Buffer, and multiplexers. We apply power gating on the VCs as well. The Duty Buffer (DB) is a small buffer queue. The minimal size of the DB is one flit. It is always powered on and ready to receive packets at run-time.

For a downstream router, when a packet reaches an input port, but input VCs are powered-off or waking up, the input controller controls the input channel and demultiplexer to store the packet into the DB. The DB replaces the corresponding powered-off VC, which the packets should go into.

When the router tries to read the packets from the powered-off VC, the input controller controls the corresponding multiplexer to select reading the packets from the DB. By replacing the powered-off VC with DB, the router can transfer the packet as if this VC was powered on.

For an upstream router, as its corresponding downstream router can use the DB to replace the powered-off VCs to store packets, there is no need for the upstream router to block the packet and to wait for the VCs in the downstream router to completely wakeup. Thus, the upstream router still can send a packet to the downstream router, when the VCs in the downstream router are powered-off or waking up. As a result, the packet latency increase caused by the power gating on VCs is reduced.

Since the DB can replace any powered-off VC, we do not need to determine which VC should be on duty. Furthermore, as all VCs in an input port share the same DB, we do not need to keep powered-on buffers for each class of VCs. Thus, we can keep as few buffers on duty in the DB as possible, and power off as many buffers in VCs as possible. In this way, we keep a certain transmission ability with minimal static power consumption.

3.4.2 Power Gating on VCs

In our extended input port, as shown in Figure 3.1(b), the input controller uses one switch to control the power of all VCs in the port. This is because powering on all VCs at the same time is beneficial to guarantee the NoC performance. That is, the traffic load in many-core systems is bursty [DMMD09], so routers tend to use a larger number of VCs in an input port at the same time. On the other hand, the average traffic load is very low [DMMD09]. This makes a higher percentage of input VCs idle during the applications execution. So, simultaneously powering off all VCs also can reduce significantly the static power consumption.

The rest of the components in a router, for instance, the routing computation unit, the virtual channel allocation unit, the switch allocation unit, the crossbar and the output ports, are always powered on. In this way, if VCs in some input ports are powered off, the other input ports still can normally work. Packets will not be blocked in the router pipeline. Furthermore, compared with VCs, the power consumption in the rest of the router's components is much lower [CZZ+15]. Always keeping them powered on, routers will not waste much static power.

3.4.3 Power Gating Scheme

In order to correctly use our DB to reduce the latency increase during the waking up process, we have to achieve the following goals:

- **Keeping the flits order in a packet.** The flits of a packet may be separately stored in the DB and a VC when the state of VCs switches from waking up state to charging complete state. In order to keep the packet transmission correct, the transmission order of the flits in the same packet must not be changed.
- **No deadlock occurs.** In general, different class packets are allocated with different VCs. This is because a network interface has to finish processing the reply packet first, then to deal with the new request packet. If a request packet and a replay packet are stored in the same buffer queue (FIFO), the request packet may be stored in front of the reply packet and it stalls the reply packet to access the network interface. As a consequence, the request packet and the replay packet block each other and the deadlock occurs [DT04]. Therefore, in the wakeup process, we have to guarantee that only packets with the same VC address enter the input port to prevent deadlock occurrence. This VC address should be determined by the head flit of the packet, which wakes up the powered-off VCs in the downstream router.

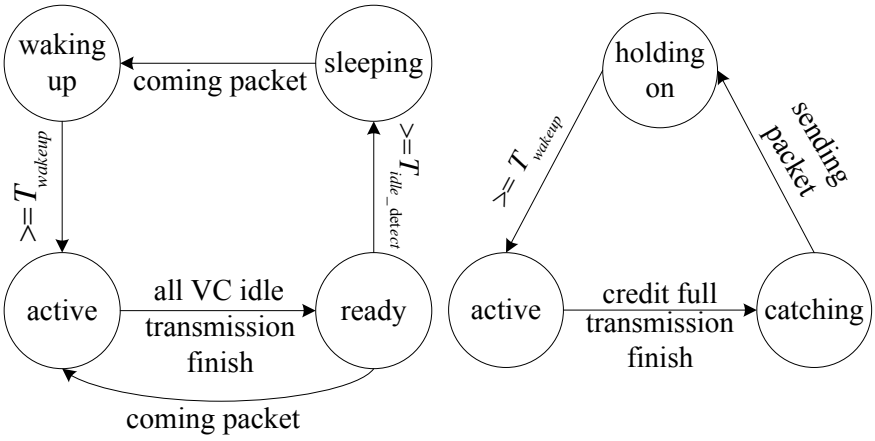
In order to achieve the aforementioned goals, the input controller and the output controller are added to a router, as shown in Figure 3.1(b) and Figure 3.1(c). In the following subsections, we introduce the working mechanisms of the input controller and the output controller.

Input Controller

In the extended input port, as shown in Figure 3.1(b), the input controller monitors the VC control to determine if it can switch off the power of VCs. The states of the input controller are shown in Figure 3.2(a).

In the **active** state, VCs in the input port are powered on. The input controller controls the input channel to inject packets into the corresponding VCs and keeps the DB idle. When all VCs and the DB are empty and the packet transmission is completed (the tail flit of the packet has left), the input port controller moves to the **ready** state and waits for T_{idle_detect} clock cycles to switch off the power of the VCs. We use the equation, $T_{idle\_detect} = T_{credit\_delay} + T_{flit\_delay}$, to compute the T_{idle_detect}, where T_{credit_delay} and T_{flit_delay} are the credit and flit transmission delaies between neighbor routers, as shown in Figure 2.4. and they are dependent on design parameters.

In the **ready** state, the VCs are still powered on and can be used to store packets. If there are incoming packets in the T_{idle_detect} period, the packets will be stored to



(a) Input controller states (b) Output controller states

Figure 3.2: Controllers for input and output ports.

the corresponding VC and the input controller returns back to the **active** state. The **ready** state is used to avoid unnecessary power gating activities when the idle time of VCs is short.

In the **sleeping** state, the power of all VCs is switched off and the VCs cannot be used to store packets. Once the head flit of a packet comes into the input port, it will be stored in the DB and the power of VCs will be switched on to charge the circuit. The input port goes into the **waking up** state.

In the **waking up** state, VCs are not stable and cannot accept packets. The incoming packets still are stored in the DB. After T_{wakeup} clock cycles, the VC charge is completed, and the state changes to the **active** state and the VCs can be used to store packets. The input controller stores the packets to the corresponding VC and stops injecting packets into the DB.

At the beginning of the **active** state, if there are any flits left in the DB, the input controller keeps replacing the output of the corresponding VC with the output of the DB until the DB is empty. In this way, the order of the flits in a packet will not be disturbed and the correct transmission is guaranteed.

Output Controller

As explained in the beginning of Section 3.4.3, to guarantee deadlock-free packet transmission during the wakeup process, an output port in a router has to send only packets with the same VC address to the downstream router. In order to achieve this,

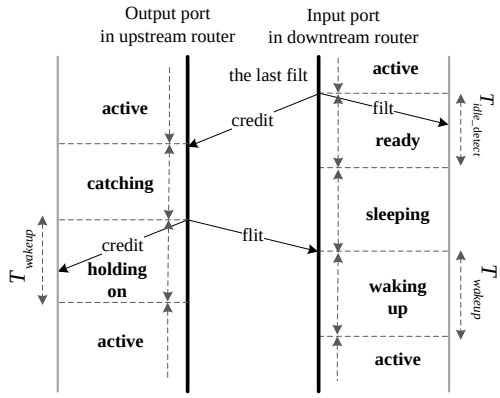


Figure 3.3: The interactive process between an output port in the upstream router and the input port in the corresponding downstream router.

the output controller and some AND gates are added to the output port, as shown in Figure 3.1(c). In this extended output port, the output controller monitors the credit counter and the grant to use the crossbar from the switch allocator unit, to identify the state of the input port in the downstream routers and controls the validation signals for the virtual channel allocator unit and the switch allocator unit, indicating which input VCs in the downstream router are available. The states of the output controller are shown in Figure 3.2(b).

In the **active** state, the credit counter is not full, or the packet transmission is not completed (the tail flit has not left). This indicates that the VCs in the corresponding input port in the downstream router are powered on. So, the upstream router can normally send packets and receive credits.

When the credit counter is full and packet transmissions are completed, the output controller moves to the **catching** state. In the **catching** state, the output controller considers that the VCs of the corresponding input port in the downstream router are powered-off, even if, at this time, this input port may be in the **ready** state. The output port allows the virtual channel allocator unit to normally allocate VCs and the switch allocator unit to grant the use of the crossbar in the **catching** state.

Once a head flit of a packet is granted by the switch allocator unit to use the crossbar, the output controller marks the allocated VC address and moves to **holding on** state. In the following T_{wakeup} clock cycles, only packets allocated to this marked VC address are allowed to use the crossbar and transferred to the downstream router. In the **holding on** state, the output controller assumes that all the flits sent are stored in the DB of the input port in the corresponding downstream router. The output con-

Table 3.1: Parameters.

topology	2D tours
router pipeline	4-stage
network size	4×4
VC count	4 VCs per port
buffer depth	4 flits per VC
packet size	1 or 8 flits, 8B/flit
T_{credit_delay}	1 clock cycle
T_{flit_delay}	1 clock cycle
BET	10 clock cycles
T_{wakeup}	10 clock cycles

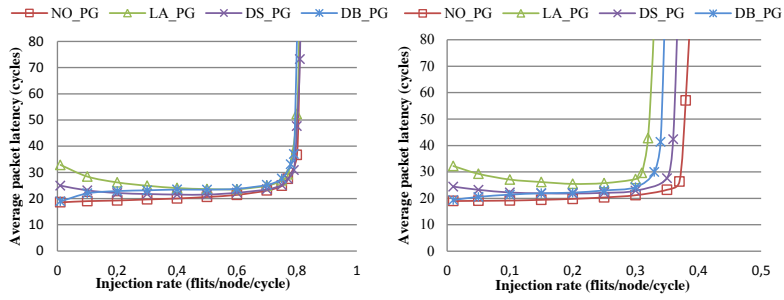
troller guarantees that the used credits does not exceed the depth of the DB. In this way, the output controller can guarantee that there is no buffer overflow in the wakeup process. After T_{wakeup} clock cycles, the output controller moves to the **active** state. The virtual channel allocator unit and the switch allocator unit can normally allocate router resource for the packets.

In better understand the interactive process between the upstream router and the downstream router, we use Figure 3.3 to the state switching in the output controller and the input port controller.

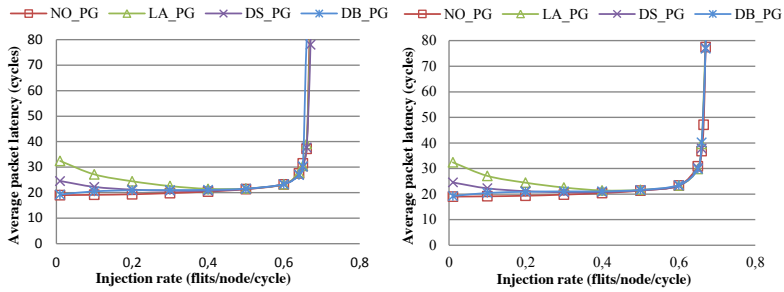
3.5 Experimental Results

In order to evaluate our approach in terms of performance and power consumption, we have implemented our approach on the cycle-accurate interconnection network simulator Booksim2.0 [JBB+13]. The parameters set in Booksim2.0 are shown in Table 3.1. Each input port of a router has four 4-flit depth VCs. The total number of buffers in a router is consistent with [DNSD13]. According to the prior works [DNSD13, ZOG+15] , the credit transmission delay T_{credit_delay} and the flit transmission delay T_{flit_delay} are set to 1 clock cycle. Based on the prior work in [DNSD13], T_{wakeup} is set to 10 clock cycles. To calculate the power consumption cost of power gating when the power of VCs is switched on and off, we set the Break even time (BET), which is the number of sleep cycles required to compensate the energy overhead for charging VCs in a power gating process. The BET is 10 clock cycles in our experiments and it is consistent with the prior work in [ZOG+15].

For comparison purpose, we have implemented the following schemes in Booksim2.0: (1) NO_PG is the baseline NoC. It uses the conventional four-stage pipeline router introduced in Section 2.1.4 without power gating; (2) LA_PG is a NoC using the lookahead scheme [MKWA08] to hide 4 clock cycles of the wakeup delay; (3) DS_PG is a NoC where the VCs of a router are implemented by the drowsy SRAM [ZOG+15].



(a) Uniform (b) Transpose



(c) Bitcomp (d) Tornado

Figure 3.4: Average packet latency across full range of workloads.

In the active state, the drowsy SRAM has the same power consumption as the conventional SRAM. When staying in the drowsy state, it only consumes about 10% of the SRAM static power consumption and has 2 clock cycles wakeup delay; (4) DB_PG is a NoC using our DB-based approach, presented in Section 3.4, with different depths of the DB.

3.5.1 Evaluation on Synthetic Workloads

In order to explore our DB_PG behavior under a wider range of packet injection rates, in this section, we evaluate the performance of NO_PG, LA_PG, DS_PG and our DB_PG under synthetic traffic patterns. Booksim2.0 provides abundant synthetic traffic patterns. We select four synthetic traffic patterns: 1) Uniform random: packets' destinations are randomly selected; 2) Transpose: packets from source node (x, y) are sent to destination node (y, x); 3) Bitcomp: packets from (x, y) are sent to $(N - x, N - y)$, N is the number of nodes in the X and Y dimensions of a NoC; 4) Tornado: packets from (x, y) are sent to $(x + \frac{N}{2} - 1, y)$. In this experiment, 1-flit

packets are injected to the NoCs. Figure 3.4 shows the average packet latency curves with different injection rates under the different traffic patterns. Since DB_PG with different depths of the DB has similar trend in terms of the average packet latency curve, in order to clearly show the experimental results, in Figure 3.4, we only show our DB_PG with DB of depth 1.

The zero-load latency is an important performance indicator for a NoC. As shown in Figure 3.4, when the injection rate is almost zero (the injection is about 0.01 packets/node/cycle), where the packet latency is close to the zero-load latency, our DB_PG has less packet latency than LA_PG and DS_PG. This is because, when the injection rate is low, most of the input ports of routers are idle. A packet experiences multiple wakeup processes along the routing path. As a consequence, a packet in LA_PG and DS_PG accumulates a large wakeup delay. In contrast, by keeping a certain transmission ability in the wakeup process, our proposed DB_PG can avoid the situation where a packet accumulates too much latency.

With the injection rate increasing from 0 packets/node/cycle to about 0.2 packets/node/cycle, as shown in Figure 3.4, our DB_PG still has less packet latency increase than LA_PG and DS_PG. However, the packet latency in our DB_PG slowly increases, while the packet latency in LA_PG and DS_PG decreases. This is because, with the injection rate increasing, more and more input ports of routers are always busy and cannot be powered off. The probability of accumulating a larger wakeup delay becomes lower. On the other hand, multiple packets may compete for a transfer to a downstream router in the same wakeup process, but our DB only can replace one VC in a signal wakeup process in order to avoid deadlock occurrence. As a consequence, in the wakeup process, some of the packets are blocked in our DB_PG.

When the injection rate further increases and is close to the saturation injection rate where the packet latency sharply increases, as shown in Figure 3.4(a) and Figure 3.4(b), the packet latency in the LA_PG and DB_PG is higher than that in DS_PG, because, in this situation, the wakeup delay in a single wakeup process becomes the main reason for the packet latency increase. The LA_PG and DB_PG have higher wakeup delay than DS_PG. However, because of the DB structure, our DB_PG outperforms LA_PG.

Based on the results of the synthetic traffic patterns, when the injection is below 0.2 packets/node/cycle, our DB_PG efficiently prevents a packet to accumulate a large wakeup delay along the routing path and achieves better performance than LA_PG and DS_PG. However, when the injection rate is close to the saturation injection rate, in some traffic patterns, our DB_PG causes a little bit performance loss in terms of the average packet latency. Considering that the injection rate of real applications [DMMD09] is much lower than 0.2 packets/node/cycle, our DB_PG can be widely used to reduce the packet latency increase caused by the power gating.

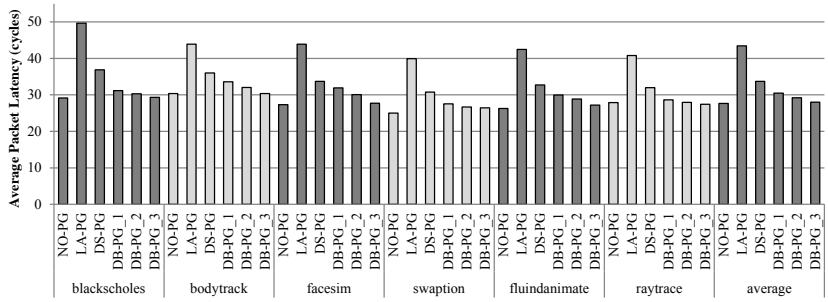


Figure 3.5: The average packet latency.

3.5.2 Evaluation on Real Application Workloads

In order to evaluate our DB-based approach and compare it with the other approaches mentioned above, on real workloads of applications, we use six applications from the Parsec [BKSL08] benchmark suite: blockscholes, bodytrack, facesim, swaption, fluindanimate, and raytrace. We use Synfull [BJ14] to capture the traffic behaviour of these applications. Synfull generates packets and feeds packets to Booksim 2.0 to evaluate the network performance. Based on the collected data from Booksim2.0, we use Dsent [SCK$^+$12] to compute the power consumption under the 45nm technology, the voltage of $1V$, and the NoC frequency of $1GHz$.

Effect on Performance

Figure 3.5 shows the average packet latency for the six different real application workloads. The seventh set of bars in Figure 3.5 gives the average result over these six applications.

We use the average packet latency to measure the network performance. The LA_PG hides 4 cycles of the wakeup delay by sending the wakeup signal in advance. However, compared with the baseline NO_PG, it still incurs an average of 57% (about 16 cycles) packet latency increase throughout these six applications. The latency in the DS_PG also increases by 21.75% (about 6 cycles) on average, even though the wakeup delay in the DS_PG is only 2 clock cycles. These results indicate that packets experience over three wakeup processes on average and the cumulative wakeup delay is the main reason for the performance degradation.

In contrast, the average packet latency in our DB_PG only increases by 9.67%, 5.67%, and 2.02% with 1, 2, and 3 flits depth of the Duty Buffer, respectively. Compared with the LA_PG and the DS_PG, our DB_PG approach, with only one flit depth of the DB, achieves 47.33% and 12.08% less packet latency increase, respectively.

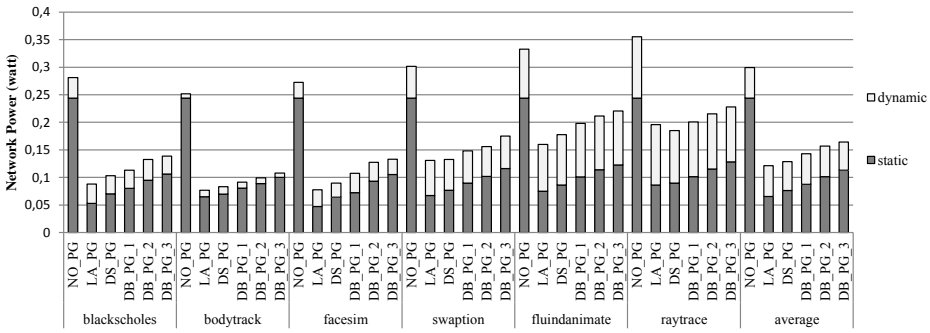


Figure 3.6: The breakdown of the NoC power consumption.

This is because our DB can replace any sleeping VC in the wakeup process to store packets. The upstream routers do not need to block packets and wait for the VCs to completely waked up. Furthermore, powering on all of the VCs in an input port at the same times can effectively deal with the bursty traffic load of real applications.

The depth of our DB also has an obvious effect on the network performance, as shown in Figure 3.5. Since we use a five-stage pipelined router, there is a notable pipeline delay and credit round-trip delay in the router. In our DB_PG with the smallest DB (DB_PG_1 in Figure 3.5), the packet transmission may be intermittently interrupted because of the credit round-trip delay between routers. In addition, as the DB only can replace one VC in a signal wakeup process, packets with different VC address may be blocked in the upstream routers, which also causes packet latency increase.

Effect on Power Consumption

Figure 3.6 shows the breakdown of the NoC power consumption across the six benchmarks and the seventh set of bars shows the average over these six benchmarks. The network power is broken down into dynamic and static power consumption.

Compared with the baseline NO_PG, the LA_PG and DS_PG can reduce an average of 73.14%, and 68.83% of the static power consumption, respectively. The static power consumption in our DB_PG with 1, 2, and 3 flits depth of the DBs is reduced by 64.11%, 58.49% and 53.63% on average, respectively. For the total power consumption, compared with the baseline NO_PG, the LA_PG and DS_PG reduce the total NoC power consumption by 59.39% and 57.05%, respectively. Our DB_PG with different DB depths reduces with 52.19%, 47.55%, and 45.14% the total power consumption, which is comparable to the LA_PG and DS_PG.

It is obvious that we can simply realize a non-blocking power gating scheme by combining the LA_PG with the DS_PG. However, in the sleeping state, the static

Table 3.2: Area of router components.

	NO_PG	DB_PG_1	DB_PG_2	DB_PG_3
input port ($\times 5$)	0.0919	0.0973	0.0992	0.1025
crossbar+VA+SA	0.0061	0.0061	0.0061	0.0061
output port ($\times 5$)	0.0065	0.0072	0.0072	0.0072
total (mm^2)	0.1045	0.1106	0.1125	0.1158

power consumption of the drowse SRAM increases with the increase of the total number of buffers in a router. As a consequence, when a NoC has a large number of buffers, the drowse SRAMs still cause significant static power consumption in the sleeping state. While, in our approach, the static power consumption in the sleeping state is only determined by the number of DBs (instead of the total number of buffers), which presents better scalability and is more suitable for a NoC with a larger number of buffers.

Effect on Area

In order to evaluate the area overhead of our DB_PG scheme compared to NO_PG, we use Synopsys Design Compiler to synthesize the routers used in our experiments under the 45nm NanGate Open Cell Library [Sil].

The area of each component is shown in Table 3.2. Compared with the baseline router used in the NO_PG, the routers used in our DB_PG_1, DB_PG_2 and DB_PG_3 cause about 5.76%, 7.65% and 10.73% area increase, respectively. Most of the area overhead in our DB_PG is contributed by the duty buffers and multiplexers in the input ports, while the area overhead of the input controller is very low. This is because the input controller is made up of simple logic circuits and a small number of registers, that do not cause large area overhead.

As our DB_PG has no influence on the crossbar, VA and SA, there is no area overhead on these components. Compared with the output ports in the baseline router (No_PG), the output ports in our DB_PG have 9.8% area overhead. However, these area overhead takes negligible percentage of the total router area because the structure of the output port controller is simple logic circuits and only contains a small number of registers.

3.6 Discussion

In this work, we proposed a novel Duty Buffer structure and DB-based power gating approach to reduce the packet latency increase caused by power gating in a NoC. As

the DB can replace any VC in the input port, the input port can power off all VCs without the need to consider which VC will be used in the future. In this way, we can keep minimal number of buffers "on duty" to reduce the packet latency increase caused by the power gating, and power off most of the VCs to reduce the static power consumption. The experimental results show that our approach outperforms the lookahead and drowsy SRAM approaches. With a small amount of additional hardware overhead, our DB-based approach can efficiently reduce the static power consumption, which is comparable with the lookahead and drowsy SRAM approaches. In later experimental results, presented in Chapter 4 and Chapter 5, we show that our DB-based power gating approach even can be more effective on reducing the power consumption under a NoC configuration with more VCs. However, being a fine-grained power gating approach, our DB-based power gating needs to separately switch the power of each input port in a router. Thus, some times packets may experience more power gating processes, which may result in an extra packet latency increase.

Chapter 4

D-bypass Power Gating Approach

Peng Wang, Sobhan Niknam, Sheng Ma, Zhiying Wang, Todor Stefanov,
"A Dynamic Bypass Approach to Realize Power Efficient Network-on-Chip"
in Proceedings of the 21st IEEE International Conference on High Performance Computing and Communications (HPCC-2019), Zhangjiajie, China, 2019.

This chapter presents our dynamic bypass (D-bypass) power gating approach, which corresponds to **Contribution 2** introduced in Section 1.4, to further reduce the packet latency increase caused by power gating. This chapter is organized as follows. Section 4.1 highlights the advantages of bypass-based power gating approaches to overcome the drawbacks of power gating, that motivate the research and development of our D-bypass power gating approach. Section 4.2 gives a summary of the main contributions in this chapter. Then, Section 4.3 introduces the Node-Router Decoupling (NoRD) power gating approach which inspires our D-bypass power gating approach. It is followed by Section 4.4, which provides an overview of the related work. Section 4.5 elaborates our D-bypass structure and introduces the D-bypass power gating approach. Section 4.6 introduces the experimental setup and presents experimental results. Finally, a concluding discussion is given in Section 4.7.

4.1 Problem Statement

Conventional power gating approaches have two negative impacts on the NoC performance: 1) Wakeup delay, there is a notable wakeup delay (6-12 clock cycles) [CZPP15] before the powered-off routers are fully recharged to the active state. This wakeup delay blocks the packet transmission between routers and causes the packet latency to significantly increase; 2) Break even time (BET), the power gating process causes additional power consumption. Normally, we use breakeven time (BET) to measure

the idle time required to compensate the power overhead due to power gating. This implies that frequent power gating or power gating in a short time may cause more power consumption or inefficient power reduction.

Many approaches try to overcome the aforementioned drawbacks of power gating in different aspects. In order to reduce the negative impact of the wakeup delay, [MKWA08] and [CZPP15] switch on the routers ahead of packet transmission. Part of or the whole wakeup delay can be hidden, but these approaches have to power on the powered-off router every time when there is a packet going through the powered-off router, which may cause frequent power gating and results in more power consumption due to the frequent power gating. On the other hand, in order to avoid non-beneficial power gating caused by BET, many works [MKI+10, ZOG+15, WNWS17] adopt fine-grained power gating on router components, such as our duty buffer based (DB-based) power gating approach in Chapter 3. Instead of waking up the whole router, these approaches individually wake up part of the router components that are required to transfer packets and keep the rest of the router components powered off. In this way, some of the router components can have longer time to stay powered off. However, these approaches are at the expense of increasing the packet latency, because packets may experience more power gating processes over a routing path. In addition to the above mentioned approaches, bypass-based approaches such as in [CP12, BHW+17, ZL18] are more attractive and comprehensive to realize power efficient NoCs. This is because, by bypassing the powered-off routes along a routing path, packets do not need to be blocked and wait for the powered-off routers to be fully charged. Thus, the packet latency increase caused by the power gating is reduced. Furthermore, without frequent interruption of the sleeping state of the powered-off routers, routers have more idle time to stay powered-off and have less power consumption overhead caused by the power gating.

In [CP12], Chen proposes one feasible and applicable bypass-based NoC power gating approach called Node-Router Decoupling (NoRD). By using a bypass latch (in the network interface (NI)) in a downstream router as a transfer station, a packet can be ejected from the NoC to the network interface without the need of storing the packet into a powered-off router buffer. Then the packet can be re-injected (forwarded) to the next router without the need of going through the crossbar in the powered-off router. By repeatedly forwarding packets, the NoRD approach allows packets to go through the powered-off routers in any hop count. Meanwhile, as packets still go through powered-off routers, the conventional credit-based flow control is available to guarantee that there is no buffer overflow. Compared with other bypass-based NoCs [BHW+17], this feature greatly simplifies the flow control. However, NoRD does not support bypass in all directions, i.e., in a powered-off router, the bypass latch in a network interface can accept packets from only one specific upstream router and

forward packets to only one specific downstream router. As a consequence, when packets try to bypass the powered-off routers, there is only one available transmission direction and packets are forced to follow detour routing paths, not the shortest routing paths, which results in an inefficient packet transmission and poor scalability.

4.2 Contributions

In order to overcome the aforementioned drawback, in this thesis, we propose a dynamic bypass (D-bypass) power gating approach. Based on a reservation mechanism to dynamically reserve a bypass latch in a powered-off router, the same bypass latch can be used by different upstream routers to dynamically build the bypass path. Thus, packets can bypass a powered-off router in any direction, which makes it possible for packets to always follow their shortest routing paths. Furthermore, as the reservation process is executed in parallel (overlaps) with the router pipeline, the timing overhead caused by the reservation process is minimized. The specific novel contributions of this work are summarized as follows:

- We extend the router structure to allow a bypass latch in a powered-off router to accept packets from any upstream router. Then, we propose a reservation mechanism to allow different upstream routes to share the same bypass latch at different times. In this way, the bypass path can be dynamically built based on the routing information of packets. Thus, when packets bypass the powered-off router, they can always follow the shortest routing paths.

- By experiments, we show that our D-bypass power gating approach can effectively reduce the power gating negative impacts on the performance and power consumption. Taking a conventional NoC without power gating as the baseline, our D-bypass power gating approach causes only 2.55% performance penalty, which is less than the 28.67% penalty in [MKWA08], 19.27% in [CP12], 7.24% in [WNWS17], and 5.69% in [ZL18]. Compared with a conventional NoC without power gating on real application workloads, our D-bypass power gating approach reduces on average 77.77% of the total power consumption of the conventional NoC, which is slightly better than 72.94%, 76.11%, 73.55% and 75.30% reductions in [MKWA08], [CP12], [WNWS17], and [ZL18], respectively. However, as a coarse-grained power gating approach, when the packet injection rate increases, most of the routers cannot be powered-off to reduce the power consumption. As a consequence, our D-bypass power gating approach is effective in reducing the power consumption only at low workloads.

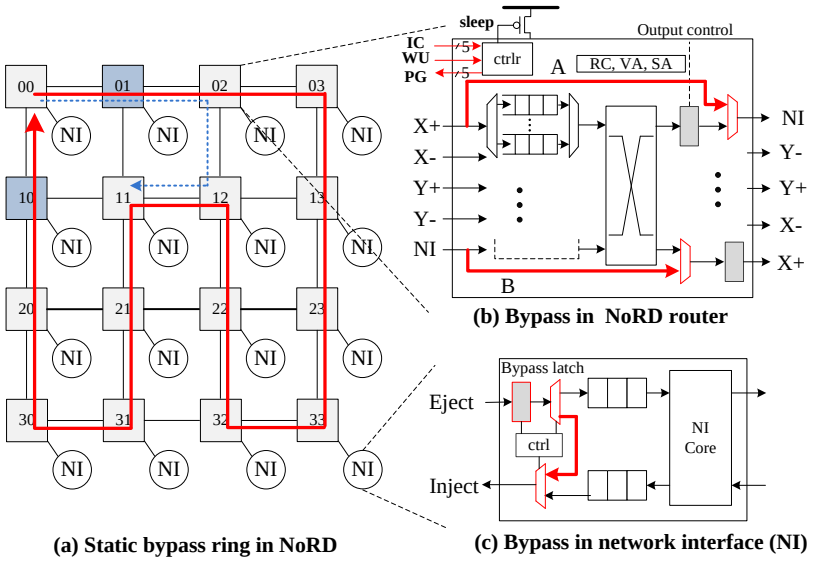


Figure 4.1: Node-Router Decoupling.

4.3 Background

In order to better understand the contributions of this chapter, in this section, we briefly introduce the bypass-based power gating approach called Node-Router Decoupling (NoRD).

NoRD [CP12] introduces a feasible way to bypass the powered-off routers to transfer packets. As shown in Figure 4.1(b), two bypass paths are added in a router. When the router is powered-off, packets directly go through bypass path A in Figure 4.1(b) and are stored in the bypass latch shown in Figure 4.1(c). Then, packets go through bypass path B in Figure 4.1(b) to be forwarded to the next router. In this way, packets can go through the powered-off router and be forwarded to the next router. Furthermore, as the packets still go through the powered-off router, the conventional credit-based flow control still works to guarantee that there is no buffer overflow. However, constrained by the router structure, NoRD does not support bypassing of the powered-off router in all directions, i.e., in a powered-off router, each network interface can accept packets from only one specific upstream router and forward packets to only one specific downstream router. As shown in Figure 4.1(a) with the tick red arrow, in NoRD, a bypass ring is statically constructed to achieve full connectivity among routers. To bypass a powered-off router, packets have to go along the static bypass ring path. For example, as shown in Figure 4.1(a), $Router00$ tries to send packets to $Router11$,

and its two downstream routers $Router01$ and $Router10$ are powered-off. $Router00$ only can send packets to bypass $Router01$. However, as $Router01$ only can forward packets along the bypass ring, packets are transferred to $Router02$ in spite of the fact that there is only one hop form $Router01$ to $Router11$. Then, after going through $Router02$ and $Router12$, the packets reach the destination $Router11$. In this example, as NoRD only can forward packet to a special direction, packets have to be transferred in a detour/longer routing path, which undermines the transmission effectiveness. Furthermore, for a large size NoC, this static bypass ring is quite long, which extremely limits the scalability of NoRD.

4.4 Related Work

A few approaches explore a bypass-based power gating NoC. Fly-over [BHW+17] switches off the power of an entire router (including output ports) and allows packets to bypass the powered-off routers, but Fly-over supports bypass in the horizontal ($X+$ $/X-$) and vertical ($Y+/Y-$) directions. When a packet needs a router to change its transmission direction ($X+$ to $Y-/Y+$, $X-$ to $Y+/Y-$, $Y+$ to $X+/X-$, and $Y-$ to $X+/X-$), this router must be woken up. Furthermore, as the output ports are powered off and all the credit information is lost, Fly-over has to utilize a complex flow control to recover the credit information when a powered-off router is powered on, which requires significant hardware overhead (a router needs 48 extra links to support this special flow control). Compared with Fly-over, Node-Router Decoupling (NoRD) [CP12] just uses the conventional credit-based flow to control the packet transmission. However, as we have introduced in Section 4.3, NoRD supports only one direction bypass in each powered-off router, which results in an inefficient packet transmission and poor scalability. Our D-bypass power gating approach also adopts the conventional credit-based flow that is similar to NoRD. However, in contrast to Fly-over [BHW+17] and NoRD [CP12], our D-bypass power gating approach is based on a reservation mechanism to dynamically build the bypass path, thus packets can bypass the powered-off routers in any direction and in any hop count. Furthermore, the reservation mechanism needs just 10 extra links for each router, which is much less than the 48 extra links in Fly-over [BHW+17]. With these aforementioned differences, our D-bypass power gating approach has better scalability than Fly-over [BHW+17] and has lower packet latency and less power consumption than NoRD [CP12].

EZ-bypass [ZL18] has similar bypass structure with our D-bypass power gating approach and allows packets to bypass the powered-off router in any direction. In EZ-bypass, each input port of a router needs one bypass latch to temporarily store packets. When a packet bypasses powered-off routers, this packet has to experience the multiple pipeline stages of routers to resolve the contention between packets that

may be in different input ports. However, in our D-bypass power gating approach, there is only one bypass latch in a router. Before using the bypass latch to go through the powered-off router, the upstream routers need to reserve this bypass latch first. In the process of the reservation, the contention between packets is resolved. In this way, when a packet is granted to use this bypass latch to go through the powered-off router, there are no other packets in the downstream powered-off router to contend with it and the router pipeline stages in the downstream powered-off router can be reduced to one stage, and some packet transmissions are accelerated. Furthermore, based on the number of reservation signals from the upstream routers, the powered-off router can detect the contention earlier. Thus, our D-bypass power gating approach can switch on the power of the powered-off router earlier than EZ-bypass.

4.5 D-bypass Approach

Fly-over [BHW+17] and NoRD [CP12] does not support bypassing in all directions. This limitation is mainly caused by the fact that the bypass latch cannot be shared by all upstream routers to forward packets. Therefore, in our D-bypass power gating approach, we first add one special hardware bypass structure in each router, which allows a bypass latch to accept packets from any of its upstream routers. Then, we propose a reservation mechanism to allow different upstream routers to use the same bypass latch at different times. By reserving the bypass latch at different times, the same bypass latch can be used to dynamically build the bypass paths from any upstream router to any downstream router. Consider the same example as described in Section 4.3, where a packet has to be sent from $Router00$ to $Router11$ and where $Router01$ and $Router10$ are powered off. Before packets are sent to the bypass latch in $Router01$, $Router00$ reserves the bypass latch in $Router01$. Next the head flit of a packet is sent to the bypass latch in $Router01$ and based on the routing information in the head flit, the bypass path is dynamically built from $Router01$ to $Router11$, see Figure 4.2(a). Then, $Router01$ can forward the packet to $Router11$. In this way, when packets go through the powered-off routers, they can always follow the shortest routing paths to their destinations.

4.5.1 Extended Router Structure

In this section, we introduce the extended router structure to support our D-bypass power gating approach. As shown in Figure 4.2(b)(c), and in contrast to NoRD [CP12], we remove the bypass latch from the NI and place it in the router, and put a NI controller (NI ctrlr) in the NI, which is used to reserve the bypass latch. In order to allow packets from all directions to skip the process of being stored in input buffers, thus

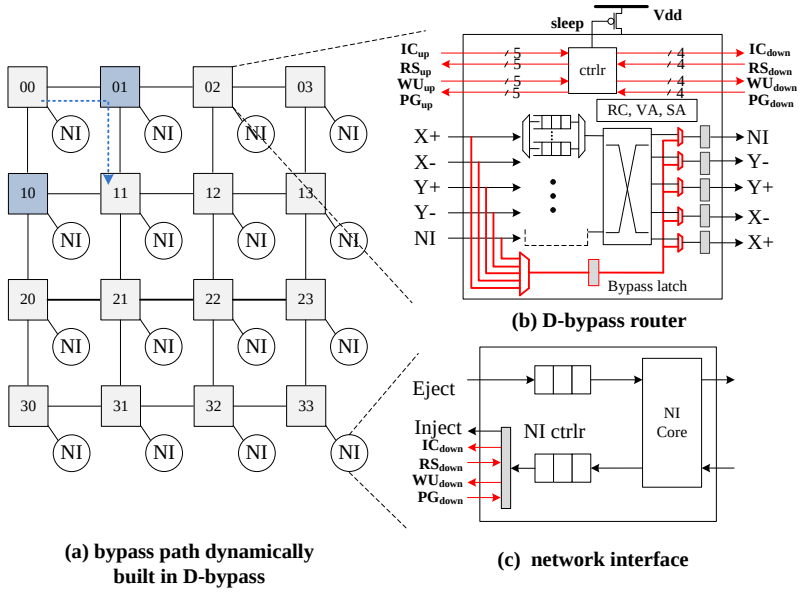


Figure 4.2: Extended router structure in D-bypass.

directly being stored in the bypass latch, we add a special hardware bypass structure to connect the input ports ($X+$, $X-$, $Y+$, $Y-$, and output Inject of the NI) with the input multiplexer. We also add five multiplexers, one in each output port, and connect the bypass latch to these output multiplexers. Based on the above mentioned extension, without the need of the crossbar, the bypass latch can accept packets from all input directions and forward packets to any of the output directions. All multiplexers are controlled by the ctrlr unit.

When multiple upstream routers or the NI need the bypass latch to forward packets, since there is only one bypass latch, as shown in Figure 4.2(b), the bypass latch cannot simultaneously forward packets coming from multiple upstream routers and the NI. However, it is possible for multiple upstream routers and the NI to share the same bypass latch by using it at different points in time. To achieve such sharing, we have devised a reservation mechanism and its hardware support. As shown in Figure 4.2(b), the handshaking control signals, i.e., the incoming signals (IC) and reservation success signals (RS), are added between routers. The indexes up and $down$ in Figure 4.2(b) and Figure 4.2(c) are used to distinguish which router (upstream and downstream) the signals are connected to. The IC signals are also used in NoRD. In an upstream router, the IC signal is asserted to inform a downstream router that a packet is coming.

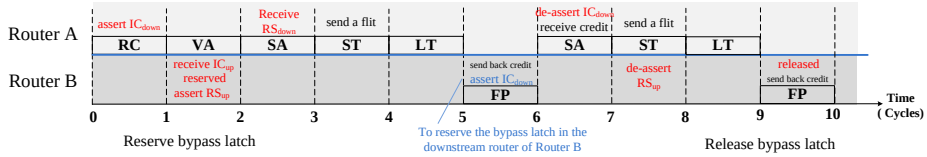


Figure 4.3: Example of the reservation process.

Besides the aforementioned IC signal functionality in NoRD, the important role of the IC signal in our D-bypass power gating approach is to reserve the bypass latch in the powered-off router. When an upstream router tries to send packets to a powered-off router, instead of asserting the WU_{down} signal, it asserts the IC_{down} signal to reserve the bypass latch in the powered-off downstream router. When the ctrlr unit in the powered-off downstream router detects this IC signal (for this downstream router, it is IC_{up}), the ctrlr unit marks the bypass latch as reserved and does not allow other upstream routers to use it. Meanwhile, the downstram router asserts the RS_{up} to inform the upstream router that it gets the right to use this bypass latch to forward packets. Once the upstream router receives this RS signal (for this upstream router, it is RS_{down}), it can send packets to that powered-off router. As our D-bypass router can forward packets to any output direction, when the packet is stored in the bypass latch, the ctrlr unit can, based on the routing information in the packet, forward the packet along its shortest routing path. In this way, according to the requirement of the packet transmission, the bypass path in a powered-off router can be dynamically built. When the upstream router finishes the packet transmission, it clears the IC_{down} signal. Then, the powered-off downstream router releases the reservation of the bypass latch and allows other upstream routers to reserve it.

Based on the aforementioned reservation mechanism, at different times, the bypass latch in a powered-off router can be used by different upstream routers and the bypass path can be dynamically built to forward packets along their shortest routing path.

4.5.2 An Example of the Reservation Process

In order to show the details of our reservation mechanism, we use the example in Figure 4.3 to illustrate the reservation process in our D-bypass power gating approach. We assume a four-stage pipeline router, which consists of route computation (RC), virtual channel allocation (VA), switch allocation (SA), and switch traversal (ST). The link traversal (LT) takes one more clock cycle. $RouterA$ tries to send packets to $RouterB$, but $RouterB$ is powered-off. The reservation process is shown in Figure 4.3.

In Cycle 0, $RouterA$ executes the RC stage for a packet and is aware that the

packet should go to $RouterB$. So, $RouterA$ asserts the IC_{down} to reserve the bypass latch in $routerB$.

In Cycle 1, $RouterA$ executes the VA stage for packets. Meanwhile, the ctrlr unit in $RouterB$ receives the IC signal (for $RouterB$, it is IC_{up}), sets the input multiplexer to select the corresponding input port, marks the bypass latch as reserved, and asserts the corresponding RS_{up} signal to acknowledge that $RouterA$ can forward packets through $RouterB$. If there are multiple IC_{up} signals simultaneously received to reserve the bypass latch, the ctrlr unit utilizes a round robin arbitration to grant the bypass latch to one of the upstream routers asserted these ICs.

In Cycle 2, $RouterA$ executes the SA stage. As the RS (for $RouterA$, it is RS_{down}) signal has arrived at this moment, $RouterA$ gets the right to forward packets to $RouterB$. The head flit of one packet is granted to go to $RouterB$. The rest of the flits are blocked at the SA stage until that $RouterA$ receives the credit from $RouterB$ or $RouterB$ is powered on.

In Cycle 3, in the ST stage of $RouterA$, the head flit of the packet is sent to the crossbar. Then, in Cycle 4, in the LT stage of $RouterA$, the head flit is sent to $RouterB$.

In Cycle 5, $RouterB$ stores the head flit in the bypass latch. As no other packets can enter $RouterB$, there is no need to execute the VA, SA, and ST stages, so the pipeline stages are reduced to one stage, i.e, Forward Packet (FP). In the FP stage, according to the routing information in the head flit, the ctrlr unit builds the bypass path for the packet, i.e., the ctrlr unit determines the output port and selects an available VC for the packet, then sets the corresponding output multiplexer to forward the head flit and the rest of flits of the packet to the downstream router of $RouterB$ (if $RouterB$ is the destination router, the packet will be directly ejected to the NI). In this way, the bypass path can be dynamically built. Furthermore, if there are multiple packets transfers through $RouterB$ at different times, different bypass paths can be dynamically built for each packet.

It should be noted that the IC_{down} signal from $RouterB$ to a downstream router of $RouterB$ is also asserted in this clock cycle. If the downstream router of $RouterB$ is also powered off, the head flit is blocked at the FP stage until $RouterB$ gets the RS_{down} signal from its downstream router. In this way, the packet can bypass multiple powered-off routers. When one flit leaves $RouterB$, one credit is sent to $RouterA$.

In Cycle 6, $RouterA$ gets the credit to send another flit. In our example, the packet has two flits, so, the packet transmission is finished in this clock cycle and the IC_{down} signal is de-asserted.

In Cycle 7, $RouterA$ executes the ST stage for the last flit. $RouterB$ is aware that the IC coming from $RouterA$ signal (it is IC_{up} for $RouterB$) is de-asserted and de-asserts the corresponding RS_{up} signal.

After experiencing the LT stage in Cycle 8, the last flit arrives in $RouterB$. In Cycle 9, the last flit is forwarded to the downstream router of $RouterB$. The ctrlr unit in $RouterB$ releases the reservation of the bypass latch and allows other upstream routers to reserve the bypass latch.

Based on the reservation process exemplified above, the bypass latch in the powered-off routers can be used by all upstream routers and the NI to forward packets to any direction at different times. By reserving multiple bypass latches in different routers, packets can bypass multiple powered-off routers along their routing path. Furthermore, as shown in this example, the reservation process is executed in parallel (overlaps) with the router pipeline. Thus, the timing overhead of the reservation process is minimized.

4.5.3 Power Gating Conditions

In this section, we introduce the conditions which drive the ctrlr unit in Figure 4.2(b) to control the power supply of a router.

Powering off a router

When there is no packet left in a router, and the ICs and WUs signals from all its upstream routers are de-asserted, the router goes into the idle state and the PG signals are asserted to all upstream routers, but at this moment, the power supply is not cut off yet. After waiting T_{idle_detect} clock cycles, the ctrlr unit asserts the sleep signal (Figure 4.2(b)) and cuts off the power supply. If there is any IC or WU signals asserted during T_{idle_detect}, the ctrlr unit immediately de-asserts the PG signals. By waiting T_{idle_detect} clock cycles to cut off the power supply, we can avoid non-beneficial power gating caused by short idle time of routers, which causes frequent power gating and additional power consumption.

Powering on a router

To keep good NoC performance, the routers should be powered on at the right moment to deal with high traffic workloads. In our D-bypass power gating approach, we use two metrics to determine when a router should be powered on.

- N_{IC} is the number of IC_{up}s simultaneously received by a powered-off router. In a powered-off router, when N_{IC} exceeds a threshold th_{IC}, the powered-off router is woken up. In this situation, the condition of powering on a router is triggered by the IC_{up} signals. As an IC_{up} signal is sent ahead of a packet transmission, part of the wakeup delay is hidden. Furthermore, during the time of

charging the powered-off router, one of the upstream routers can forward packets through the powered-off router. Thus, the packet latency increase caused by the wakeup delay is reduced.

- N_{IVC} is the number of input VCs, in one upstream router, contending for the same downstream router to forward packets. N_{IVC} indicates the workload of an upstream router. As there is only one bypass latch in a router, our D-bypass power gating approach has significant credit round-trip delay, which blocks a packet transmission to wait for credits. Powering on the downstream routers can reduce this impact. In an upstream router, when N_{IVC} to a powered-off downstream router exceeds a threshold th_{IVC}, the corresponding WU signal is asserted to wakeup the downstream router. During the time of waiting the downstream router to fully charge, the upstream router can forward packets through the bypass latch of the downstream router, so the impact of the wakeup delay is also reduced.

It is clear that there is a risk of deadlock when multiple upstream routers need the same powered-off router to transfer packets, but the powered-off router may be continuously occupied by a router and the other routers cannot get a chance to send packets. In order to a avoid this deadlock problem, we set the threshold $th_{IC} = 1$. On the other hand, in order to avoid performance penalties as much as possible, we aggressively set the threshold $th_{IVC} = 1$, which implies that when multiple packets are sent simultaneously to the same powered-off router, the powered-off router should be powered on. The low th_{IC} and th_{IVC} may tend to trigger more often the condition of powering on a router, which may cause frequent power gating on a router. However, considering the low average injection rate in real applications, there is still high probability of transferring packets through powered-off routers without frequently triggering the condition for powering on a router.

4.6 Experimental Results

In order to evaluate our approach in terms of performance and power consumption, we have implemented our approach using a full-system simulator called Agate [CZPP16]. Agate is based on the widely used full-system simulator GEM5 [BBB$^+$11], and Agate supports the simulation of the key items in NoC power gating techniques. The NoC model and power model used in Agate are based on Garnet [AKPJ09] and Dsent [SCK$^+$12], respectively. The key parameters used in our experiments are shown in Table 4.1. We choose a four-stage pipeline router. The number of VCs and the buffer size of control VCs and data VCs are set based on the related works [CZPP15] and [CP12]. For simplicity, we use a X-Y deterministic routing algorithm in our D-bypass power

Table 4.1: Parameters.

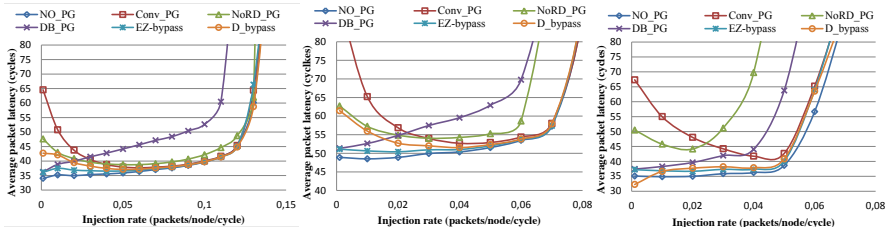
Network topology	8×8 mesh
Router	4-stage pipeline
Virtual channel	2 VCs/VN, 3 VNs
Input buffer size	1-flit/ ctrl VC, 5-flit / data VC
Routing algorithm	X-Y, Adaptive
Link bandwidth	128 bits/cycle
Wakeup delay	8 clock cycles
Break even time	10 clock cycles
Private I/D L1$	32 KB
Shared L2 per bank	256 KB
Cache block size	16 Bytes
Coherence protocol	Two-level MESI
Memory controllers	4, located one at each corner

gating approach and other related approaches, but for the NoRD approach, we have implemented the special adaptive routing algorithm required by NoRD [CP12] to fairly compare with the NoRD approach. The value of the wakeup delay and break even time (BET) are according to the related works [CZPP15] and [CP12]. As there are additional components added in our D-bypass router and the routers in related approaches, in order to evaluate the power consumption of these components, we use Dsent [SCK+12] to estimate the power consumption of the major components, such as the buffers and multiplexers, to make the experimental results more accurate.

For comparison purpose, we have implemented the following power gating approaches: (1) NO_PG: the baseline NoC without power gating; (2) Conv_PG: conventional power-gating NoC, which is deeply optimized by sending WU (Look ahead [MKWA08]) and de-asserting PG signals [CZPP16] in advance, thus 6 clock cycles of the wakeup delay are hidden in our experiments; (3) NoRD_PG [CP12]: the power gating NoC with the NoRD approach; (4) DB_PG [WNWS17]: our DB-based power gating approach introduced in Chapter 3. In each input port of a router, a one-flit size duty buffer is added to implement the DB-based power gating approach; (5) EZ_bypass [ZL18]: the power gating NoC with the EZ-bypass approach in which the bypass structure is similar to our approach; (6) D-bypass: the NoC with our D-bypass power gating approach introduced in Section 4.5.

4.6.1 Evaluation on Synthetic Workloads

In order to explore the behavior of our D-bypass power gating approach under a wider range of packet injection rates, in this section, we evaluate the performance of our D-bypass power gating approach under synthetic traffic patterns. We select three syn-



(a) Uniform random (b) Bit-complement (c) Transpose

Figure 4.4: Packet latency across different injection rates.

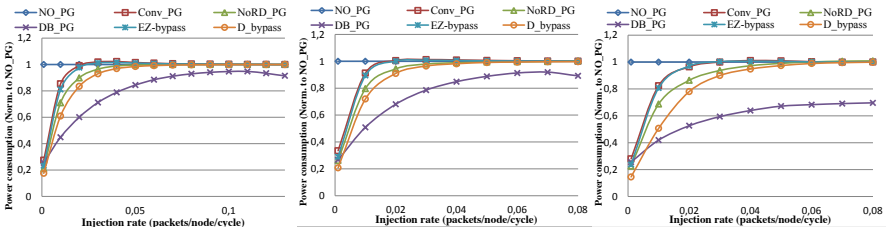
thetic traffic patterns: 1) Uniform random: packets' destinations are randomly selected; 2) Bit-complement: packets from source router (x, y) are sent to destination router (N-x, N-y), N is the number of routers in the X and Y dimensions of a NoC; 3) Transpose: packets from source router (x, y) are sent to destination router (y, x);

Effect on NoC Network Latency

As shown in Figure 4.4(a) and Figure 4.4(b), when the injection rate is around 0.001 packets/node/cycle, our D-bypass has higher average packet latency than DB_PG and EZ_PG, but lower than Conv_PG and NoRD_PG. This is because in our D-bypass approach, multiple packets cannot simultaneously bypass the same powered-off routers at the same time, and some packets are blocked due to power gating. However, compared with Conv_PG, there are significant number of packets that can bypass the powered-off routers. On the other hand, when the packet bypasses the powered-off router, the powered-off router pipeline stages are reduced to one stage and some packets' transmissions can be accelerated. Thus, in Figure 4.4(c), our D-bypass has the lowest packet latency among all the approaches.

With the injection rate increasing up to the saturation injection rate (around 0.13 packets/node/cycle in uniform random, 0.07 packets/node/cycle in bit-complement, 0.05 packets/node/cycle in transpose), the curve of the average packet latency in our D-bypass approach slowly drops, and it is lower than the curve of Conv_PG and NoRD_PG, and gradually gets close to the curve of NO_PG. This indicates that our D-bypass approach can more efficiently deal with high bursty traffic workloads than Conv_PG and NoRD_PG, which meets requirements of real applications where traffic workloads are bursty.

The saturation injection rate is also an important parameter to evaluate the NoC performance. A NoC with higher saturation injection rate can achieve higher throughput. As shown in Figure 4.4, our D-bypass approach has the same saturation injection rate as the baseline NO_PG, but NoRD_PG and DB_PG have lower saturation in-



(a) Uniform random (b) Bit-complement (c) Transpose

Figure 4.5: Power consumption across different injection rates.

jection rate. This is because, at the saturation injection rate, all routers are powered on and our D-bypass approach works the same as NO_PG. However, the routers in NoRD_PG are not as efficient as the routers in NO_PG. This is because NoRD_PG needs VCs to support its special adaptive routing along the bypass ring. As a consequence, NoRD_PG cannot fully utilize VCs to achieve the same saturation injection rate as NO_PG. Therefore, compared with the bypass-based power gating scheme NoRD_PG, our D-bypass approach can achieve higher throughput.

Effect on NoC Power Consumption

As shown in Figure 4.5, when the packet injection rate is 0.001 packets/node/cycle, our D-bypass approach has the lowest power consumption. This is because, at such low injection rate, our D-bypass approach can transfer packets through the powered-off routers without the need of powering them on. Thus, our D-bypass approach can reduce more the power consumption compared to Conv_PG. Furthermore, compared with DB_PG and EZ-bypass, we need less hardware to implement our D-bypass approach. It means that our D-bypass approach causes less extra power consumption. Thus, when most of the routers are powered-off in a NoC, our D-bypass approach consumes less power than DB_PG and EZ-bypass. In addition, compared with NoRD_PG, our D-bypass transfers packets through the powered-off routers along the shortest routing path, which is more efficient in transferring packets and helpful to reduce the power consumption.

However, when the injection rate increases, the power consumption in our D-bypass approach increases and reaches the power consumption of NO_PG, which is the same for Conv_PG, NoRD_PG, and EZ_PG. This is a common drawback for all coarse-grained power gating approaches, because power gating is applied on the granularity of a router. When the injection rate increases, more routers become busy and cannot be powered off. As a consequence, compared with DB_PG, which is a fine-grained power gating approach and is effective in reducing the power consumption

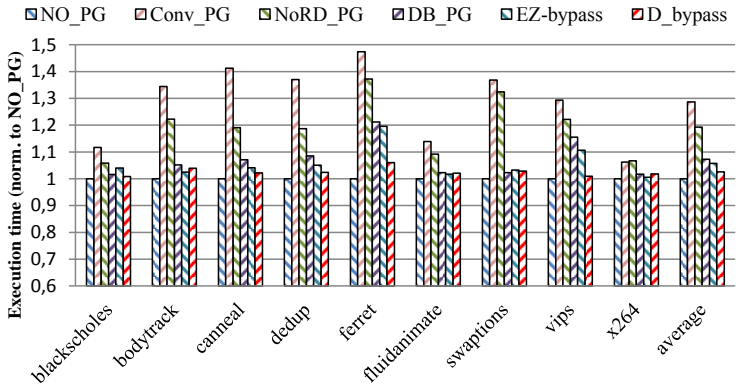


Figure 4.6: Execution time.

under a wider range of packet injection rates, our D-bypass approach can efficiently reduce the power consumption only at low packet injection rates.

4.6.2 Evaluation on Real Application Workloads

In this section, we use real application workloads to compare the approaches in terms of the application performance, the NoC average packet latency, and the NoC power consumption. To do so, we use nine applications from the Parsec [BKSL08] benchmark suite.

Effect on Application Performance

Figure 4.6 shows the execution time of the nine applications, which is normalized to the baseline NO_PG, and the tenth set of bars in Figure 4.6 gives the average results over these nine applications. Our D-bypass approach causes less performance penalty (execution time increase) than the related approaches. Compared with the baseline NO_PG, our D-bypass causes an average of 2.55% performance penalty, which is less than the 28.67% performance penalty in Conv_PG, 19.27% in NoRD_PG, 7.24% in DB_PG, and 5.69% in EZ_bypass. In the ferret benchmark, our D-bypass has its largest performance penalty of 6.03%, and Conv_PG, NoRD_PG, DB_PG, and EZ_bypass have also their largest performance penalty of 47.39%, 37.18%, 21.22%, and 19.51%, respectively.

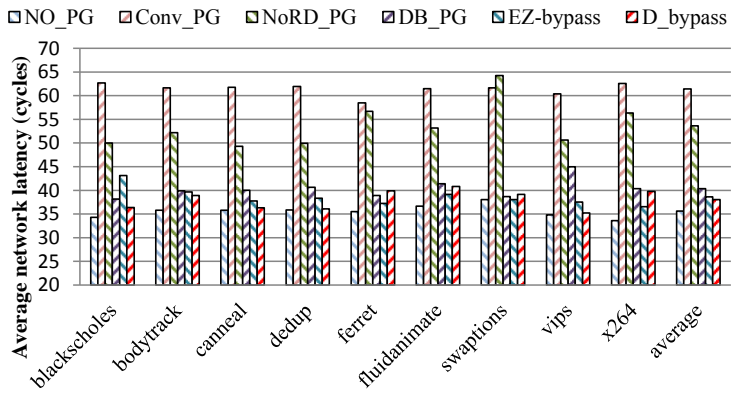


Figure 4.7: Average packet latency.

Effect on NoC Network Latency

Figure 4.7 shows the average network latency across the nine applications. Our D-bypass approach can efficiently reduce the network latency increase caused by power gating. Compared with NO_PG across the applications, the average network latency in our D-bypass approach slightly increases, but is much lower than Conv_PG and NoRD_PG. This is because our D-bypass approach can dynamically build the bypass path and allow packets to bypass the powered-off router in all directions. Thus, packets can go along the shortest routing paths to bypass the powered-off routers, and are not blocked due to the power gating processes.

In most of the applications, our D-bypass approach has slightly lower average network latency than DB_PG and EZ_bypass. This is because DB_PG is a fine-grained power gating approach and causes more power gating processes. Compared with EZ-bypass, our D_bypass is based on a reservation mechanism which can power on the powered-off router earlier when multiple upstream routers need the same powered-off router to forwards packets. However, in the benchmarks ferret, fluidanimate, swaptions, and x264, our D-bypass approach has slightly higher average network latency than EZ_bypass, because each input port in EZ_bypass has a bypass latch to hold one flit of a packet, whereas in our D-bypass approach, all input ports in a router have to share one bypass latch to forward packets, which may result in more contention and blocking of some packet transmissions. However, in our D-bypass, as only one packet is allowed to go through a powered-off router at a time, the router pipeline stages can be reduced to one stage when packets bypass the powered-off routers. Thus, some packet transmissions are accelerated and our D-bypass approach has lower application execution time than EZ_bypass in ferret and swaptions, in spite of the fact that

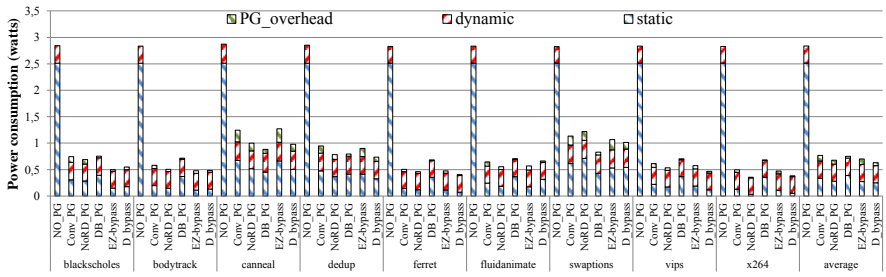


Figure 4.8: Breakdown of the NoC power consumption.

our D-bypass approach has slightly higher average packet latency than EZ_bypass.

Effect on NoC Power Consumption

Figure 4.8 shows the breakdown of the NoC power consumption across the nine applications and the tenth set of bars shows the average over these nine applications. The NoC power consumption is broken down into three parts: the extra power consumption caused by the power gating (PG_overhead) and the dynamic/static power consumption of routers (dynamic/static).

As can be seen in Figure 4.8, our D-bypass approach reduces slightly more the power consumption than the related approaches. Compared with NO_PG, our D-bypass reduces on average 77.77% of the total NoC power consumption, which is slightly better than 72.94% in Conv_PG, 76.11% in NoRD_PG, 73.55% in DB_PG, and 75.30% in EZ_bypass. This is because, for real application workloads, the traffic is busty for very short periods of time, thus the average packet injection rate is low for a long period of time. Therefore, all of these power gating approaches can power off routers for a long time to reduce the static power consumption. In addition, our D-bypass approach can transfer packets through the powered-off routers without waking them up. Thus, our D-bypass can power off the routers for even longer time and it can reduce more the router static power consumption and PG_overhead compared to Conv_PG. Even though NoRD_PG is also a bypass-based power gating approach, it does not support bypass in all directions and forces packets to go along the bypass ring. Packets have to go through more routers, which may cause more power gating processes. As a consequence, NoRD_PG consumes slightly more router static power and PG_overhead than our D_bypass. Furthermore, in order to transfer packets through the powered-off routers or the powered-off input ports, our D_bypass, EZ-bypass, and DB_PG need to always keep some components powered on, that always consume static power. However, compared with DB_PG and EZ-bypass, our D_bypass needs

to keep fewer components always powered-on. Therefore, our D_bypass is more efficient to reduce the static power consumption of the routers.

4.7 Discussion

In this chapter, we propose a dynamic bypass (D-bypass) power gating approach to allow packets to bypass powered-off routers in any hop count and in any direction. Based on a reservation mechanism, all the upstream routers can share the same bypass latch to dynamically build the bypass path for different packets. In this way, packets can be transferred along their shortest routing paths. With small hardware overhead, our D-bypass approach can efficiently reduce the power consumption and has less performance penalty.

Even though our D-bypass power gating approach allows packets to bypass the powered-off routers in any direction, the efficiency of the bypass path is limited by the single bypass latch in a router. The packets maybe frequently blocked to wait for the free bypass latch. As a result, in some applications, there is still significant packet latency increase in our D-bypass power gating approach. Furthermore, like most of the course-grained power gating approaches, our D-bypass power gating approach cannot fully utilize the idle time of each component in a router. When the traffic workload is high, most of the routers in a NoC become busy and cannot be powered off to reduce the static power consumption. As a consequence, our D-bypass power gating approach is effective in reducing the power consumption only at low traffic workloads.

Chapter 5

EVC-based Power Gating Approach

Peng Wang, Sobhan Niknam, Sheng Ma, Zhiying Wang, Todor Stefanov,
"EVC-based Power Gating Approach to Achieve Low-power and High Performance NoC"
in Proceedings of the Euromicro Conference on Digital System Design (DSD), Chalkidiki, Greece, 2019.

In this chapter, we present our EVC-based power gating approach, which corresponds to **Contribution 3** introduced in Section 1.4. By using our EVC-based power gating approach, not only the packet latency increase caused by power gating can be further reduced, but also the power consumption can be reduced at high traffic workloads. The remainder of this chapter is organized as follows. Section 5.1 further elaborates on the problem of the low efficiency in the bypass path mentioned in Section 4.7. Section 5.2 summarizes the main contributions in this chapter. Section 5.3 briefly introduces the express virtual channel scheme (EVC). Then, we provide an overview of the related bypass-based power gating approaches in Section 5.4. It is followed by Section 5.5, which elaborates our EVC-based power gating approach. Section 5.6 introduces the experimental setup and presents experimental results. Finally, Section 5.7 concludes this chapter.

5.1 Problem Statement

As we have introduced in Chapter 4, bypass-based power gating approaches are more comprehensive to reduce the power consumption and the packet latency increase caused by the power gating. However, in many bypass-based approaches, there are only a few bypass latches to temporarily store packets on a bypass path. Before bypassing powered-off routers, packets have to be blocked until there are available bypass

latches, which significantly undermines the efficiency of the bypass paths. As a result, in most of the bypass-based approaches, the bypass paths are not very efficient to transfer packets. For example, the bypass paths in [CP12, FTKH16, ZL18] and in our D-bypass power gating approach [WNM+19a] (Chapter 4) cannot continuously transmit packets via bypass powered-off routers. Even though the approach in [BHW+17] can continuously transmit packets via bypass powered-off routers, it has significant timing overhead and hardware overhead to recover the routing information that is lost in the powered-off routers. As a consequence, all aforementioned bypass-based approaches still have significant packet latency increase caused by the power gating. Furthermore, like most of the coarse-grained power gating approaches, these bypass-based power gating approaches cannot fully utilize the idle time of each component in a router to reduce the power consumption. When the traffic workload becomes high, most of the routers become busy and cannot be powered off to reduce the power consumption of a NoC. As a consequence, these bypass-based power gating approaches are effective in reducing the power consumption only at low traffic workloads.

5.2 Contributions

In order to overcome the aforementioned drawbacks, we propose an express virtual channel based (EVC-based) power gating approach. In our approach, multiple virtual bypass paths are pre-defined at design time. Packets can take these virtual bypass paths to bypass intermediate routers that can be powered-on or powered-off. When a packet takes a virtual bypass path, the sink router of the virtual bypass path is powered-on. There is sufficient amount of buffers in sink routers to hold packets. Thus, packets can continuously go through a virtual bypass path. Furthermore, compared with other bypass-based approaches [CP12, FTKH16, BHW+17, ZL18, WNM+19a] in which the packets can only bypass powered-off routers, in our EVC-based approach, packets can bypass powered-on routers as well. Thus, even at a high traffic workload, our approach also can reduce the power consumption by reducing the dynamic power. The specific novel contributions of this chapter are the following:

- We propose a specific distribution of virtual bypass paths on a NoC, which allows more packets to take the virtual bypass paths compared to the conventional EVC scheme [KPKJ07]. More importantly, we extend the router structure to guarantee that a virtual bypass path cannot be blocked by powered-off routers. Thus, by allowing packets to go through the virtual bypass paths without blocking, these packets can avoid experiencing the wake up process at the intermediate routers. Furthermore, based on our extended router structure, a certain transmission ability of the powered-off/being charged routers is kept to transfer packets going through the normal paths. In this way, the packet latency

increase caused by the power gating is further reduced. We also propose an effective power gating scheme to control the power switching of routers. Finally, we propose an approach to freeze virtual bypass paths in order to resolve starvation, which is a common issue in EVC-based NoCs [KPKJ07].

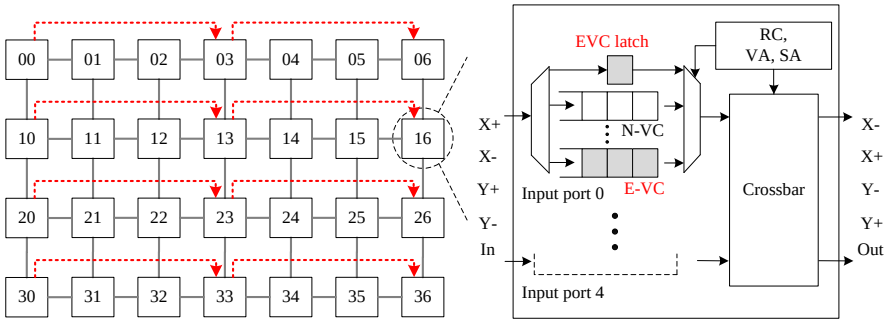
- By experiments, we show that our EVC-based power gating approach can effectively reduce the power gating negative impacts on the performance and power consumption. Thanks to the efficient virtual bypass paths, our EVC-based power gating approach can achieve lower network latency than the related approaches [MKWA08, WNWS17, ZL18, WNM$^+$19a], even lower than a NoC without power gating. However, our EVC-based power gating approach causes unbalanced resource allocation, which results in slight performance penalty in the total execution time of real applications. Compared with a NoC without power gating, our EVC-based power gating approach achieves notable reduction of the power consumption, which is comparable with the related approaches [MKWA08, WNWS17, ZL18, WNM$^+$19a]. Furthermore, by allowing packets to bypass powered-on routers as well, our approach consumes less power than the related approaches [MKWA08, WNWS17, ZL18, WNM$^+$19a] under high traffic workloads.

5.3 Background

In order to better understand the novel contributions of this chapter, in this section, we introduce the conventional EVC [KPKJ07] scheme that allows packets to bypass intermediate routers along a virtual bypass path.

The express virtual channel (EVC) scheme [KPKJ07] is a classical virtual bypass technique. As shown in Figure 5.1(a), the virtual bypass paths (red dashed lines) are pre-defined on a NoC topology. These virtual bypass paths are implemented without the need for extra physical wires, but based on the virtual channels in a router to share the existing wires. The basic EVC router architecture is shown in Figure 5.1(b). Compared with the conventional router in Figure 2.5, in each input port, one EVC latch is added, and the virtual channels are partitioned into two groups, normal virtual channels (N-VCs) and express virtual channel (E-VCs). N-VCs are used to accept packets from neighbor upstream routers. E-VCs in the sink routers of the virtual bypass paths are used to accept packets from the source routers of the virtual bypass paths.

By allocating E-VCs to packets, the source router in a virtual bypass path can determine if the packet takes the virtual bypass path. For example, in Figure 5.1, assume that a packet has to be sent form $Router00$ to $Router04$. Based on the transmission distance, $Router00$ is aware that by taking the virtual bypass path from $Router00$ to $Router03$, the packet will have lower latency. So, $Router00$ treats this packet as



(a) A NoC with virutal bypass paths

(b) EVC router

Figure 5.1: Express virtual channel.

an E-packet (the packet going through a virtual bypass path) and allocates one E-VC in $Router03$ for this packet. When the packet reaches $Router01$ and $Router02$, this packet is temporarily held in the EVC latch with the highest priority. Then, this packet is directly sent without experiencing the pipeline stages in $Router01$ and $Router02$, and reaches $Router03$. When this packet reaches $Router03$, this packet is stored at the allocated E-VC. $Router03$ knows this packet should go to the normal path to its destination $Router04$, and treats this packet as a N-packet (the packet going through the normal path between routers) and allocates a N-VC in $Router04$ for this packet. After experiencing the pipeline stages in $Router03$, this packet is sent to its destination $Router04$.

By taking virtual bypass paths, E-packets do not need to experience the pipeline stages in the intermediate routers. This implies that most of the components in the intermediate routers are unnecessary to transfer E-packets. This characteristic is attractive and promising for realizing a power gating NoC to allow packets to bypass powered-off routers. We exploit effectively this characteristic in this thesis to realize our EVC-based power gating approach.

5.4 Related Work

Several approaches propose a bypass-based power gating NoC. In NoRD [CP12], a virtual ring is pre-defined on a NoC, which works as a backup NoC. When a packet is blocked by a powered-off router, it can go along this virtual ring to bypass the powered-off router. However, limited by the low efficiency and poor scalability of

the virtual ring, packets may be detoured for a long distance to their destinations. As a consequence, NoRD has significant packet latency increase and is not suitable for large NoCs. In contrast, in our approach, we pre-define multiple virtual bypass paths, which are separately distributed on the whole NoC. Packets go along their shortest routing path and separately take these virtual bypass paths to bypass the powered-off routers. Thus, our EVC-based power gating approach has lower packet latency and better scalability.

In Turn-on on Turn (TooT) [FTKH16], a bypass path is pre-defined in the horizontal ($X+/X-$) and vertical ($Y+/Y-$) directions. Thus, packets can bypass a powered-off router if the packets do not need the powered-off router to change the transmission direction or to eject from the NoC. So, TooT does not need to frequently power on the powered-off routers and can more efficiently reduce the static power consumption. However, limited by a few bypass latches on a bypass path, packets have to be blocked until there are available bypass latches. As a consequence, the bypass paths are inefficient to transmit packets in order to bypass the powered-off routers and TooT still has significant packet latency increase. In contrast, in our EVC-based power gating approach, when a packet goes through a virtual bypass path, the sink router is powered on. Thus, there are more buffers to be used to hold packets and packets can continuously go through the virtual bypass path. As a consequence, bypass paths in our approach are more efficient than TooT in terms of transmitting packets, therefore the packet latency increase is reduced.

Similar to TooT, Fly-over [BHW$^+$17] also allows packets to bypass powered-off routers in the horizontal ($X+/X-$) and vertical ($Y+/Y-$) directions but Fly-over does not need to block packets to wait for available bypass latches between the neighbor routers. This is because Fly-over dynamically realizes the credit-based flow control [DT04] between the source router and the sink router on a bypass path to guarantee that there is no buffer overflow. When a source router transmits packets to bypass the intermediate powered-off routers, the sink router must be powered-on. Thus, there is sufficient amount of buffers available to be used to hold packets and Fly-over can continually transmit packets. However, Fly-over has to utilize a complex mechanism to realize the credit-based flow control between the source router and the sink router, which causes significant timing and hardware overhead. In contrast, in our EVC-based power gating approach, the virtual bypass paths are (static) pre-defined. Thus, our EVC-based approach has no such extra timing overhead.

In contrast to TooT and Fly-over, the bypass path in EZ-bypass [ZL18] is dynamically built to allow packets to bypass the powered-off routers in any direction. Thus, a packet can bypass a powered-off router, even when this router is required to change the transmission direction. As a result, EZ-bypass is more flexible and can be more efficient to reduce the power consumption. However, in EZ-bypass, when a

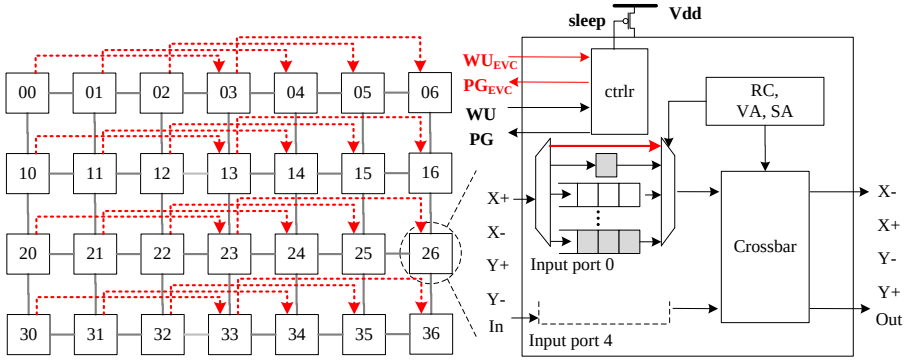
packet bypasses powered-off routers, this packet has to stay in the powered-off routers for multiple clock cycles to experience the pipeline stages of routers. As a consequence, the bypass latch is occupied by one packet for a long time and the bypass path is frequently blocked, which undermines the efficiency of the bypass path. In contrast, in our EVC-based power gating approach, when a packet bypasses intermediate routers, this packet does not experience the router pipeline stages. Thus, our EVC-based power gating approach can achieve lower packet latency than EZ-bypass. Furthermore, compared with NoRD [CP12], TooT [FTKH16], Fly-over [BHW$^+$17], and EZ-bypass [ZL18] in which the packets can bypass only powered-off routers, in our EVC-based approach, packets can bypass powered-on routers as well. Thus, even at a high workload traffic, our approach also can reduce the power consumption by reducing the dynamic power.

5.5 Our EVC-based Power Gating

In this section, we present our novel power gating approach which uses and extends the conventional EVC scheme to allow packets to bypass powered-off routers. First, in Section 5.5.1, we propose a distribution of the virtual bypass paths to allow more packets to take the virtual bypass paths. Then, in Section 5.5.2, we extend the EVC router structure to guarantee that the virtual bypass paths are not blocked by powered-off routers. Thus, packets can always take a virtual bypass path to bypass the intermediate routers that may be powered-off. Furthermore, based on our extended router structure, a powered-off router has certain transmission ability to transfer also packets that take the normal paths. So, even though some packets do not take a virtual bypass path, they can avoid as much as possible to be blocked by powered-off routers. In Section 5.5.3, we describe our power gating scheme used in our EVC-based power gating approach, and in Section 5.5.4, we use an example to illustrate our power gating scheme. Finally, in Section 5.5.5, we propose an approach to resolve the starvation which may occur when using our EVC-based power gating approach.

5.5.1 Distribution of Virtual Bypass Paths

In the EVC scheme, packets can bypass the intermediate routers only when they take virtual bypass paths. So, in order to allow packets to bypass the intermediate routers that may be powered-off, we have to allow more packets to take the virtual bypass paths. To achieve this goal, in each direction, we pre-define one virtual bypass path between each two routers with three hops. As shown in Figure 5.2(a), in the $X+$ direction, we set one virtual bypass path between $Router00$ and $Router03$, $Router01$ and $Router04$ and so on. The virtual bypass paths in the $X-$, $Y+$, and $Y-$ directions have similar settings, but are not shown in Figure 5.2(a) for the sake of clarity.



(a) Distribution of virtual bypass path in the X+ direction

(b) Extended EVC router

Figure 5.2: Extended EVC-based power gating approach.

Compared with the conventional distribution of the virtual bypass paths [KPKJ07] in Figure 5.1(a), the packets in Figure 5.2(a) have higher probability to take a virtual bypass path. For example, in a 8×8 2D mesh NoC, there are in total 4032 routing paths from one source router to a destination router. Based on the distribution of the virtual bypass paths in Figure 5.1(a), the average number of virtual bypass paths on a routing path is 0.56, while, based on our distribution of the virtual bypass paths in Figure 5.2(a), the average number of virtual bypass paths on a routing path is 1.13.

In our EVC-based power gating approach, routers always try to send packets to a virtual bypass path. Only when there is no virtual bypass path available, the packets are sent along the normal path between routers.

5.5.2 Extended Router Structure

We have extended the basic EVC router in Figure 5.1(b) to enable and support our novel power gating scheme. As shown in Figure 5.2(b), one power control (ctrlr) unit is added in the router. Handshaking control signals WU (wakeup) and PG (power gating) are added between routers. Compared with the conventional power gating, introduced in Section 2.3, extra handshaking control signals, WU_{EVC} and PG_{EVC} are added between the source router and the sink router for a virtual bypass path. In each input port, one direct link is added (e.g., the red arrow in Input port 0, shown in Figure 5.2(b)). These direct links are used to build the bypasses in the direction from $X+$ to $X-$, $X-$ to $X+$, $Y+$ to $Y-$, and $Y+$ to $Y-$. **To avoid N-packets to be blocked by the powered-off routers, in our EVC based power gating approach, the EVC latch is also used to hold N-packets when the router is powered-off or**

being charged. When a router is powered off and the EVC latch in an input port is used to hold a N-packet, a bypass path is setup by using the direct link in the input port and the crossbar for E-packets. For example, when a router is powered-off and the EVC latch in the $X+$ input port holds a N-packet, a bypass path from $X+$ to $X-$ is built by using the direct link in the $X+$ input port and the crossbar in this router for E-packets. Then, if an E-packet is coming, it directly goes through this router by taking this directly built bypass in the router. In this way, we guarantee that the virtual bypass path always works for E-packets even when the EVC-latch is occupied by a N-packet. Furthermore, the powered-off router has certain transmission ability to transfer N-packets through the normal paths. In this way, the N-packets have less probability to be blocked by powered-off routers.

To transfer N-packets though a powered-off router, the routing computation unit, the EVC latches, the virtual channel allocator unit, the switch allocator unit, and the crossbar are always powered on to execute the router pipeline stages. The power control (ctrlr) unit only cuts off the power supply of VCs. In this way, even at the powered-off state, the router still keeps a certain ability to transfer packets. Thus, the packets going through the normal paths have less probability to be blocked by the powered-off routers. Furthermore, as these units consume much less power than VCs [WNWS17, ZOG+15], our EVC-based power gating approach still can efficiently reduce the static power consumption by powering off the idle VCs.

5.5.3 Power Gating Scheme

In this section, we introduce the conditions which drive our ctrlr unit in Figure 5.2(b) to control the power supply of a router.

Powering off a router

When there are no packets left in EVC latches, N-VCs, E-VCs, or the crossbar in a router, and the WU and WU_{EVC} signals from all its upstream routers are de-asserted, the router goes into the idle state, the PG_{EVC} and PG signals are asserted to all upstream routers, but at this moment, the power supply is not cut off yet. After waiting T_{idle_detect} clock cycles, the ctrlr unit asserts the sleep signal (Figure 5.2(b)) and cuts off the power supply. If there is any WU or WU_{EVC} signal asserted during T_{idle_detect}, the ctrlr unit immediately de-asserts the PG_{EVC} and PG signals. By delaying T_{idle_detect} clock cycles to cut off the power supply, we can avoid non-beneficial power gating caused by short idle time of routers, which causes frequent power gating and additional power consumption.

Powering on a router

The process of powering on a router in our EVC-based power gating approach is an extension of the wakeup process shown in Figure 2.9 explained in Section 2.3. If a source router determines that a packet should take the virtual bypass path to the sink router, this source router asserts the corresponding WU_{EVC} to power on the sink router. If a router determines that a packet should take the normal path to the downstream router, this router asserts WU to power on the downstream router. Once the powered-off router receives the WU_{EVC} signal or the WU signal, the powered-off router starts to charge and goes into the wakeup state. After $T_{wakeup} - MARGIN_{EVC}$ clock cycles, the router de-asserts PG_{EVC} and the source router can send packets to this router using the virtual bypass path. After $T_{wakeup} - MARGIN$ clock cycles, the router de-asserts PG and the upstream router can send packets to this router using the normal path. By setting properly $MARGIN_{EVC}$ and $MARGIN$, a router can send packets before the powered-off router is fully charged, but it is guaranteed that when a packet reaches the powered-off router, this router is just fully charged. In this way, we can hide part of the wakeup delay and optimize the power gating process. It should be noted that $MARGIN_{EVC}$ is larger than $MARGIN$. This is because by taking virtual bypass paths, E-packets have more time on the transmission via multiple hops than N-packets taking the normal path to transfer over a single hop. This implies that the wakeup delay has less negative impact on the virtual bypass paths. Thus, it is more beneficial for packets to take the virtual bypass paths to avoid the negative impact of the power gating.

5.5.4 Example of Our Power Gating Approach

In this section, we use the example shown in Figure 5.3 to clearly illustrate our EVC-based power gating approach.

In Figure 5.3(a), at time $T = 0$, $Router0$ and $Router1$ are powered-on and $Router2$ and $Router3$ are powered-off. $Router0$ is going to send an E-packet (the red blocks in Figure 5.3) to $Router3$ by using the virtual bypass path, so $Router0$ asserts the WU_{EVC} signal to wakeup $Router3$. $Router1$ is going to send one packet to $Router3$, but there is no virtual bypass path available, so $Router1$ treats this packet as a N-packet (the blue blocks in Figure 5.3) and sends it by using the normal path to $Router2$ first. So, $Router1$ has to asserts the WU signal to wakeup $Router2$.

At time $T = 1$, $Router2$ and $Router3$ receive the WU and WU_{EVC} and begin to power on, respectively. At time $T = 0, 1, 2, 3$, $Router1$ executes the router pipeline stages for its N-packet. The head flit of the N-packet leaves $Router1$ at time $T = 3$. At time $T = 4$, this head flit is going through the link, as shown in Figure 5.3(b). At time $T = 2$, $Router2$ and $Router3$ de-asserts the PG_{EVC} signals, but the E-packet

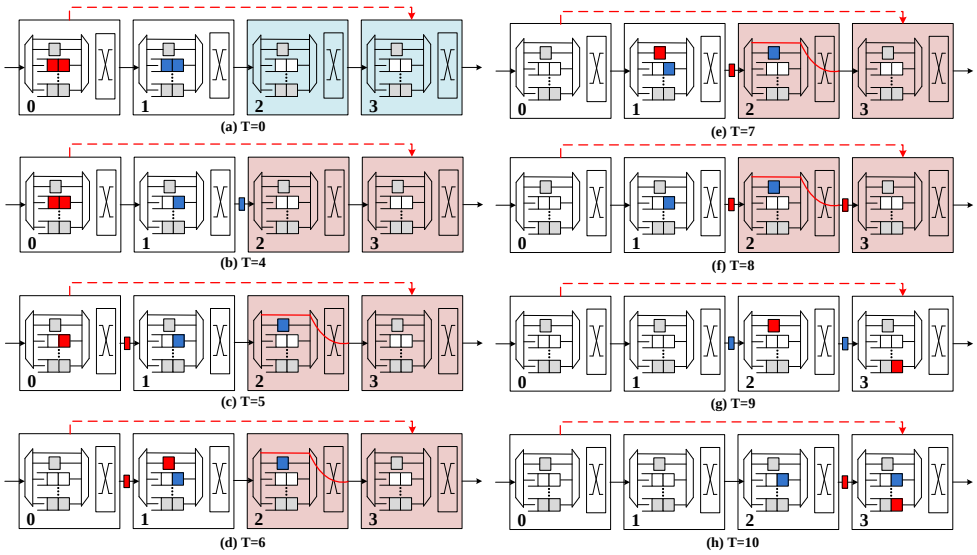


Figure 5.3: An example of our power gating approach.

is still blocked for one clock cycle at $Router0$. So, at time $T = 4$ (Figure 5.3(b)), the E-packet has not been sent yet.

In Figure 5.3(c), at time $T = 5$, the head flit of the N-packet reaches $Router2$ and $Router2$ holds this head flit at its EVC latch. At the same time, in $Router2$, one bypass path is setup by using the direct link and the crossbar. The head flit of the E-packet leaves $Router0$ and is traversing the link.

In Figure 5.3(d), at time $T = 6$, as $Router2$ has to execute the router pipeline stages for the N-packet. The head flit of the N-packets has to occupy the EVC latch for multiple clock cycles. For the E-packet, the head flit reaches $Router1$ and is held at the EVC latch. The tail flit of the E-packet also leaves $Router0$.

In Figure 5.3(e), at time $T = 7$, the head flit of the E-packet leaves $Router1$ and the tail flit of the E-packet is held at the EVC latch of $Router1$.

In Figure 5.3(f), at time $T = 8$, the head flit of the E-packet directly goes through the directly built bypass path in $Router2$, and is traversing the link from $Router2$ to $Router3$. The tail flit of the E-packet is traversing the link from $Router1$ to $Router2$.

In Figure 5.3(g), at time $T = 9$, the head flit of the N-packet leaves $Router2$ and the bypass path in $Router2$ is demolished. For the E-packet, the head flit reaches its destination $Router3$. $Router3$ is just fully charged and stores this flit into the allocated E-VC. The tail flit of the E-packet is held at the EVC latch in $Router2$.

In Figure 5.3(h), at time $T = 10$, the head flit of the N-packet is stored in $Router3$

and the tail flit of this N-packet is stored in $Router2$. As $Router2$ and $Router3$ are already full charged. These flits are stored in the corresponding N-VCs.

This example clearly shows that, by temporarily holding the packets in the EVC latches, the powered-off/ being charged routers can keep certain transmission ability to transfer N-packets. Thus, the N-packet can avoid as much as possible to be blocked by the powered-off/being charged routers. Furthermore, this process does not block the virtual bypass paths at all.

5.5.5 Resolving Starvation

Starvation is a common issue in EVC-based NoCs [KPKJ07]. When an E-packet goes through an intermediate router along one virtual bypass path, the E-packet has the highest priority and the intermediate router has to send it first. If the source router continuously transfers E-packets through the virtual bypass path, the N-packets in the intermediate router cannot get a chance to be sent and starvation occurs. In order to resolve the starvation, we use the approach provided in [KPKJ07] to detect the starvation and then we use our novel approach to temporarily freeze the related virtual bypass paths. For example, considering Figure 5.2(a), if $Router01$ continuously sends E-packets to $Router04$ or $Router02$ continuously sends E-packets to $Router05$, $Router03$ cannot send packets to its downstream $Router04$. Once such starvation occurs, $Router03$ needs to freeze these two virtual bypass paths. To simplify the control between routers, we use two different ways to freeze these two virtual bypass paths: 1) To freeze the virtual bypass path from $Router01$ to $Router04$, $Router03$ informs the sink $Router04$ to assert PG_{EVC} in the direction $X-$. In this way, $Router01$ cannot send E-packets to $Router04$; 2) At the same time, to freeze the virtual bypass path from $Router02$ to $Router05$, $Router03$ informs the source $Router02$ to stop allocating E-VCs in the $X+$ direction to packets. In this way, $Router02$ cannot send E-packets to $Router05$ and the virtual bypass path is freezed. Thus, as all the virtual bypass paths through $Router03$ are freezed, no E-packets prevent $Router03$ to send its packets, thereby resolving the starvation. When the packets, initially affected by the starvation, leave $Router03$, then $Router03$ informs $Router04$ to de-assert the PG_{EVC} signal as well as $Router03$ allows $Router02$ to allocate E-VCs to packets. In this way, the frozen virtual bypass paths are activated and can be used again.

5.6 Experimental Results

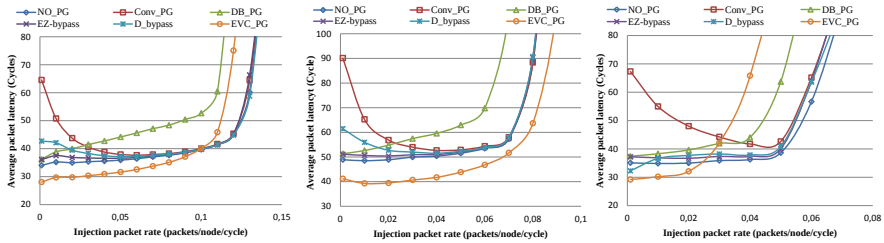
In order to evaluate our EVC-based power gating approach in terms of performance and power consumption, we have implemented our approach using the full-system simulator called Agate [CZPP16]. Agate is based on the widely used full-system

Table 5.1: Parameters used in experiments.

Network topology	8×8 mesh
Router	4-stage pipeline
Virtual channel	(1 N-VC, 1 E-VC)/VN, 3 VNs,
Input buffer size	1-flit/ ctrl VC, 5-flit / data VC
Routing algorithm	X-Y DoR
Link bandwidth/delay	128 bits/cycle, 1 clock cycle
Wakeup delay	8 clock cycles
Break even time	10 clock cycles
T_{idle_detect}	8 clock cycles
$MARGIN_{EVC}$ / $MARGIN$	6/4 clock cycles
Private I/D L1$	32 KB
Shared L2 per bank	256 KB
Cache block size	16 Bytes
Coherence protocol	Two-level MESI
Memory controllers	4, located one at each corner

simulator GEM5 [BBB+11] and Agate supports the simulation of the key items in NoC power gating techniques. The NoC model and power model used in Agate are based on Garnet [AKPJ09] and Dsent [SCK+12], respectively. The key parameters used in our experiments are shown in Table 5.1. We choose a four-stage pipeline router. There are three virtual networks (VNs): two data VNs and one control VN. In each input port, there is one N-VC and one E-VC for each VN. The value of the wakeup delay and break even time (BET) are set according to the related works [CZPP15] and [CP12]. Based on the NoC configuration, we set T_{idle_detect}, $MARGIN_{EVC}$, and $MARGIN$ such that we keep the correctness of the NoC.

For comparison purpose, we have implemented the following power gating approaches: (1) NO_PG: the baseline NoC without power gating; (2) Conv_PG: conventional power-gating NoC, which is deeply optimized by sending WU (Look ahead [MKWA08]) and de-asserting PG signals [CZPP16] in advance, thus 6 clock cycles of the wakeup delay are hidden in our experiments; (3) DB_PG [WNWS17]: the NoC with our DB-based power gating approach introduced in Chapter 3. In each input port of a router, a one-flit size duty buffer is added to implement the Duty Buffer approach. (4) EZ-bypass [ZL18]: the power gating NoC with the EZ-bypass scheme to reduce the negative impact of the power gating process. Compared with other bypass-based related approaches [CP12, FTKH16, BHW+17], EZ-bypass is more flexible to allow packets to bypass the powered-off routers. (5) D_bypass: the NoC with our D-bypass power gating approach introduced in Chapter 4. (6) EVC_PG: the NoC with our EVC-based power gating approach.



(a) Uniform random (b) Bit-complement (c) Transpose

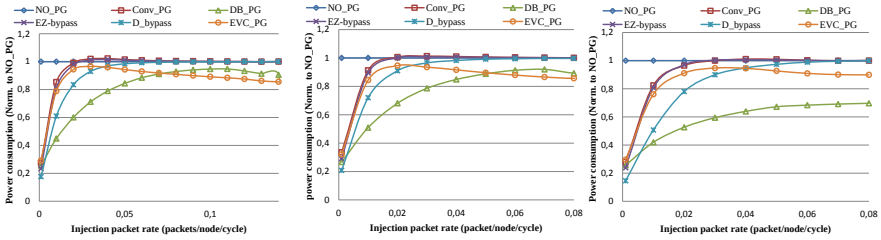
Figure 5.4: Latency across different injection rates.

5.6.1 Evaluation on Synthetic Workloads

In order to explore the behaviour of our EVC_PG, in this section, we evaluate the performance and power consumption of our EVC_PG approach under synthetic traffic patterns. We select three synthetic traffic patterns: 1) Uniform random: packets' destinations are randomly selected; 2) Bit-complement: packets from source router (x, y) are sent to destination router (N-x, N-y), N is the number of routers in the X and Y dimensions of a NoC; 3) Transpose: packets from source router (x, y) are sent to destination router (y, x).

Effect on NoC Network Latency

Figure 5.4 shows the average packet latency under different injection rates. Compared with NO_PG, Conv_PG, DB_PG, EZ-bypass, and D_bypass, our EVC_PG has the lowest average packet latency. These results indicate that our EVC_PG can effectively reduce the negative impact of the wakeup delay and can be used to achieve low latency communication. On the other hand, our EVC_PG has lower saturation points than NO_PG, Conv_PG, EZ-bypass, and D_bypass for the Uniform random and Transpose patterns, but has higher saturation point for the Bit-complement pattern. The lower saturation points indicate that our EVC_PG causes some throughput loss. This is because, in order to support the EVC scheme, the VCs in our EVC_PG are partitioned into E-VCs and N-VCs, which may undermine the flexibility and effectiveness of VCs. Since, Conv_PG, EZ-bypass, and D_bypass are based on NO_PG, they have the same saturation points as NO_PG. However, the impact caused by the partition of E-VCs and N-VCs highly depends on the traffic pattern. Thus, for Bit-complement, our EVC_PG achieves higher saturation point.



(a) Uniform random (b) Bit-complement (c) Transpose

Figure 5.5: Power consumption across different injection rates.

Effect on NoC Power Consumption

Figure 5.5 shows the power consumption normalized to NO_PG under different injection rates. When the injection rate is around 0.001 packets/node/cycle, our EVC_PG has slightly higher power consumption than Conv_PG, EZ-bypass, and D_bypass, but much lower than NO_PG. This is because, in order to avoid packets to be blocked by powered-off routers, we always keep some components powered on in the powered-off routers, which causes extra power consumption but this power consumption is rather low. When the injection rate increases, more and more routers become busy and cannot be powered off. The power reduction in Conv_PG, DB_PG, EZ_PG, D_bypass, and EVC_PG becomes lower and lower, but DB_PG has much higher power reduction than the other approaches. This is because DB_PG can separately power off VCs in each input port of routers whereas Conv_PG, EZ-bypass, D_bypass, and EVC_PG can power off a router only when all of the input ports of the router are idle. Thus, DB_PG fully utilizes the idle time of each input port to reduce the power consumption.

When the injection rate is higher than 0.02 packets/node/cycle in Figure 5.5(a), 0.02 packets/node/cycle in Figure 5.5(b), and 0.03 packets/node/cycle in Figure 5.5(c), Conv_PG and EZ-bypass become ineffective in reducing the power consumption, while DB_PG, D_bypass, EVC_PG still can effectively reduce the power consumption. The power reduction in our EVC_PG is due to the fact that packets can also bypass powered-on routers, which saves some dynamic power.

When the injection rate further increases, the dynamic power takes higher and higher portion of the total power consumption. Our EVC_PG reduces more the dynamic power consumption, which causes the curves for our EVC_PG in Figure 5.5(a), Figure 5.5(b), and Figure 5.5(c) to decline. As a result, when the injection rates are higher than 0.04 packets/node/cycle in Figure 5.5(a), 0.03 packets/node/cycle in Figure 5.5(b), and 0.04 packets/node/cycle in Figure 5.5(c), our EVC_PG consumes less power than D_bypass. When the injection rates are higher than 0.07 packets/node/cycle

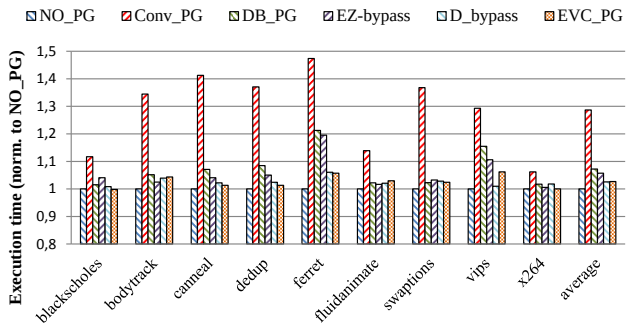


Figure 5.6: Execution time.

in Figure 5.5(a) and 0.05 packets/node/cycle in Figure 5.5(b), our EVC_PG has lower power consumption than DB_PG. However, in Figure 5.5(c), DB_PG has always lower power consumption than our EVC_PG. This is because DB_PG and EVC_PG reach their saturation points at low packet injection rates as shown in Figure 5.4(c). So, the dynamic power consumption takes small portion of the total power consumption. As a consequence, the efficient reduction of the dynamic power consumption in our EVC_PG does not play a significant role in reducing the total power consumption in this case, whereas DB_PG more efficiently reduces the static power consumption by separately powering off input ports of routers, leading to better reduction of the total power consumption in this case.

5.6.2 Evaluation on Real Application Workloads

In this section, we use real application workloads to compare the approaches in terms of the application performance, the average network latency, and the NoC power consumption. To do so, we use nine applications from the Parsec [BKSL08] benchmark suite.

Effect on Application Performance

Figure 5.6 shows the total execution time of the nine applications, which is normalized to the baseline NO_PG, and the tenth set of bars in Figure 5.6 gives the average results over these nine applications. Our EVC_PG approach causes less performance penalty (execution time increase) than the related approaches. Compared with the baseline NO_PG, our EVC_PG causes, on average, 2.67% performance penalty, which is less than the 28.67% performance penalty in Conv_PG, 7.24% in DB_PG, and 5.69% in EZ-bypass, and comparable with the 2.55% performance penalty in

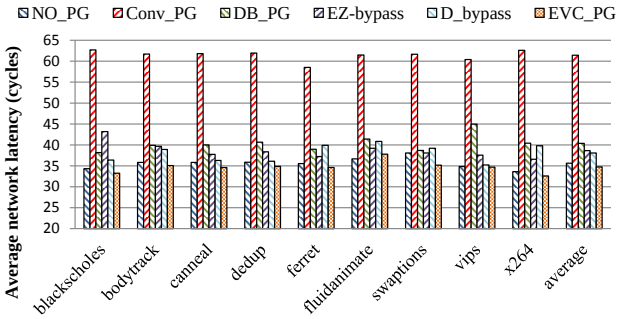


Figure 5.7: Average network latency.

D_bypass. For benchmarks blackscholes and x264, our EVC_PG has slightly lower execution time than NO_PG. In the vips benchmark, our EVC_PG has its highest performance penalty of 6.17%, which is still lower compared to Conv_PG, DB_PG, and EZ-bypass, but higher than D_bypass. For the ferret benchmark, Conv_PG, DB_PG, EZ-bypass and D_bypass have their highest performance penalty of 47.39%, 21.21%, 19.51%, and 6.03%, respectively.

Effect on Average Network Latency

Figure 5.7 shows the average network latency across the nine applications. Compared with NO_PG across the applications, the average network latency in our EVC_PG approach is slightly lower, whereas Conv_PG, DB_PG, EZ-bypass, and D_bypass have higher average network latency compared to NO_PG. As DB_PG uses a fined-grain power gating scheme, packets in DB_PG suffer more wake up processes. As a consequence, DB_PG has much higher average network latency than our EVC_PG and EZ-bypass. EZ-bypass allows packets to bypass powered-off routers, but packets have to stay at powered-off routers for a long time experiencing the router pipeline stages. In contrast, in our EVC_PG, the packets can bypass the intermediate routers without the need to experience the router pipeline stages. Thus, our EVC_PG has lower average network latency than EZ_PG. While, compared with D_bypass, which needs extra time to reserve the bypass latch in the powered-off routers, our EVC_PG is more efficient to transfer packet to bypass the powered-off routers.

Even though our EVC_PG has a slightly lower average network latency compared to NO_PG (see Figure 5.7), our EVC_PG still causes a slightly higher execution time in most of the applications compared to NO_PG (see Figure 5.6). This is because EVC_PG causes unbalance NoC resource allocation when E-packets take the virtual bypass paths to bypass intermediate routers and have a higher priority compared to N-packets.

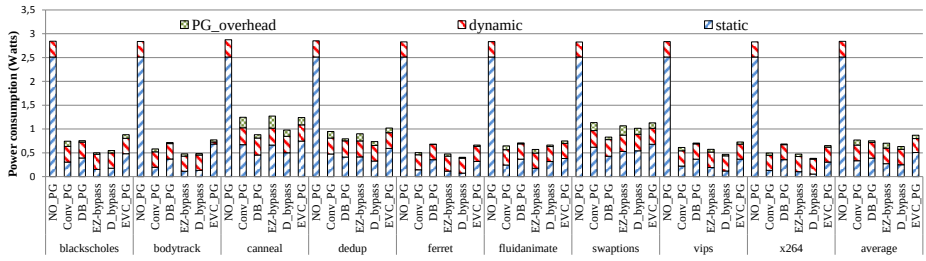


Figure 5.8: Power consumption.

Effect on NoC Power Consumption

Figure 5.8 shows the breakdown of the NoC power consumption across the nine applications and the tenth set of bars shows the average over these nine applications. The NoC power is broken down into three parts: the power consumption caused by power gating (PG_overhead) and the static/dynamic power consumption of routers (static/dynamic).

As shown in Figure 5.8, our EVC_PG approach consumes slightly higher total power than the related approaches Conv_PG, DB_PG, EZ_PG, and D_bypass. This is because our EVC_PG needs more components in a router to be always powered on, which causes slightly more static power consumption compared to Conv_PG, DB_PG, EZ_PG, and D_bypass. As the traffic workloads in real applications are low, the dynamic power consumption is low. As a result, the dynamic power reduction in our EVC_PG does not play a significant role in reducing the total power consumption. Compared with NO_PG, our EVC_PG reduces on average 68.29% of the total power consumption, which is comparable with the 72.94%, 73.56%, 75.30%, and 77.77% reduction of the total power consumption in Conv_PG, DB_PG, EZ-bypass, and D_bypass, respectively.

5.7 Discussion

In this chapter, we propose an EVC-based power gating approach. In our approach, packets can take pre-defined virtual bypass paths to bypass intermediate routers that may be powered-on or powered-off. Furthermore, even though some packets do not take a virtual bypass path, our approach tries to ensure that these packets avoid as much as possible blocking in the powered-off routers. As a result, our approach reduces more efficiently the packet latency increase caused by power gating. Furthermore, by allowing packets to bypass powered-on routers to reduce dynamic power

consumption, our approach can achieve lower power consumption under high traffic workloads.

Chapter 6

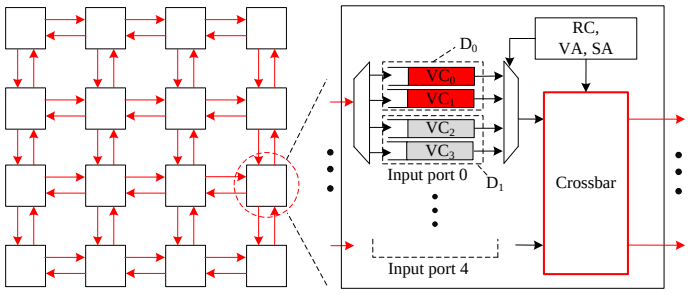
Energy-Efficient Confined-interference Communication

Peng Wang, Sobhan Niknam, Sheng Ma, Zhiying Wang, Todor Stefanov,
"Surf-Bless: A Confined-interference Routing for Energy-Efficient Communication in NoCs",
in Proceeding of the 56th Annual Design Automation Conference (DAC), Las Vegas, United States, 2019,
Winner of 2019 HiPEAC Paper Award

In this chapter, we present our Surfing on a Bufferless NoC (Surf-Bless) routing approach, which corresponds to **Contribution 4** introduced in Section 1.4 to solve the research **Problem 2** formulated in Section 1.3.2. The remainder of this chapter is organized as follows. Section 6.1 further introduces the high power consumption problem caused by supporting confined-interference communication on conventional NoCs. Section 6.2 summarizes the main contributions in this chapter. To better understand our Surf-Bless routing approach, Section 6.3 gives some background information about the Surf-routing [WGO+13] and the BLESS-routing [MM09], that have inspired our Surf-Bless routing approach. Section 6.4 provides an overview of the related work. It is followed by Section 6.5, which elaborates our novel Surf-Bless routing approach. Section 6.6 introduces the experimental setup and presents experimental results. Finally, Section 6.7 concludes this chapter.

6.1 Problem Statement

A Network-on-Chip (NoC) with low latency, high bandwidth, and good scalability is a promising communication infrastructure for large size many-core systems.



(a) Link time-multiplexing (b) VC assigment

Figure 6.1: Confined-interference communication on a NoC.

However, as the NoC resources are shared by different packets, there may be significant interference between packets, which influences the packet transmission time. If the packets come from different applications, the temporal behaviour of an application depends on the behavior of other applications, thereby making a many-core system non-composable. In a composable system, applications are completely isolated and cannot affect each others functional or temporal behaviors [HGBH09]. One way to remove the applications dependency caused by the interference between packets, a necessary step to make the system composable, is to group the packets of different applications into different domains and keep non-interference between domains [WGO+13, P+16]. Thus, the packets in one domain have no influence on the transmission time of the packets in other domains. Such packet transmission scheme we call confined-interference communication.

The interference between packets is caused by the contention on the shared resources of a NoC, such as virtual channels (VCs), crossbars, input/output ports, and links. In order to support confined-interference communication, these shared resources should be separated in space or in time. A straightforward way to implement confined-interference communication is to assign different VCs to store packets of different domains (isolation in space) and to split the utilization time of input/output ports, crossbars and links into multiple time slots, then assign different time slots to different domains at design-time (isolation in time). For example, as shown in Figure 6.1, the NoC is used to transfer packets of two domains (D_0 and D_1). The VCs (VC_0, VC_1, VC_2, and VC_3) are assigned to different domains. VC_0 and VC_1 are assigned to D_0. VC_2 and VC_3 are assigned to D_1. Packets of D_0 and D_1 can be stored only in their own VCs. At even clock cycles, only packets of D_0 can go through the crossbars, output ports, and links (distinguished by the red color in Figure 6.1(a) and 6.1(b)). At odd clock cycles, only packets of D_1 can go through the crossbars, out-

put ports, and links. In this way, there is no interference between D_0 and D_1 because packets of different domains are completely isolated in space and time. However, to separate the packets in space, each domain requires at least one VC. This leads to a large number of buffers when there are a lot of domains and causes high static and dynamic power consumption [CZPP15, WNWS17]. For example, the NoC with two 16-flit VCs per input port in the Intel 80-tile chip [HVS+07] and the NoC with four 1-flit VCs and two 3-flit VCs per input port in the SCORPIO chip [DCS+14] consume up to 28% and 19% of the total system power, respectively, see Table 1.1. In fact, such high power consumption of a NoC has become the major bottleneck that prevents applying NoCs on many-core systems [EBA+11]. Therefore, it is critical to implement confined-interference communication with low power/energy consumption.

A bufferless NoC is a low power consumption communication fabric. By eliminating VCs (buffers) in routers, bufferless NoCs [MM09, FCM11, LSMJ16] can significantly reduce the power consumption of a NoC. However, as there are no VCs in routers to store packets, packets have to keep moving on the bufferless NoC. Therefore, when contention occurs between packets on the same output port, some of the packets must be deflected to other output ports, which makes the interference between packets severe. As a consequence, the conventional bufferless NoCs do not support confined-interference communication. Furthermore, as VCs are eliminated, the bufferless NoCs cannot easily accommodate the transfer of multiple class packets in a cache coherence protocol, because they cannot isolate/confine the interference between difference class packets.

6.2 Contributions

Considering the discussion above, in this chapter, we address the problem of how to achieve confined-interference communication on a bufferless NoC. In order to take advantage of the low power consumption of a bufferless NoC and to achieve confined-interference communication, we propose a novel routing approach called Surfing on a Bufferless NoC (Surf-Bless). The specific novel contributions of this chapter are the following:

- We propose a specific assignment and scheduling of domains onto input/output ports, crossbars, and links. This specific assignment and scheduling can be visualized as multiple “waves" (see Figure 6.3) that move in space and time over the NoC in a specially designed repetitive pattern. Different domains are assigned to different waves and the packets of a domain “surf" on their own waves, thereby achieving packet routing with no interference between packets belonging to different domains. Our specific repetitive movement of the waves guarantees that packets surfing on any wave do not need to stop and wait at any

router along their path to the destination. Thus, VCs are not needed in routers, thereby enabling the utilization of bufferless NoCs to transmit packets.

- We propose an extension for the architecture of a conventional bufferless NoC router that implements in hardware our specific assignment and scheduling introduced in the aforementioned contribution. By adding three simple hardware schedulers in each router, our Surf-Bless routing approach is implemented in a distributed way, which makes our approach scalable and applicable for large size NoCs.

- By experiments, we show that our Surf-Bless routing approach is effective in supporting confined-interference communication and consumes much less energy than Surf-routing [WGO+13]. Furthermore, our Surf-Bless approach overcomes the drawback of the conventional bufferless NoC [MM09] which cannot support the transfer of multiple class packets for a cache coherence protocol.

6.3 Background

In order to better understand the novel contributions of this chapter, in this section, we give some background information about the Surf-routing [WGO+13] which realizes confined-interference communication and requires VCs (buffers) in routers, as well as we briefly describe the BLESS-routing [MM09] which is a typical bufferless routing approach with no support for confined-interference communication.

6.3.1 Surf-routing

As introduced in the begin of this chapter, packets in VCs assigned to a domain can be transferred only in the corresponding time slots in order to achieve isolation in time. As a consequence, packets have to be blocked extra clock cycles at each router to wait for their own time slots, which results in high packet latency. In order to avoid this extra blocking time, Surf-routing [WGO+13] assigns and schedules the domains on two kinds of “waves": the north-west wave W_{NW} and the south-east wave W_{SE}. The wave pattern is shown in Figure 6.2. These waves move in space and in time over the network. The red links (bolded arrows) on the waves indicate that these links are assigned to domain D_0 and only packets in domain D_0 can go through these red links. The packets of domain D_0 “surf" on the waves to the next routers. When packets arrive at the next routers, the waves also arrive at the same routers. Thus, packets can be continuously transferred and do not need to spend too much extra time on waiting for their time slots. Therefore, the packet latency in a domain can be reduced.

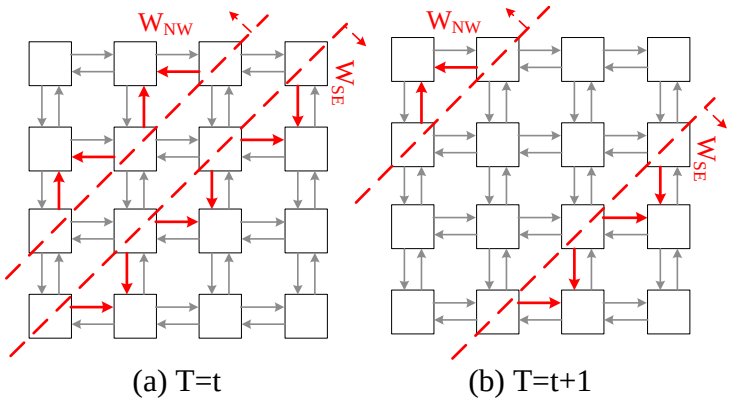


Figure 6.2: The wave pattern in Surf-routing.

However, VCs in routers are still necessary for the Surf-routing to separately store packets because when contention occurs between packets in the same domain, some packets must be stored in VCs assigned to the same domain. As a consequence, each domain in Surf-routing needs at least one VC.

6.3.2 BLESS-routing

By eliminating the VCs (buffers) in a router, BLESS-routing [MM09] can significantly reduce the power consumption. As there are no VCs to temporarily store packets, packets in BLESS-routing have to keep moving in the links (or in the router pipeline). When multiple packets contend for the same output port in a router, some of the packets have to be deflected to other output ports, which requires that each packet should always find a free output port to go. This packet deflection is guaranteed because the number of the input ports in a router is equal to the number of the output ports. The packet deflection may cause a livelock problem, i.e., packets keep moving in the network but never reach their destination. In order to avoid such livelock problem, the old-first arbitration policy [DT04] is used to prioritize packets when they contend for the same output port. When a packet becomes the oldest, it cannot be deflected to other output ports any more. Thus, the livelock is avoided.

6.4 Related Work

As VCs (buffers) consume a large portion of the NoC power, several bufferless NoCs [MM09, FCM11, LSMJ16] have been proposed to reduce the power consumption. Based

on packet deflection, BLESS [MM09] eliminates the need for VCs in routers and proves that the bufferless NoC is effective when the links utilization is low. CHIPPER [FCM11] proposes a permutation network deflection routing which simplifies the router structure and reduces the bandwidth overhead caused by the old-first arbitration policy [DT04]. Without deflecting packets but dropping packets, Runahead [LSMJ16] further simplifies the router structure and it can be used to achieve low latency communication. By eliminating the VCs, [MM09, FCM11, LSMJ16] can significantly reduce the NoC power consumption. However, as there are no VCs to temporarily store packets, [MM09, FCM11, LSMJ16] cannot support confined-interference communication, thus cannot be used in composable many-core systems. In contrast, our Surf-Bless routing is an approach to achieve confined-interference communication on a bufferless NoC. Thanks to our Surf-Bless routing approach, it becomes possible for a bufferless NoC to be used in composable systems to achieve energy-efficient communication. Thus, compared with [MM09, FCM11, LSMJ16], our Surf-Bless routing extends the applicability of the bufferless NoCs.

As introduced in Section 6.3.1, by scheduling packets in different domains onto different waves, the Surf-routing [WGO+13] can reduce the packet latency caused by waiting for the corresponding time slots. As an extension of Surf-routing, the router pipeline in PhaseNoC [P+16] is time-multiplexed by all domains. PhaseNoC reduces the part of the hardware overhead caused by separating the NoC resources for different domains. However, each domain in Surf-routing and PhaseNoC needs at least one VC, which may lead to a large number of buffers and cause high power consumption in case of multiple domains. In contrast, based on a bufferless NoC, our Surf-Bless routing approach achieves confined-interference communication without the need of VCs. Thus, our Surf-Bless routing approach is much more power efficient than the approaches in [WGO+13] and [P+16].

By reserving time slots along a routing path, [GDR05, HSG09, SMG14] realize contention-free routing and guarantee that there is no interference between packets. So, the NoC provides composable and predictable services. As data flows are time-multiplexed to use the NoC resources, the router structures in [GDR05, HSG09, SMG14] are simple and do not need too much buffers, so the power consumption is low. However, to achieve the contention-free routing, [GDR05, HSG09, SMG14] need to globally schedule all data flows on the NoC at design time. As a consequence, the contention-free routing in [GDR05, HSG09, SMG14] is only applicable to static data (packets) traffic, where traffic information, such as traffic patterns, the data volume of each data flow, etc., are known at design time. So, for dynamic traffic patterns, i.e., traffic unknown at design time, [GDR05, HSG09, SMG14] are not applicable. In contrast, our Surf-Bless routing approach schedules domains on the NoC and does not need to schedule each packet. Thus, the scheduling in our Surf-routing approach

is simpler than [GDR05, HSG09, SMG14]. Furthermore, our Surf-Bless routing approach does not need the aforementioned traffic information at design time to schedule domains onto a NoC. Thus, our Surf-Bless routing is applicable to dynamic traffic patterns as well, providing composability services to a system but cannot provide complete predictability for dynamic traffic.

6.5 Surf-Bless Routing Approach

The key idea of our novel Surf-Bless routing approach is to assign and schedule domains onto waves that move in space and time over the NoC in a specially designed repetitive pattern. Packets of a domain can go only through the input ports, crossbars, output ports, and links on the waves assigned to the same domain. Thus, as there is no interference between waves, it is guaranteed that there is no interference between different domains. Furthermore, based on the special wave pattern in our Surf-Bless routing approach, it is guaranteed that in a router, the number of input ports assigned to one domain at the current clock cycle, is equal to the number of output ports assigned to the same domain at the next clock cycle. This makes it possible to always deflect packets if needed in order to keep packets moving in the network instead of storing them temporarily in VCs. Thus, we achieve confined-interference communication on a bufferless NoC, i.e., a NoC without VCs.

6.5.1 Wave Pattern in Surf-Bless

In order to keep packets of a domain traveling to their destinations without the need to stop and wait at routers, we assign and schedule domains onto NoC resources in a special repetitive pattern, which can be visualized as the “waves" shown in Figure 6.3. For the sake of clarity, in Figure 6.3, we show only two waves which are distinguished by the blue color and the red color. These waves move in space and time over the network and the pattern repeats after 6 time slots T. In the following explanation, we take the red wave as an example to describe our special wave pattern. The wave consists of three sub-waves, the south-east sub-wave (W_{SE}), the north sub-wave (W_N), and the west sub-wave (W_W). These sub-waves of a wave must respect the following two rules: **Rule-1**: when W_{SE} reaches the routers on the south-border of the network or the routers on the east-border of the network, W_N at the router on the south-border and W_W at the router on the east-border are triggered. For example, at time slot $T = 0$, W_{SE} reaches router 30 on the south-border and router 03 on the east-border. At the same time, W_N at router 30 and W_W at router 03 are triggered to move over the network. **Rule-2**: when W_N reaches a router on the north-border or W_W reaches a router on the west-border, W_{SE} must arrive at the corresponding routers as well. For example, at time slot $T = 4$, W_N and W_W reach the north-border at router 01

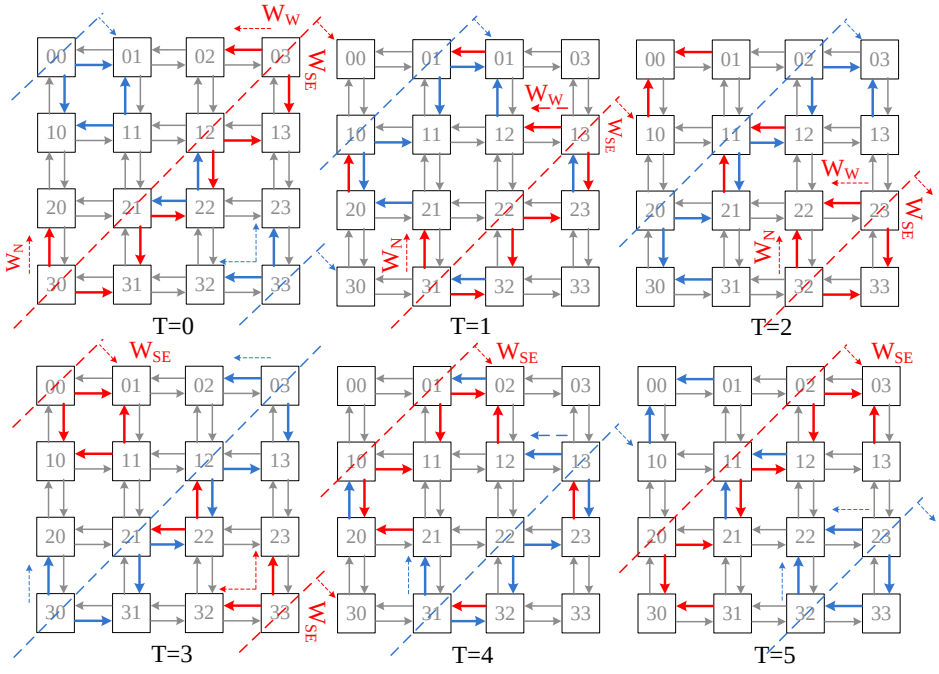


Figure 6.3: Wave pattern in Surf-Bless routing.

and the west-border at router 10, respectively. At the same time, W_{SE} reaches the same router 01 and 10. Constrained by these two rules, a wave moves over the network repetitively, like a reverberating wave. As there is no overlapping between any two waves, it can be guaranteed that there is no interference between domains that are assigned to different waves.

In contrast to the wave pattern in the Surf-routing shown in Figure 6.2, the wave pattern in our Surf-Bless routing guarantees that a router has a certain number of input ports receiving packets from the links belonging to a given wave at time slot (clock cycle) T. At the next time slot $T + 1$, this router has the same number of output ports sending packets to the links belong to the same wave. Thus, thanks to our special wave pattern, it is possible to deflect packets when contention occurs on the same output port. Therefore, the packets can keep moving in the network and we can achieve confined-interference communication on a bufferless NoC.

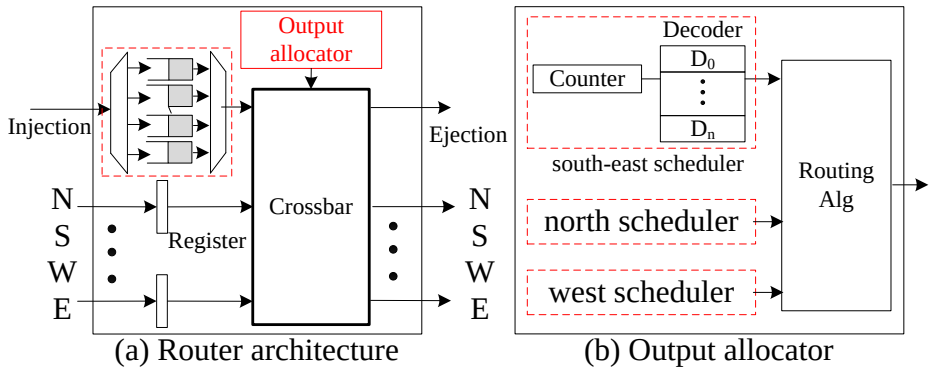


Figure 6.4: Router architecture in Surf-Bless.

6.5.2 Router Architecture

In order to assign and schedule domains on the waves, shown in Figure 6.3, to support confined-interference communication, we extend a conventional bufferless router, as shown in Figure 6.4. Our extensions are indicated by the red color. The router consists of five input ports, five output ports, one output allocator and one crossbar.

In the input ports at the directions of north (N), south (S), west (W), and east (E), VCs are eliminated but one register per input port is used to construct the router pipeline. For the injection input port, there are multiple VCs. These VCs are necessary for a bufferless NoC to temporarily store packets coming from the network interface. These VCs can be put in the router [MM09] or put in the network interface [FCM11]. We put these VCs in the router to simplify the control of the packet injection. In contrast to [MM09] and [FCM11] with only one VC in each injection input port, we utilize multiple VCs and each VC is assigned to one domain. In this way, the packets in a domain cannot be blocked by packets in other domains, i.e., the Head-of-Line blocking [DT04] between packets in different domains is avoided.

The output allocator is used to implement our novel wave pattern introduced in Section 6.5.1. Based on our routing algorithm described in Section 6.5.3, it allocates the output ports for each incoming packet. In order to implement our wave pattern, three schedulers, i.e., the south-east scheduler, the north scheduler, and the west scheduler, are realized in the output allocator as shown in Figure 6.4(b). Each scheduler corresponds to one sub-wave and is responsible for a pair of input ports and output ports. The south-east scheduler corresponds to sub-wave W_{SE} and is responsible for the pair $\{input = \{N, W, Injection\}, output = \{S, E, Ejection\}\}$. This implies that packets can be injected or ejected only on the W_{SE} sub-wave. The north scheduler and the west scheduler correspond to W_N and W_W and are responsible for

$\{input = \{S\}, output = \{N\}\}$ and $\{input = \{E\}, output = \{W\}\}$, respectively.

These schedulers have similar hardware structure. Taking the south-east scheduler as an example, shown in Figure 6.4(b), it consists of a counter and a decoder. The counter cyclically counts from 0 to $S_{max} - 1$ indicating the wave index of the current wave at the router. S_{max} is the maximal number of waves, which is determined by the NoC size, the pipeline stages of a router, and the link delay. It can be calculated by $S_{max} = 2 \times P \times (N - 1)$, where P is the hop delay in clock cycles (the delay of a packet to go through one router and one link), and N is the number of routers in one dimension on a $N \times N$ NoC. (The maximal number of waves is also equal to the repeat period of a wave. For example, in Figure 6.3, the repeat period of the red wave is 6 time slots (the hop delay $P = 1$) in this example, so the maximal number of waves in this example is 6.) By setting specific initial values for the counters in the router, we can realize the wave pattern shown in Figure 6.3. The initial value for each counter in a router (x, y) (x and y are the router positions in each dimension on the NoC) is computed by the following equations:

$$Initial_{SE} = (S_{max} - P \times (x + y)) \mod S_{max} \tag{6.1}$$

$$Initial_{W} = (S_{max} + P \times (x - y)) \mod S_{max} \tag{6.2}$$

$$Initial_{N} = (S_{max} - P \times (x - y)) \mod S_{max} \tag{6.3}$$

The decoder is a table and is used to assign different waves to different domains. Based on these schedulers in each router, we implement our special assignment and scheduling in a distributed way without the need of global control.

6.5.3 Surf-Bless routing algorithm

The Routing Algorithm unit in the output allocator in Figure 6.4(b) is used to allocate output ports for incoming packets. In our Surf-Bless approach, a packet in an input port assigned to a domain can go only through the output ports that are assigned to the same domain. To respect this rule, we propose the following Surf-Bless routing algorithm, which consists of two steps:

Step-1: select the highest priority packet from input ports; For simplicity, our Surf-Bless routing uses the old-first arbitration policy [DT04]. Packets carry “age" information, i.e., the longer the packet moves in the network the older it becomes. The oldest packet has the highest priority. The packets in the injection input ports have the lowest priority.

Step-2: choose an output port in the same domain; First, our Surf-Bless routing algorithm uses X-Y routing to determine the output port for the packet selected in Step-1. If the input port and the determined output port are in the same domain checked by comparing the outputs of the schedulers as well as the output port is not granted to another packet, the output port is granted to the selected packet. If this check fails, our Surf-Bless routing tries the Y-X routing. If the same check fails again, one of the free output ports assigned to the same domains is randomly granted to the packet, thereby deflecting the packet. For a packet in the injection input port, if there is one free output port that is assigned to the same domain with the injection input port, the packet in the corresponding VC can be transferred, otherwise the packet is blocked in the corresponding VC.

As there are no VCs (buffers) for the N, S, W, and E input ports in our routers, the routers need to deflect packets, which may cause livelock and deadlock problems. In Step-1, our Surf-Bless algorithm uses the old-first arbitration policy [DT04] which guarantees that our Surf-Bless routing approach is deadlock and livelock free. Dependent packets may cause the protocol deadlock in a bufferless NoC. For example, the protocol packets described in [HGR07] consist of a request packet and a reply packet, where the reply packet must be injected to the network before a new request packet arrives, otherwise the new request packet cannot be ejected to the network interface. However, in a bufferless NoC, a new request packet in the NoC has higher priority than the reply packets staying at the network interface, so a new request packet blocks the reply packets to be injected into the NoC. As a consequence, cyclic dependency occurs between the request packet and the reply packet, which may result in deadlock. In our Surf-Bless routing approach, such dependent packets can be separately assigned to different domains. As there is no interference between packets in different domains, the cyclic dependency between packets is removed and the deadlock is avoided.

6.6 Experimental Results

In order to evaluate our approach in terms of the performance and the energy consumption, we have implemented our Surf-Bless routing in the full-system simulator GEM5 [BBB+11]. The NoC model and power model are based on Garnet2.0 [AKPJ09] and Dsent [SCK+12], respectively. The key parameters used in our experiments are shown in Table 6.1. Considering the previous works [MM09] and [FCM11], the router pipeline delay in the bufferless NoC is set to 2 clock cycles. For a conventional virtual channel router, we choose a 4-stage pipeline router.

For comparison purpose, we have implemented the following approaches: (1) WH [DT04]: the baseline wormhole NoC, which does not support confined-interface

Table 6.1: Parameters.

Network topology	8×8 mesh
Router	2-stage and 4-stage pipelines
Virtual channel	1 ctrl VC and 2 data VCs
Input buffer size	1-flit/ ctrl VC, 5-flit / data VC
Flit size	128 bits
Routing algorithm	X-Y DOR, Surf and Surf-Bless
Link bandwidth	128 bits/cycle
Private I/D L1$	32 KB
Shared L2 per bank	256 KB
Cache block size	16 Bytes
Coherence protocol	Two-level MESI
Memory controllers	4, located one at each corner

communication; (2) SURF [WGO+13]: a confined-interference NoC with the Surf routing briefly introduced in Section 6.3.1; (3) BLESS [MM09]: the bufferless baseline NoC briefly introduced in Section 6.3.2, which does not support confined-interference communication; (4) SB: our Surf-Bless approach supporting confined-interference communication on a bufferless NoC.

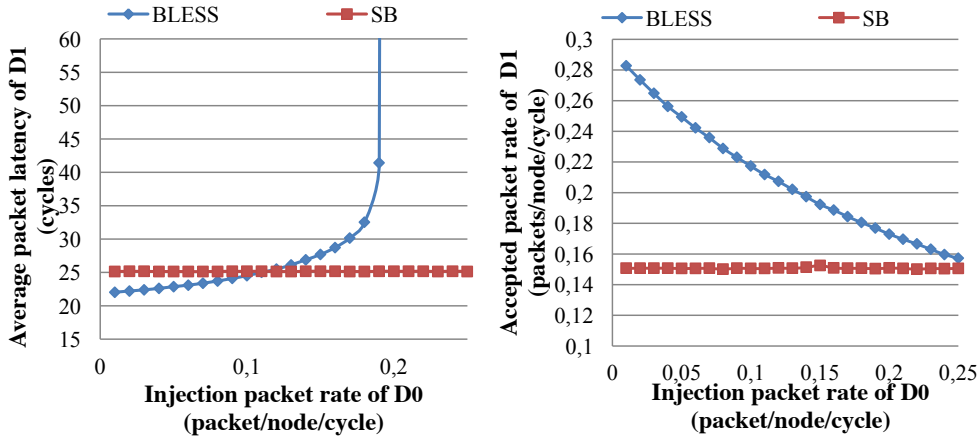
6.6.1 Evaluation on Synthetic Workloads

In this section, we evaluate our approach in terms of energy consumption, average packet latency, and the ability of supporting confined-interference communication. Based on our NoC configuration in Table 6.1 and the S_{max} equation introduced in Section 6.5.2, there are $S_{max} = 2 \times 3 \times (8-1) = 42$ waves. The domains are equally and evenly assigned to these waves. The uniform random traffic pattern described in [DT04] is used to inject 1-flit packets into the NoCs in our experiments.

Confined-interference Communication

In order to test the ability of confining the interference between packets from different domains in our SB, we use two domains D_0 and D_1. Domain D_0 is assigned to the even waves and domain D_1 is assigned to the odd waves. Domain D_0 is regarded as interference traffic. By observing the latency and throughput of domain D_1, we can check if there is any interference between packets in the different domains D_0 and D_1.

Figure 6.5(a) and Figure 6.5(b) show the average packet latency and the maximal throughput provided by our SB and BLESS for domain D_1 considering different injection rates of the traffic generated by domain D_0, respectively. By increasing the injection rate of domain D_0, the average packet latency and the throughput provided by our SB for domain D_1 stays constant, which indicates that SB is effective and



(a) Latency of domain D_1.

(b) Throughput of domain D_1.

Figure 6.5: Non-interference between domains.

guarantees no interference between domains. In contrast, as BLESS does not support confined-interference communication, the average packet latency and throughput provided by BLESS for domain D_1 are significantly impacted by the traffic of domain D_0.

Energy Consumption under Different Number of Domains

In this section, we evaluate the energy consumption considering different number of domains, from one domain (1_D) up to nine domains (9_D). In SURF, each input port (N, S, W, E, and Injection) in a router has multiple 4-flit VCs where VC corresponds to one domain. In contrast, in our SB, as shown in Figure 6.4, only the Injection input port has multiple 4-flit VCs where corresponds to one domain. The domains are equally and evenly assigned to the waves. The packets are equally assigned/injected to each domain. In our experiment, we evaluate the energy consumption of the NoCs in a time period of 1 million clock cycles (the frequency is $1GHz$) under packet injection rate of 0.05 packets/node/cycle.

Figure 6.6 shows the energy consumption across 1_D, 2_D, $\cdots$, 9_D. The NoC energy consumption is broken down into the dynamic and the static router energy consumption and the total link energy consumption. Compared to SURF with different domains, the NoC energy consumption in our SB is significantly reduced. This energy reduction increases when the number of domains increases. This is because, by eliminating the VCs for the N, S, W, and E input ports in a router, the router in our SB has much lower static energy consumption and simpler architecture than the router in SURF. Furthermore, with the number of domains increasing, in our SB, only injection

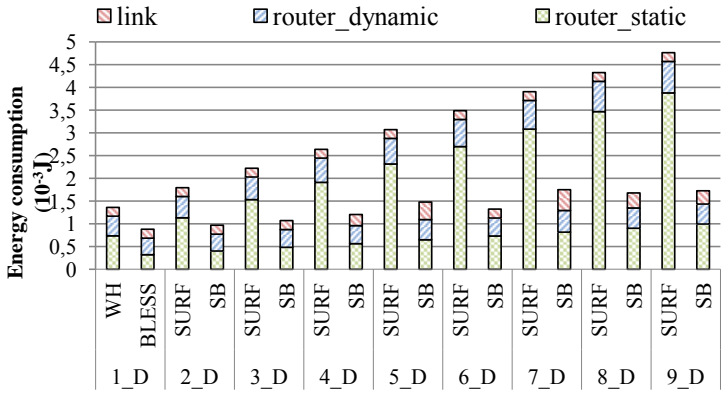


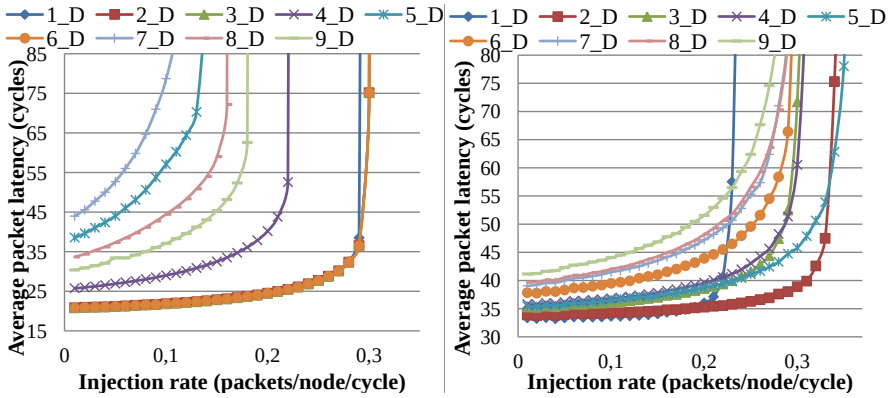
Figure 6.6: Energy consumption across different number of domains.

input ports need to increase the number of VCs to separately store packets for different domains. In SURF, all of the input ports need to increase the VCs to separately store packets for different domains, which causes significant increase of the static energy consumption. On the other hand, compared with BLESS, as our SB needs multiple VCs in the injection input ports, SB causes higher static energy consumption than BLESS.

It should be noted that the dynamic power consumption does not increase linearly with the number of domains for SB. This is because under different number of domains, the average packet latency is different, which will be introduced in Section 6.6.1. As the packet cannot stop in our SB, the higher average packet latency means that packets need to be transferred through more routers, which results in high dynamic power consumption.

Average Packet Latency under Different Number of Domains

Figure 6.7(a) and Figure 6.7(b) show the average packet latency of SB and SRUF under 1_D, 2_D,$\cdots$, 9_D, respectively. In Figure 6.7(a), the curves for 2_D, 3_D, and 6_D are overlapped and have the lowest average packet latency. 4_D, 5_D, 7_D, 8_D, and 9_D have higher average packet latency. This is because, in our SB, packets of a domain can be transferred to the output ports only in the time slots allocated to the this domain. Sometimes, some packets of a domain have to be deflected because some output ports are not in the time slots allocated to the same domain. Therefore, compared with BLESS (1_D), our SB causes more packet deflections in 4_D, 5_D, 7_D, 8_D, and 9_D. However, such high number of packet deflections caused by the time slot allocation does not happen in 2_D, 3_D, and 6_D. This is because, based on



(a) BLESS (1_D) and SB

(b) WH (1_D) and SURF

Figure 6.7: Latency across different number of domains.

the waves for 2_D, 3_D, and 6_D, the time slots of all of the output ports in a router are always allocated to the same domain. Thus, the time slot allocation in our SB does not cause extra packet deflections in 2_D, 3_D, and 6_D. Another reason resulting in the packet latency increase is that, in our SB, the packet injection/ejection happens only on the south-east sub-wave W_{SE}. This means that when a packet in a domain reaches its destination router on the north sub-wave W_N or the west sub-wave W_W, and W_{SE} assigned to the same domain has not reached this router yet, this packet cannot be ejected and has to be deflected to a downstream router, in spite of the fact that this packet has already reached its destination router. This negative impact is significant for 4_D, 5_D, 7_D, 8_D, and 9_D, but this negative impact does not happen for 2_D, 3_D, and 6_D because all of the output ports in a router are always in the time slots allocated to the same domain.

Comparing Figure 6.7(a) with Figure 6.7(b), when the number of domains is 5,7,8, and 9, SB has higher average packet latency than SURF. This is because the packet injection/ejection happens only on the W_{SE} sub-waves in SB. Only when the sub-wave W_{SE} for a domains reaches the destination router, a packet can have a chance to be ejected. Otherwise, the packet has to be deflected. When the number of domains increases, each domain has fewer waves assigned to it. Packets in a domain have less chance to be ejected, but more chance to be deflected. As a consequence, the packet latency sharply increases in SB. In contrast, routers in SURF have VCs to temporarily store packets and do not need to deflect packets. Therefore, even though the packet latency still increases with the increase of the number of domains, the increase of the packet latency in SURF is much less than in our SB.

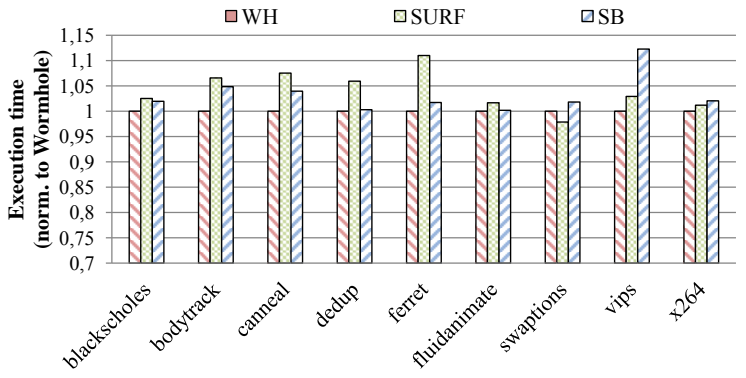


Figure 6.8: Application execution time.

6.6.2 Transfer of Multiple Class Packets

In a cache coherence protocol, there are multiple class packets, that should be separately stored in different VCs and transferred in different virtual networks [DT04]. It is necessary to guarantee that there is no protocol deadlock. However, as there are no VCs, the conventional bufferless NoCs does not support the transfer of multiple class packets. In contrast, by supporting confined-interference communication, our SB can easily support the transfer of different class packets for a cache coherence protocol.

To do so, we apply our SB on transferring different class packets for the MESI protocol [PP84], which needs two data virtual networks to transfer 5-flit packets and one control virtual network to transfer 1-flit packets. To separate the packets of the three different virtual networks, we set three domains and the packets of each virtual network are assigned/injected to one domain. There are 42 waves in our experiment. To continuously transfer 5-flit packets, the packets of one data virtual network are assigned to the waves $\{0, 1, 2, 3, 4, 15, 16, 17, 18, 19, 30, 31, 32, 33, 34\}$, and packets of the other data virtual network are assigned to the waves $\{7, 8, 9, 10, 11, 22, 23, 24, 25$ $26, 37, 38, 39, 40, 41\}$. In order to guarantee that the 5-flit packets in these two data virtual networks are transferred continuously, the head-flit of a 5-flit packet can be transferred only on the waves $\{0, 15, 30\}$ and $\{7, 22, 37\}$, respectively, and the rest of flits follow the head-flit. The 1-flit packets of the control virtual network are assigned to the rest of the waves, i.e., $\{5, 6, 12, 13, 14, 20, 21, 27, 28, 29, 35, 36\}$. As BLESS does not support the transfer of multiple class packets with various number of flits, BLESS is not considered in this experiment.

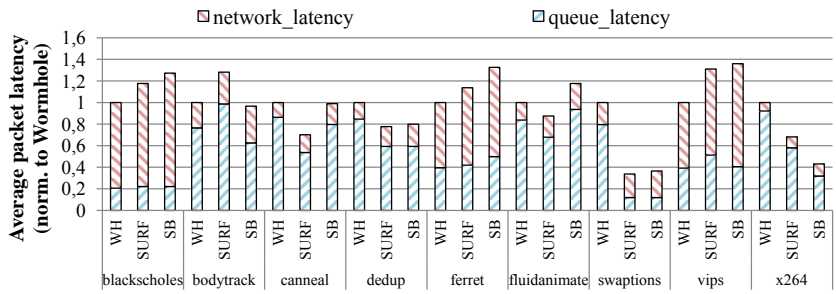


Figure 6.9: NoC packet latency.

Application Execution Time

Figure 6.8 shows the execution time of nine applications from the Parsec [BKSL08] benchmark suite, where the execution time is normalized to WH as the baseline (each input port has two data VCs and one control VC to build three virtual networks). Compared with WH, our SB causes an average of 3.23% performance penalty, which is less than the 4.13% performance penalty in SURF. In benchmarks dedup and fluidanimate, our SB achieves less than 1% performance penalty. In the vips benchmark, our SB has its largest performance penalty of 12.27%, whereas, for SURF, the largest performance penalty of 10.99% occurs for the ferret benchmark.

An interesting observation is that SURF achieves 2.14% performance improvement for the swaption benchmark. This improvement can be attributed to the acceleration of some packets' transmission. As shown in Figure 6.9, SURF has lower average packet latency than WH. Even though SB also has lower average packet latency than WH, our SB does not achieve such performance improvement. This is because the packet latency in SB is not significantly reduced as in SURF. Furthermore, most of the reduction of the packet latency in our SB and SURF comes from the control packets, which has a limited impact on the overall application performance. Only when the reduction of the packet latency is significant enough, SURF and SB can gain performance improvement.

NoC Packet Latency

Figure 6.9 shows the average packet latency for the nine applications. The average packet latency is broken down into the blocking time in the network interface (queue latency) and the transmission time in the NoC (network latency). Compared with WH, the average packet latency in SURF and our SB is significantly reduced in the benchmarks dedup, swaption, and x264, whereas, in benchmarks blackscholes, ferret, and

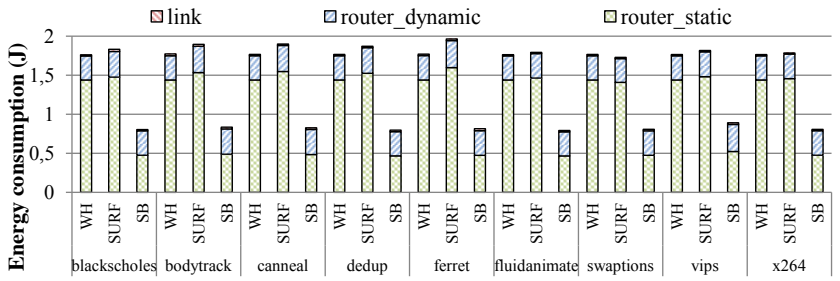


Figure 6.10: NoC energy consumption.

vips, the average packet latency increases. Most of the latency decrease comes from the queue latency reduction for the packets, whereas the latency increase is caused by the increase of the network latency. This is because based on the wave assignment in SURF and our SB, the network resources are reserved by different virtual networks to achieve confined-interference communication. However, since a domain can use the resources only in the time slots allocated to it, these resource reservation mechanisms cannot efficiently utilize the resource and undermine the efficiency of the networks, which causes the network latency increase.

By analyzing the result reports of GEM5, we found that most of the latency decrease comes from the queue latency reduction of the control (1-flit) packets, which is not essential to improve the overall application performance. As a consequence, even though the average packet latency in SURF and SB is reduced for some applications (see Figure 6.9), SURF and SB still cause performance penalty in the application execution time as shown in Figure 6.8.

NoC Energy Consumption

Figure 6.10 shows the NoC energy consumption for the nine applications. It is broken down into three parts, the static and dynamic energy consumption of routers and the total energy consumption of links. As can be seen in Figure 6.10, our SB consumes much less energy than the related approaches. Compared with WH, our SB reduces the total NoC energy consumption with 53.6% on average and compared with SURF, our SB reduces the total NoC energy consumption with 55.53% on average. This high energy reduction is achieved by eliminating the VCs for the N, S, W, and E input ports in a route for our SB to reduce the static energy consumption. In contrast, SURF has higher energy consumption than WH and our SB. This is because, with similar hardware structure of routers, WH and SURF have similar power consumption, but SURF needs more time to execute the applications. The link energy consumption is

negligible because we use the low workload mode in the Parsec benchmark suite.

6.7 Discussion

In this chapter, we propose the Surf-Bless routing approach and extend the router architecture to implement Surf-Bless in a distributed way. Based on our Surf-Bless approach, a conventional bufferless NoC is extended to support confined-interference communication. Furthermore, by taking advantage of the low power consumption of a bufferless NoC, our Surf-Bless routing approach achieves confined-interference communication with low energy consumption. The utilization of the wave pattern introduced in Section 6.5 is only suitable for a NoC with $N \times N$ 2D mesh topology. It is possible to extend our Surf-Bless to a $M \times N$ 2D mesh topology, but M and N must be the relation of integer multiple. It is also possible to extend our Surf-Bless to other mesh-based topologies, such as the 3D mesh topology and the concentrated mesh topology. However, it is not easy to extend our Surf-Bless to the tours topology, because the wire delay on a tours topology is not the same, which makes it difficult to construct the wave pattern. However, considering that NoCs with such topology are widely used in real chips, as shown in Table 1.1, our Surf-Bless routing approach is very promising and widely applicable.

Chapter 7

Summary and Conclusion

With low network latency, high bandwidth, good scalability, and reusability, a Network-on-Chip is a promising communication fabric for the future many-core systems. However, NoCs consume too much power in real chips, which constraints the utilization of NoCs in future large-scale many-core systems. Meanwhile, with more advanced semiconductor technologies, applied in chip manufacturing, the static power consumption takes a larger proportion of the total power consumption. Thus, in this thesis, we have focused our attention on reducing the static power consumption of NoCs in two directions: applying efficient power gating on NoCs to reduce the static power consumption and realizing a confined-interference communication on a simplified NoC infrastructure to achieve energy-efficient packet transmission.

By powering off the idle components/routers in a NoC, power gating is an effective way to reduce the power consumption of a NoC. However, when the power gating is applied on a NoC, the powered-off components/routers block the packet transmission and cause significant packet latency increase. This is because the powered-off components/routers need some clock cycles to be fully charged (i.e., to be powered-on). During the time period of charging powered-off routers, some packets cannot be transferred and have to be blocked until the powered-off routers are fully charged. As a consequence, applying power gating on a NoC causes significant packet latency increase. Furthermore, the power gating process (i.e., switching off/on the power of components/routers) itself consumes extra power. This implies that frequent power gating or power gating in a short time may cause more power consumption or inefficient power consumption reduction. Thus, to reduce the packet latency increase caused by power gating and achieve significant reduction of the power consumption in NoCs, we have proposed three novel power gating approaches: duty buffer based (DB-based) power gating, dynamic bypass (D-bypass) power gating, and express virtual channel based (EVC-based) power gating. These power gating approaches are

effective in reducing the power consumption of NoCs, but with different properties, they have different advantages. We summarize the properties of the DB-based power gating approach (DB_PG), the D-bypass power gating (D-bypass), and the EVC-based power gating approach (EVC_PG) in Figure 7.1. In Figure 7.1, the axes PL_l, PL_m, and PL_h represent the packet latency (PL) in a NoC under low traffic workloads (l), medium traffic workloads (m), and high traffic workloads (h), respectively. The axes PC_l, PC_m, and PC_h represent the power consumption (PC) of a NoC under low traffic workloads (l), medium traffic workloads (h), and high traffic workloads (h), respectively. For example, the PL_m axis crosses the block edges of DB_PG, D-bypass, and EVC_PG at three points, respectively. These points represent the packet latency (normalized to the same baseline) of DB_PG, D-bypass, and EVC_PG under medium traffic workloads. Thus, according to Figure 7.1, under medium traffic workloads, DB_PG has the highest packet latency among our three approaches, whereas EVC_PG has the lowest packet latency. Based on the different properties of our power gating approaches, shown in Figure 7.1, we draw the following conclusions:

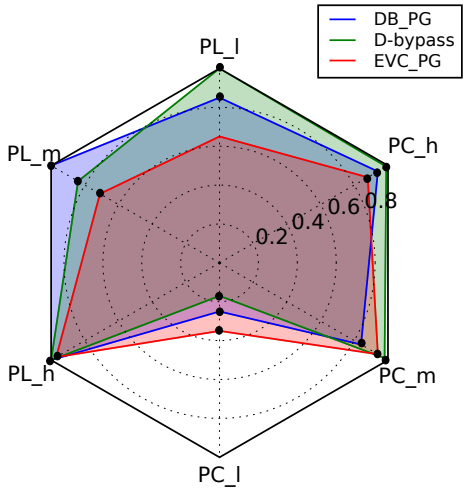


Figure 7.1: Packet latency (PL) and power consumption (PC) at low traffic workloads (l), medium traffic workloads (m), and high traffic workloads (h).

- **Our DB-based power gating approach is effective in reducing the power consumption of a NoC in a wide range of traffic workloads, but at medium traffic workloads, it has the highest packet latency among our three power gating approaches.** This is because, our DB-based power gating is a fine-grained power gating approach, in which each input port of a router can be

separately powered-off. In this way, our DB-based power gating approach can fully utilize the idle time of each input port in a router to reduce the static power consumption. Thus, at different traffic workloads, our DB-based power gating approach achieves significant reduction of the power consumption in a NoC. Furthermore, taking advantage of our novel duty buffer (BD) structure to replace the powered-off input port to transfer packets, our DB-based power gating approach achieves lower packet latency then D-bypass at low traffic workloads in as shown Figure 7.1. However, being a fine-grained power gating approach, our DB-based power gating approach needs to separately switch the power of each input port in a router. At medium traffic workloads, packets experience many power gating processes. As a consequence, our DB-based power gating approach has the highest packet latency among our three approaches at medium traffic workloads.

- **At low traffic workloads, our D-bypass power gating is the most power-efficient approach among our three approaches, and it is effective in reducing the power consumption of a NoC only at low traffic workloads. However, at low traffic workloads, our D-bypass power gating has the highest packet latency among our approaches.** This is because, in our D-bypass power gating approach, we add one special hardware bypass structure in each router. When a router is powered-off, only this special hardware bypass structure is kept powered-on. Compared with the DB-based power gating approach and the EVC-based power gating approach, our D-bypass power gating approach can power off more components in a router to reduce the static power consumption. Thus, at low traffic workloads, in which most of the routers are idle and can be powered-off, our D-bypass power gating approach consumes the least power among our three approaches. Furthermore, the special hardware bypass structure in each router makes it possible for packets to bypass powered-off routers. In this way, our D-bypass power gating approach can efficiently reduce the extra power consumption caused by power gating. However, being a course-grained power gating approach, our D-bypass power gating approach cannot fully utilize the idle time of each component in a router. When the traffic workload increases, most of the routers in a NoC become busy and cannot be powered off to reduce the static power consumption. As a consequence, our D-bypass power gating approach is effective only at low traffic workloads. In terms of the packet latency, as packets can bypass powered-off routers in our D-bypass power gating approach, the packet latency increase caused by power gating is reduced. However, limited by the low transmission capacity of the special hardware bypass structure in powered-off routers, our D-bypass power gating approach still causes significant increase of the packet latency. As a con-

sequence, our D-bypass power gating approach has the highest packet latency among our three approaches at low traffic workloads.

- **Our EVC-based power gating approach achieves the lowest packet latency among our three approaches at different traffic workloads. Furthermore, it is also the most effective approach in reducing the power consumption at high traffic workloads.** This is because, in the EVC-based power gating approach, we pre-define multiple virtual bypass paths between different routers. Packets can take these virtual bypass paths to bypass intermediate routers that can be powered-on or powered-off. Furthermore, compared with the D-bypass power gating approach, the pre-defined virtual bypass paths in our EVC-based power gating approach are much more efficient to allow packets to bypass the powered-on/powered-off routers. Therefore, our EVC-based power gating approach achieves the lowest packet latency among our three power gating approaches. In addition, packets can bypass not only powered-off routers but also they can bypass powered-on routers as well. Thus, even at high traffic workloads, our EVC-based power gating approach still can reduce the power consumption by allowing packets to bypass the powered-on routers.

A confined-interference communication in a NoC-based System-on-Chip is a useful quality-of-service. In confined-interference communication, the packets of different applications are grouped into different domains and packet interference can occur only in the same domain, whereas there is no packet interference between domains. By supporting a confined-interference communication, NoCs can support composability to facilitate the temporal verification of (hard) real-time applications. However, realizing a confined-interference communication on a conventional (virtual channel/buffer based) NoC requires a large number of virtual channels, which causes high power consumption. Therefore, there is an urgent need for realizing a confined-interference communication on a more power-efficient NoC architecture. Bufferless NoCs have simplified NoC architectures. By eliminating virtual channels/buffers in routers, bufferless NoCs consume much less power than conventional NoCs. However, as there are no buffers in bufferless NoCs to temporarily store packets, packets have to keep moving, which makes it more difficult to control the interference between packets. As a consequence, current bufferless NoCs do not support a confined-interference communication.

To overcome this issue, we have proposed a novel routing approach, called Surfing on a Bufferless NoC (Surf-Bless). **Based on our Surf-Bless routing approach, it becomes possible for bufferless NoC to support a confined-interference communication. Furthermore, our Surf-Bless routing approach is much more power/energy-efficient than related approaches.** This is because, our Surf-Bless approach is based

on a specific assignment and scheduling of the resources in a bufferless NoC. This specific assignment and scheduling can be visualized as multiple “waves" which move in space and time over the NoC in a specially designed repetitive pattern. The specially designed repetitive pattern for the waves guarantees that packets “surfing” on a wave can keep moving, which is essential to correctly use a bufferless NoC to transfer packets. This is because, in a bufferless NoC, there are no buffers and packets have to keep moving. Furthermore, the specially designed repetitive pattern also guarantees that there is no interference between different waves. Thus, by assigning different domains on different waves, there is no interference between domains and a confined-interference communication is achieved. In this way, we realize confined-interference communication on a bufferless NoC infrastructure. Furthermore, as the routers in our Surf-Bless approach do not have virtual channels/buffers, our Surf-Bless routing consumes much less power/energy than related approaches.

Bibliography

[AABC18] Fawaz Alazemi, Arash Azizimazreah, Bella Bose, and Lizhong Chen. Routerless network-on-chip. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 492–503. IEEE, 2018.

[AKPJ09] Niket Agarwal, Tushar Krishna, Li-Shiuan Peh, and Niraj K Jha. Garnet: A detailed on-chip network model inside a full-system simulator. In *2009 IEEE international symposium on performance analysis of systems and software*, pages 33–42. IEEE, 2009.

[BBB+11] Nathan Binkert, Bradford Beckmann, Gabriel Black, Steven K Reinhardt, Ali Saidi, Arkaprava Basu, Joel Hestness, Derek R Hower, Tushar Krishna, Somayeh Sardashti, et al. The gem5 simulator. *ACM SIGARCH Computer Architecture News*, 39(2):1–7, 2011.

[BD14] James Balfour and William J Dally. Design tradeoffs for tiled cmp on-chip networks. In *ACM International Conference on Supercomputing 25th Anniversary Volume*, pages 390–401. ACM, 2014.

[BDM02] Luca Benini and Giovanni De Micheli. Networks on chips: A new soc paradigm. *computer*, 35(1):70–78, 2002.

[BEA+08] Shane Bell, Bruce Edwards, John Amann, Rich Conlin, Kevin Joyce, Vince Leung, John MacKay, Mike Reif, Liewei Bao, John Brown, et al. Tile64-processor: A 64-core soc with mesh interconnect. In *2008 IEEE International Solid-State Circuits Conference-Digest of Technical Papers*, pages 88–598. IEEE, 2008.

[BHW+17] Rahul Boyapati, Jiayi Huang, Ningyuan Wang, Kyung Hoon Kim, Ki Hwan Yum, and Eun Jung Kim. Fly-over: A light-weight distributed power-gating mechanism for energy-efficient networks-on-chip. In *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 708–717. IEEE, 2017.

[BJ14] Mario Badr and Natalie Enright Jerger. Synfull: synthetic traffic models capturing cache coherent behaviour. In *2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, pages 109–120. IEEE, 2014.

[BJS+14] Haseeb Bokhari, Haris Javaid, Muhammad Shafique, Jörg Henkel, and Sri Parameswaran. darknoc: Designing energy-efficient network-on-chip with multi-vt cells for dark silicon. In *Proceedings of the 51st Annual Design Automation Conference*, pages 1–6. ACM, 2014.

[BKA10] Ali Bakhoda, John Kim, and Tor M Aamodt. Throughput-effective on-chip networks for manycore accelerators. In *Proceedings of the 2010 43rd annual IEEE/ACM international symposium on microarchitecture*, pages 421–432. IEEE Computer Society, 2010.

[BKSL08] Christian Bienia, Sanjeev Kumar, Jaswinder Pal Singh, and Kai Li. The parsec benchmark suite: Characterization and architectural implications. In *Proceedings of the 17th international conference on Parallel architectures and compilation techniques*, pages 72–81. ACM, 2008.

[Boh07] Mark Bohr. A 30 year retrospective on dennard's mosfet scaling paper. *IEEE Solid-State Circuits Society Newsletter*, 12(1):11–13, 2007.

[Bor07] Shekhar Borkar. Thousand core chipsa technology perspective. In *2007 44th ACM/IEEE Design Automation Conference*, pages 746–749. IEEE, 2007.

[BS00] J Adam Butts and Gurindar S Sohi. A static power model for architects. In *Proceedings 33rd Annual IEEE/ACM International Symposium on Microarchitecture. MICRO-33 2000*, pages 191–201. IEEE, 2000.

[CJ16] Xianmin Chen and Niraj K Jha. Reducing wire and energy overheads of the smart noc using a setup request network. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(10):3013–3026, 2016.

[CP12] Lizhong Chen and Timothy M Pinkston. Nord: Node-router decoupling for effective power-gating of on-chip routers. In *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 270–281. IEEE Computer Society, 2012.

[CPK+13] Chia-Hsin Owen Chen, Sunghyun Park, Tushar Krishna, Suvinay Subramanian, Anantha P Chandrakasan, and Li-Shiuan Peh. Smart: a single-cycle reconfigurable noc for soc applications. In *2013 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 338–343. IEEE, 2013.

[CZPP15] Lizhong Chen, Di Zhu, Massoud Pedram, and Timothy M Pinkston. Power punch: Towards non-blocking power-gating of noc routers. In *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, pages 378–389. IEEE, 2015.

[CZPP16] Lizhong Chen, Di Zhu, Massoud Pedram, and Timothy M Pinkston. Simulation of noc power-gating: Requirements, optimizations, and the agate simulator. *Journal of Parallel and Distributed Computing*, 95:69–78, 2016.

[CZZ+15] Hsiang-Yun Cheng, Jia Zhan, Jishen Zhao, Yuan Xie, Jack Sampson, and Mary Jane Irwin. Core vs. uncore: the heart of darkness. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2015.

[DCS+14] Bhavya K Daya, Chia-Hsin Owen Chen, Suvinay Subramanian, Woo-Cheol Kwon, Sunghyun Park, Tushar Krishna, Jim Holt, Anantha P Chandrakasan, and Li-Shiuan Peh. Scorpio: a 36-core research chip demonstrating snoopy coherence on a scalable mesh noc with in-network ordering. In *2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, pages 25–36. IEEE, 2014.

[DGR+74] Robert H Dennard, Fritz H Gaensslen, V Leo Rideout, Ernest Bassous, and Andre R LeBlanc. Design of ion-implanted mosfet's with very small physical dimensions. *IEEE Journal of Solid-State Circuits*, 9(5):256–268, 1974.

[DMMD09] Reetuparna Das, Onur Mutlu, Thomas Moscibroda, and Chita R Das. Application-aware prioritization mechanisms for on-chip networks. In *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, pages 280–291. ACM, 2009.

[DNSD13] Reetuparna Das, Satish Narayanasamy, Sudhir K Satpathy, and Ronald G Dreslinski. Catnap: energy proportional multiple network-on-chip. In *ACM SIGARCH Computer Architecture News*, pages 320–331. ACM, 2013.

[DT01] William J Dally and Brian Towles. Route packets, not wires: on-chip interconnection networks. In *Proceedings of the 38th annual Design Automation Conference*, pages 684–689. Acm, 2001.

[DT04] William James Dally and Brian Patrick Towles. *Principles and practices of interconnection networks*. Elsevier, 2004.

[EBA+11] Hadi Esmaeilzadeh, Emily Blem, Renee St Amant, Karthikeyan Sankaralingam, and Doug Burger. Dark silicon and the end of multicore scaling. In *2011 38th Annual international symposium on computer architecture (ISCA)*, pages 365–376. IEEE, 2011.

[FCM11] Chris Fallin, Chris Craik, and Onur Mutlu. Chipper: A low-complexity bufferless deflection router. In *2011 IEEE 17th International Symposium on High Performance Computer Architecture*, pages 144–155. IEEE, 2011.

[FDC+09] David Fick, Andrew DeOrio, Gregory Chen, Valeria Bertacco, Dennis Sylvester, and David Blaauw. A highly resilient routing algorithm for fault-tolerant nocs. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 21–26. European Design and Automation Association, 2009.

[FKM+02] Krisztián Flautner, Nam Sung Kim, Steve Martin, David Blaauw, and Trevor Mudge. Drowsy caches: simple techniques for reducing leakage power. In *ACM SIGARCH Computer Architecture News*, pages 148–157. IEEE Computer Society, 2002.

[FLY+16] Haohuan Fu, Junfeng Liao, Jinzhe Yang, Lanning Wang, Zhenya Song, Xiaomeng Huang, Chao Yang, Wei Xue, Fangfang Liu, Fangli Qiao, et al. The sunway taihulight supercomputer: system and applications. *Science China Information Sciences*, 59(7):072001, 2016.

[FTKH16] Hossein Farrokhbakht, Mohammadkazem Taram, Behnam Khaleghi, and Shaahin Hessabi. Toot: an efficient and scalable power-gating method for noc routers. In *2016 Tenth IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*, pages 1–8. IEEE, 2016.

[FYNM11] Chris Fallin, Xiangyao Yu, Gregory Nazario, and Onur Mutlu. A high-performance hierarchical ring on-chip interconnect with low-cost routers. *Computer Architecture Lab, Carnegie Mellon Univ, Tech. Rep*, 7:2011, 2011.

[Gal97] Mike Galles. Spider: A high-speed network interconnect. *IEEE Micro*, 17(1):34–39, 1997.

[GDR05] Kees Goossens, John Dielissen, and Andrei Radulescu. Æthereal network on chip: concepts, architectures, and implementations. *IEEE Design & Test of Computers*, 22(5):414–421, 2005.

[GH10] Kees Goossens and Andreas Hansson. The aethereal network on chip after ten years: Goals, evolution, lessons, and future. In *Design Automation Conference*, pages 306–311. IEEE, 2010.

[GHKM09] Boris Grot, Joel Hestness, Stephen W Keckler, and Onur Mutlu. Express cube topologies for on-chip interconnects. In *2009 IEEE 15th International Symposium on High Performance Computer Architecture*, pages 163–174. IEEE, 2009.

[GHKM11] Boris Grot, Joel Hestness, Stephen W Keckler, and Onur Mutlu. Kilo-noc: a heterogeneous network-on-chip architecture for scalability and service guarantees. In *ACM SIGARCH Computer Architecture News*, pages 401–412. ACM, 2011.

[GKS+07] Paul Gratz, Changkyu Kim, Karthikeyan Sankaralingam, Heather Hanson, Premkishore Shivakumar, Stephen W Keckler, and Doug Burger. On-chip interconnection networks of the trips chip. *IEEE Micro*, 27(5):41–50, 2007.

[GN92] Christopher J Glass and Lionel M Ni. The turn model for adaptive routing. *ACM SIGARCH Computer Architecture News*, 20(2):278–287, 1992.

[HDH+10] Jason Howard, Saurabh Dighe, Yatin Hoskote, Sriram Vangal, David Finan, Gregory Ruhl, David Jenkins, Howard Wilson, Nitin Borkar, Gerhard Schrom, et al. A 48-core ia-32 message-passing processor with dvfs in 45nm cmos. In *2010 IEEE International Solid-State Circuits Conference-(ISSCC)*, pages 108–109. IEEE, 2010.

[HGBH09] Andreas Hansson, Kees Goossens, Marco Bekooij, and Jos Huisken. Compsoc: A template for composable and predictable multi-processor system on chips. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 14(1):2, 2009.

[HGR07] Andreas Hansson, Kees Goossens, and Andrei Rădulescu. Avoiding message-dependent deadlock in network-based systems on chip. *VLSI design*, 2007, 2007.

[HHS+00] Lance Hammond, Benedict A Hubbert, Michael Siu, Manohar K Prabhu, Michael Chen, and K Olukolun. The stanford hydra cmp. *IEEE micro*, 20(2):71–84, 2000.

[HJK+00] Ahmed Hemani, Axel Jantsch, Shashi Kumar, Adam Postula, Johnny Oberg, Mikael Millberg, and Dan Lindqvist. Network on chip: An architecture for billion transistor era. In *Proceeding of the IEEE NorChip Conference*, volume 31, page 11, 2000.

[HSG09] Andreas Hansson, Mahesh Subburaman, and Kees Goossens. aelite: A flit-synchronous network on chip with composable and predictable services. In *Proceedings of the conference on design, automation and test in Europe*, pages 250–255. European Design and Automation Association, 2009.

[HVS+07] Yatin Hoskote, Sriram Vangal, Arvind Singh, Nitin Borkar, and Shekhar Borkar. A 5-ghz mesh interconnect for a teraflops processor. *IEEE Micro*, 27(5):51–61, 2007.

[HY13] Syed Minhaj Hassan and Sudhakar Yalamanchili. Centralized buffer router: A low latency, low power router for high radix nocs. In *2013 Seventh IEEE/ACM International Symposium on Networks-on-Chip (NoCS)*, pages 1–8. IEEE, 2013.

[JBB+13] Nan Jiang, James Balfour, Daniel U Becker, Brian Towles, William J Dally, George Michelogiannakis, and John Kim. A detailed and flexible cycle-accurate network-on-chip simulator. In *Performance Analysis of Systems and Software (ISPASS), 2013 IEEE International Symposium on*, pages 86–96. IEEE, 2013.

[KBD07] John Kim, James Balfour, and William Dally. Flattened butterfly topology for on-chip networks. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 172–182. IEEE Computer Society, 2007.

[KCKP13] Tushar Krishna, Chia-Hsin Owen Chen, Woo Cheol Kwon, and Li-Shiuan Peh. Breaking the on-chip latency barrier using smart. In *2013 IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*, pages 378–389. IEEE, 2013.

[KKC+08] Tushar Krishna, Amit Kumar, Patrick Chiang, Mattan Erez, and Li-Shiuan Peh. Noc with near-ideal express virtual channels using global-line communication. In *2008 16th IEEE Symposium on High Performance Interconnects*, pages 11–20. IEEE, 2008.

[KKS+07] Amit Kumary, Partha Kunduz, Arvind P Singhx, Li-Shiuan Pehy, and Niraj K Jhay. A 4.6 tbits/s 3.6 ghz single-cycle noc router with a novel switch allocator in 65nm cmos. In *2007 25th International Conference on Computer Design*, pages 63–70. IEEE, 2007.

[KKY11] Gwangsun Kim, John Kim, and Sungjoo Yoo. Flexibuffer: Reducing leakage power in on-chip network routers. In *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*, pages 936–941. IEEE, 2011.

[KPKJ07] Amit Kumar, Li-Shiuan Peh, Partha Kundu, and Niraj K Jha. Express virtual channels: towards the ideal interconnection fabric. In *ACM SIGARCH Computer Architecture News*, pages 150–161. ACM, 2007.

[KTMW03] Jason Sungtae Kim, Michael Bedford Taylor, Jason Miller, and David Wentzlaff. Energy characterization of a tiled architecture processor with on-chip networks. In *Proceedings of the 2003 International Symposium on Low Power Electronics and Design, 2003. ISLPED'03.*, pages 424–427. IEEE, 2003.

[LCL+16] Shaoli Liu, Tianshi Chen, Ling Li, Xiaoxue Feng, Zhiwei Xu, Haibo Chen, Fred Chong, and Yunji Chen. Imr: High-performance low-cost multi-ring nocs. *IEEE Transactions on Parallel and Distributed Systems*, 27(6):1700–1712, 2016.

[LSMJ16] Zimo Li, Joshua San Miguel, and Natalie Enright Jerger. The runahead network-on-chip. In *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 333–344. IEEE, 2016.

[LY18] Zhonghai Lu and Yuan Yao. Thread voting dvfs for manycore nocs. *IEEE Transactions on Computers*, 67(10):1506–1524, 2018.

[MJW12] Sheng Ma, Natalie Enright Jerger, and Zhiying Wang. Supporting efficient collective communication in nocs. In *IEEE International Symposium on High-Performance Comp Architecture*, pages 1–12. IEEE, 2012.

[MKAY09] Hiroki Matsutani, Michihiro Koibuchi, Hideharu Amano, and Tsutomu Yoshinaga. Prediction router: Yet another low latency on-chip router architecture. In *2009 IEEE 15th International Symposium on High Performance Computer Architecture*, pages 367–378. IEEE, 2009.

[MKI+10] Hiroki Matsutani, Michihiro Koibuchi, Daisuke Ikebuchi, Kimiyoshi Usami, Hiroshi Nakamura, and Hideharu Amano. Ultra fine-grained run-time power gating of on-chip routers for cmps. In *2010 Fourth ACM/IEEE International Symposium on Networks-on-Chip*, pages 61–68. IEEE, 2010.

[MKWA08] Hiroki Matsutani, Michihiro Koibuchi, Daihan Wang, and Hideharu Amano. Run-time power gating of on-chip routers using look-ahead routing. In *Proceedings of the 2008 Asia and South Pacific Design Automation Conference*, pages 55–60. IEEE Computer Society Press, 2008.

[MM09] Thomas Moscibroda and Onur Mutlu. A case for bufferless routing in on-chip networks. *ACM SIGARCH Computer Architecture News*, 37(3):196–207, 2009.

[MNTJ04] Mikael Millberg, Erland Nilsson, Rikard Thid, and Axel Jantsch. Guaranteed bandwidth using looped containers in temporally disjoint networks within the nostrum network on chip. In *Proceedings Design, Automation and Test in Europe Conference and Exhibition*, volume 2, pages 890–895. IEEE, 2004.

[MPK07] George Michelogiannakis, Dionisios Pnevmatikatos, and Manolis Katevenis. Approaching ideal noc latency with pre-configured routes. In *First International Symposium on Networks-on-Chip (NOCS'07)*, pages 153–162. IEEE, 2007.

[OM06] Umit Y Ogras and Radu Marculescu. Prediction-based flow control for network-on-chip traffic. In *Proceedings of the 43rd annual design automation conference*, pages 839–844. ACM, 2006.

[P+16] Anastasios Psarras et al. Phasenoc: Versatile network traffic isolation through tdm-scheduled virtual channels. *IEEE TCAD*, 2016.

[PD01] L-S Peh and William J Dally. A delay model and speculative architecture for pipelined routers. In *Proceedings HPCA Seventh International Symposium on High-Performance Computer Architecture*, pages 255–266. IEEE, 2001.

[PP84] Mark S Papamarcos and Janak H Patel. A low-overhead coherence solution for multiprocessors with private cache memories. *ACM SIGARCH Computer Architecture News*, 12(3):348–354, 1984.

[SCK+12] Chen Sun, Chia-Hsin Owen Chen, George Kurian, Lan Wei, Jason Miller, Anant Agarwal, Li-Shiuan Peh, and Vladimir Stojanovic. Dsent-a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling. In *2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip*, pages 201–210. IEEE, 2012.

[Sil] Silvaco. Nangate 45nm library. `http://www.nangate.com/`.

[SJJ+11] Praveen Salihundam, Shailendra Jain, Tiju Jacob, Shasi Kumar, Vasantha Erraguntla, Yatin Hoskote, Sriram Vangal, Gregory Ruhl, and Nitin Borkar. A 2 tb/s 6×4 mesh network for a single-chip cloud computer with dvfs in 45 nm cmos. *IEEE Journal of Solid-State Circuits*, 46(4):757–766, 2011.

[SMG14] Radu Andrei Stefan, Anca Molnos, and Kees Goossens. daelite: A tdm noc supporting qos, multicast, and fast connection set-up. *IEEE Transactions on Computers*, 63(3):583–594, 2014.

[TKM+02] Michael Bedford Taylor, Jason Kim, Jason Miller, David Wentzlaff, Fae Ghodrat, Ben Greenwald, Henry Hoffman, Paul Johnson, Jae-Wook Lee, Walter Lee, et al. The raw microprocessor: A computational fabric for software circuits and general-purpose programs. *IEEE micro*, 22(2):25–35, 2002.

[Val82] Leslie G. Valiant. A scheme for fast parallel communication. *SIAM journal on computing*, 11(2):350–361, 1982.

[VB81] Leslie G Valiant and Gordon J Brebner. Universal schemes for parallel communication. In *Proceedings of the thirteenth annual ACM symposium on Theory of computing*, pages 263–277. ACM, 1981.

[vdBCGB07] Jan Willem van den Brand, Calin Ciordas, Kees Goossens, and Twan Basten. Congestion-controlled best-effort communication for networks-on-chip. In *Proceedings of the conference on Design, automation and test in Europe*, pages 948–953. EDA Consortium, 2007.

[WDLW17] Ji Wu, Dezun Dong, Xiangke Liao, and Li Wang. Energy-efficient noc with multi-granularity power optimization. *The journal of Supercomputing*, 73(4):1654–1671, 2017.

[WGO+13] Hassan MG Wassel, Ying Gao, Jason K Oberg, Ted Huffmire, Ryan Kastner, Frederic T Chong, and Timothy Sherwood. Surfnoc: a low latency and provably non-interfering approach to secure networks-on-chip. In *ACM SIGARCH Computer Architecture News*, pages 583–594. ACM, 2013.

[WNM+19a] Peng Wang, Sobhan Niknam, Sheng Ma, Zhiying Wang, and Todor Stefanov. A dynamic bypass approach to realize power efficient network-on-chip. In *Proceedings of the 21st IEEE International Conference on High Performance Computing and Communications (HPCC-2019)*, 2019.

[WNM+19b] Peng Wang, Sobhan Niknam, Sheng Ma, Zhiying Wang, and Todor Stefanov. Evc-based power gating approach to achieve low-power and high performance noc. In *Proceedings of Euromicro Conference on Digital System Design*, 2019.

[WNM+19c] Peng Wang, Sobhan Niknam, Sheng Ma, Zhiying Wang, and Todor Stefanov. Surf-bless: A confined-interference routing for energy-efficient communication in nocs. In *Proceedings of the 56th Annual Design Automation Conference 2019*, page 50. ACM, 2019.

[WNWS17] Peng Wang, Sobhan Niknam, Zhiying Wang, and Todor Stefanov. A novel approach to reduce packet latency increase caused by power gating in network-on-chip. In *2017 Eleventh IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*, pages 1–8. IEEE, 2017.

[YL16] Yuan Yao and Zhonghai Lu. Dvfs for nocs in cmps: A thread voting approach. In *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 309–320. IEEE, 2016.

[ZCPP15] Di Zhu, Lizhong Chen, Timothy M Pinkston, and Massoud Pedram. Tapp: Temperature-aware application mapping for noc-based many-core processors. In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, pages 1241–1244. EDA Consortium, 2015.

[ZL18] Hao Zheng and Ahmed Louri. Ez-pass: An energy & performance-efficient power-gating router architecture for scalable nocs. *IEEE Computer Architecture Letters*, 17(1):88–91, 2018.

[ZML+16] Xia Zhao, Sheng Ma, Chen Li, Lieven Eeckhout, and Zhiying Wang. A heterogeneous low-cost and low-latency ring-chain network for gpgpus. In *2016 IEEE 34th International Conference on Computer Design (ICCD)*, pages 472–479. Ieee, 2016.

[ZOG+15] Jia Zhan, Jin Ouyang, Fen Ge, Jishen Zhao, and Yuan Xie. Dimnoc: A dim silicon approach towards power-efficient on-chip network. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2015.

List of Publications

First Author Publications

1. **Peng Wang**, Sobhan Niknam, Sheng Ma, Zhiying Wang, and Todor Stefanov, "Surf-Bless: A Confined-interference Routing for Energy-Efficient Communication in NoCs", In Proc. "56th ACM/IEEE Int. Design Automation Conference (DAC'19)", pp. 50, Las Vegas, NV, USA, June 2-6, 2019, **(Winner of 2019 HiPEAC Paper Award)**.

2. **Peng Wang**, Sobhan Niknam, Sheng Ma, Zhiying Wang, and Todor Stefanov, "A Dynamic Bypass Approach to Realize Power Efficient Network-on-Chip", In Proc. "21st IEEE International Conference on High Performance Computing and Communications (HPCC-2019)", Zhangjiajie, Hunan, China, August 10-12, 2019.

3. **Peng Wang**, Sobhan Niknam, Sheng Ma, Zhiying Wang, and Todor Stefanov, "EVC-based Power Gating Approach to Achieve Low-power and High Performance NoC", In Proc. "Euromicro Conference on Digital System Design (DSD)", Chalkidiki, Greece, August 28-30, 2019.

4. **Peng Wang**, Sobhan Niknam, Zhiying Wang, and Todor Stefanov, "A Novel Approach to Reduce Packet Latency Increase caused by Power Gating in Network-on-Chip", In Proc. "11th ACM/IEEE International Symposium on Networks-on-Chip (NOCS'17)", pp. 1-8 , Seoul, South Korea, Oct. 19-20, 2017.

Co-author Publications

1. Sobhan Niknam, **Peng Wang**, and Todor Stefanov, "Resource Optimization for Real-Time Streaming Applications using Task Replication", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), vol. 37, No. 11, pp. 2755-2767, Nov 2018.

2. Sobhan Niknam, **Peng Wang**, and Todor Stefanov, "Hard Real-Time Scheduling of Streaming Applications Modeled as Cyclic CSDF Graphs", In Proc. "22nd Int. Conf. Design, Automation and Test in Europe (DATE'19)", pp. 1528-1533, Florence, Italy, Mar. 25-29, 2019.

3. Di Liu, Jelena Spasic, **Peng Wang**, and Todor Stefanov, "Energy-Efficient Scheduling of Real-Time Tasks on Heterogeneous Multicores Using Task Splitting", In Proc. "22nd IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'16)", pp. 149-158, Daegu, South Korea, Aug. 17-19, 2016.

4. Fuchs, Christian M, Murillo, Nadia M, Plaat, Aske, van der Kouwe, Erik, **Peng Wang**, "Towards affordable fault-tolerant nanosatellite computing with commodity hardware", In Proc. "27th IEEE Asian Test Symposium (ATS'2018)", pp. 127-132, Heifei, China, Oct. 15-18, 2018.

Samenvatting

In multi/many-core System-on-Chips (SoCs) schaalt prestatie vrijwel lineair met het aantal rekenkernen. Om hogere prestaties te bereiken moeten meer rekenkernen in many-core SoCs worden geïntegreerd, waardoor communicatie tussen de kernen een bottleneck wordt voor verdere prestatieverhoging. Een Network-on-Chip (NoC), met lage netwerk vertraging, hoge bandbreedte, goede schaalbaarheid en herbruikbaarheid, lijkt een veelbelovend communicatiemedium te zijn voor de vele SoCs. Echter, NoCs consumeren in de praktijk te veel stroom, waardoor de toepasbaarheid van NoCs in toekomstige grootschalige many-core SoCs gelimiteerd wordt. Omdat steeds geavanceerdere halfgeleidertechnologieën worden gebruikt in chipfabricage, neemt het statisch stroomgebruik van een NoC het merendeel in van het totale stroomgebruik. Derhalve hebben wij ons in deze thesis gericht op het reduceren van het statische stroomgebruik van NoCs op twee manieren: door op efficiënte wijze stroomregulatie toe te passen om statisch stroomgebruik te minderen en door communicatie met beperkte interferentie toe te passen op een gesimplificeerde NoC structuur om energiezuinige pakketverzending te bewerkstelligen.

Door onbenutte componenten/routers in een NoC uit te zetten is stroomregulatie een efficiënte methode om het stroomgebruik van een NoC te reduceren. Echter, wanneer stroomregulatie op een NoC wordt toegepast, blokkeren de uitgeschakelde componenten/routers de pakketverzending en zorgen daarmee voor significante toename in pakketvertraging. Verder kost stroomregulatie (het in- en uitschakelen van componenten/routers) zelf ook extra stroom, waardoor stroomgebruik kan toenemen, of de reductie ervan inefficiënt is. Om het toenemen van pakketvertraging door stroomregulatie te reduceren en een significante vermindering in het stroomgebruik van NoCs te bereiken, stellen wij drie nieuwe stroomregulatie aanpakken voor: taken buffer gebaseerde (TB-gebaseerde) stroomregulatie, dynamische omzeiling (D-omzeiling) stroomregulatie en direct virtueel kanaal gebaseerde (DVK-gebaseerde) stroomregulatie. Deze stroomregulatie aanpakken zijn effectief in het verminderen van de stroomconsumptie van NoCs. Verder hebben de aanpakken verschillende eigenschappen, ieder met hun eigen voordelen. Door zeer precisie stroomregulatie kan onze TB-gebaseerde aanpak volledig gebruik maken van de tijd dat iedere invoerpoort

in een router onbenut is om de statische stroomconsumptie te reduceren. Hierdoor is onze TB-gebaseerde aanpak effectief in het reduceren van stroomgebruik van een NoC in een breder scala aan werkdrukte. De D-omzeiling en de DVK-gebaseerde aanpakken zorgen ervoor dat pakketten uitgeschakelde routers kunnen omzeilen. Hierdoor zijn deze aanpakken effectiever in het limiteren van pakketvertraging veroorzaakt door stroomregulatie en hebben daardoor minder prestatiekosten. Daarnaast staat de DVK-gebaseerde aanpak toe dat pakketten ook ingeschakelde routers kunnen omzeilen om tevens dynamische stroomconsumptie te verminderen. Hierdoor gebruikt de DVK-gebaseerde aanpak minder stroom bij zware werkdruk dan de D-omzeiling aanpak.

Communicatie met beperkte interferentie op en NoC-gebaseerde SoC is een nuttige quality-of-service. Bij communicatie met beperkte interferentie worden de pakketten van verschillende applicaties gegroepeerd in verschillende domeinen, waarbij gevolgtrekking uitsluitend binnen hetzelfde domein kan plaatsvinden, niet tussen domeinen. Door communicatie met beperkte interferentie te ondersteunen hebben NoCs samenstelbaarheid, waardoor temporele verificatie van (harde) real-time applicaties wordt gefaciliteerd. Echter, het gebruik van communicatie met beperkte interferentie vraagt op een conventionele NoC een groot aantal virtuele kanalen, wat een hoog stroomgebruik met zich meebrengt. Hierdoor is er een prangende vraag naar communicatie met beperkte interferentie op een energiezuinige NoC architectuur. Bufferloze NoCs hebben een gesimplificeerde NoC architectuur. Door virtuele kanalen/buffers in de routers te elimineren, hebben bufferloze NoCs een veel lager stroomgebruik dan conventionele NoCs. Er zitten echter geen buffers in bufferloze NoCs om tijdelijk pakketten op te slaan, waardoor deze moeten blijven bewegen, wat de gevolgtrekking tussen pakketten moeilijker maakt. Als gevolg hiervan ondersteunen huidige bufferloze NoCs geen communicatie met beperkte interferentie. Om dit probleem op te lossen hebben wij een nieuwe routebepaling voorgesteld, genaamd surfen op bufferloze NoC (Surf-Bless). Op basis van onze Surf-Bless routebepaling wordt het mogelijk voor bufferloze NoCs om communicatie met beperkte interferentie te ondersteunen. Bovendien is onze Surf-Bless routebepaling, gebruik makende van de lage stroomconsumptie van de bufferloze NoC, veel kracht/energie-efficiënter dan de conventionele NoC.

Acknowledgements

At the end of this thesis, I would like to thank for whom provide numerous support and help in my academic career. Without your support and help, it would not have been possible for me to approach the end of Ph.D journey.

First of all, I'd like to thank China Scholarship Council for sponsoring me to pursue my Ph.D in Leiden University.

At Nation University of Defense Technology (NUDT), I were very lucky to join in the group of State Key Laboratory of High Performance Computing. I am grateful to the team leader Zhiying Wang. He is not only a wise educator to guide me on the way to be a researcher, but also a kind of elder to provide selfless help in life. I also want to give my big thank to Sheng Ma. He provided hug help at the start of my research. I also need to thank for my friends in NUDT, Wei Chen, Cheng Qian, Qi Yu, Boqian Wang, Lu Wang, Quan Deng, and Peng Xun. It is benefit and joyful for me to share research experiences with them.

Studying in Leiden Embedded Research Center (Lerc) in Leiden University is a pleasure and unforgettable memory. In Lerc, I am very lucky for having so many kind and helpful colleagues, Di Liu, Jelena Spasic,Sobhan Niknam, Erqian Tang, Svetlana Minakova, and Christian Fuchs, I enjoy each interest talk and your helpful suggestions on my researches. I wish you have more excellent publications in your future academic career. My big thank goes to Sobhan Niknam. I really appreciate the assistance and advice I obtained from him. In the long Ph.D journey, we have experienced a lot of frustrated and lost moments in our researches. We shared our feeling and research experience, and encourage each other to go forward. As a nice friend, he was always enthusiastic about helping others. No matter when I ware confused or fell in trouble, he always tried his best to help me.

In Leiden, I have been really lucky to make many terrific friends. I am grateful to Qinggang Hao, Kaifeng Yang, Zhuang Li, Zhen Wang, and Jialong Liu for their help in many ways. My gratitude also goes to their families. Family life with kids is always busy, like a battle, but in weekends, we always had wonderful family parties, which was not only a wonderful time to have a rest, but an opportunity to share the feeling of being parents.

Finally, I want to thank my family member for their unselfish support and patience, my sister-in-law Zhe Zhang, my mother-in-law Xiuping Chang, and my father-in-law Weidong Zhang. At the first two years of my Ph.D, you helped me to take care of my newborn son in China. Without your help and support, I could not make the decision to start this oversea journey and I could not have so much time and energy on my research. I am sincerely indebted to my sister He Wang, my mother Yajun Xie, and my father Renyi Wang. Without their unconditional support and continuous love, I could not have the energy to pursue my goals or dreams. In the journey of a life, there are some important and critical points along the track of a person. The most important point for me is to meet my perfect and wonderful wife Jian Zhang. I really appreciate your sacrifice and selfless dedication for our family. Your comfort and support makes me full of motivation to deal with every frustrated moment on the journey of life. My brave and naughty knight, Linchuan Wang, you are the sunlight of our family. Thank you for taking so much joyful time for our family.

Curriculum Vitae

Peng Wang was born on February 20, 1990 in Liaoning, China. He obtained his B.Eng degree in Electronic Information Science and Technology from Harbin Institute of Technology, China in 2012, and got M.Eng degree in Computer Science from National University of Defense Technology, China, in 2015. After his graduation, he joined the Leiden Embedded Research Center at Leiden University as a PhD student, where his research work was mainly sponsored by China Scholarship Council.