Improving GPGPU NoC Performance with Memory Controller Placement Awareness

AN ABSTRACT OF THE PROJECT OF

Lakshman Madhav Kollipara, for the degree of Master of Science in Computer Science presented on December 8, 2016.

Title: Improving GPGPU NoC Performance with Memory Controller Placement Awareness.

Abstract approved:

_____

Dr. Lizhong Chen

General Purpose Graphics Processing Units (GPGPUs) are used in modern computational workloads for their thread level parallelism (TLP) and highly programmable cores which allow thousands of threads to execute in parallel. The fast-scaling of GPGPUs have increased the demand for performance optimizations on Network-On-Chip(NoC) designs. Previous works have exploited NoC designs in the Chip Multiprocessor (CMP) environments but not much in GPGPU systems. Unlike CMPs, traffic is highly asymmetric in the GPGPUs because of many cores and only few Memory Controllers (MCs).

The highly asymmetric traffic impacts the resource utilization and performance of NoC. This work aims at analyzing the maximum channel load associated with different MC placements and VC partitioning on a single network NoC. The prior work has introduced the concept of VC monopolization which increases the performance but also increases the link contention and power consumption. With this work, we further optimize the VC monopolization scheme by reducing the link contention resulting in higher performance and reduced power consumption. The proposed schemes improve performance by 14% in terms of IPC and reduces latency by 21% compared to baseline versions.

Improving GPGPU NoC Performance with Memory Controller Placement Awareness

by

Lakshman Madhav Kollipara

A PROJECT

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented December 8, 2016
Commencement June 2017

Master of Science project of Lakshman Madhav Kollipara presented on December 8, 2016

APPROVED:

Major Professor, representing Electrical and Computer Engineering

Director of the School of Electrical Engineering and Computer Science

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Lakshman Madhav Kollipara, Author

ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor, Dr. Lizhong Chen for his constant support and guidance throughout this work. Working with him for 15 months and attending his courses helped me in analyzing and understanding the important concepts required for this work.

I'd like to thank my fellow researchers in our STAR lab for their support and setting up the environment for running simulators. I am also thankful to Dr. Anita Sharma and Dr. Bechir Hamdaoui for accepting the invitation and taking the time to be part of my Graduate committee.

I'd like to thank Nicole Thompson, Darcy Miller and the other members of the EECS office and graduate school who were very helpful for the logistics to complete my Master's program.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure Page

LIST OF TABLES

Table Page

Improving GPGPU NoC Performance with Memory Controller Placement Awareness

Chapter 1: Introduction

General purpose Graphics Processing Units (GPGPUs) have emerged to run a wide range of high-performance computing workloads which have a high thread level parallelism (TLP) [3]. GPGPUs contain numerous programmable computational cores called as Shader Cores to run simultaneous active threads. These shader cores are connected to memory modules called Memory Controllers (MCs) through an interconnection network. As the number of on-chip cores increases, a scalable and high-bandwidth interconnects to connect them is critically important [4,5,6]. The fast packet switched interconnects have replaced existing buses and crossbars to reduce the congestion and area on the chip. The two primary costs associated with an on-chip network are area and power. In traditional interconnect, routers are all input-queued which means at each port of the router we have input buffers. These input buffers are responsible for storing packets when they arrive at the router and housing them throughout their duration in the router.

Ideally, the interconnects should ensure deadlock freedom, low latency and high throughput while using minimal network resources such as Virtual Channels (VCs) and Physical channels(links). VCs are used in interconnects to avoid deadlocks and Head-of-line blocking [7]. Head-of-line blocking occurs when there is a single queue at each input; when a packet at the head of the queue is blocked, it stalls the subsequent packets that are behind it even there are available resources for stalled packets. VC is basically a separate queue in the router. Multiple VCs share the physical link between the routers and switch cycle by cycle. When a packet holding a virtual channel is blocked, other packets will traverse the physical link through other VCs. Figure 1 shows the virtual channel usage and how the packets pass through the stalled packets.

NoCs are being used in Chip-Multiprocessors(CMPs) because of their flexibility and scalability and the work on this domain has matured whereas, there are only a few works on NoC design in GPGPUs [8,9,10]. In CMPs, traffic is uniformly distributed among the cores resulting in better load balancing on the NoC. In the case of GPGPUs, traffic is highly asymmetric between cores and MCs. GPGPUs have many shader cores and only a few MCs which results in poor load balancing on the interconnect. Many request packets go from all the cores to each MC and each MC sends a reply packet to each core. This results in a high contention on the links around MCs and these MCs become hotspots [8] leading to skewed usage of NoC resources. The reply traffic from MCs to the core is specifically high degrades the overall system performance and causes a network bottleneck.

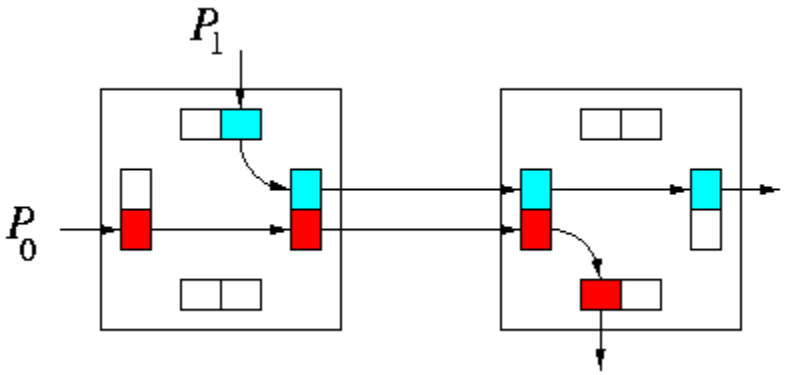

Figure 1. Virtual Channel Usage and Switching (2 VCs)

Prior work [8,9,11] proposed partitioning interconnect into two independent equally divided subnetworks to avoid cyclic dependencies between different classes of traffic to avoid protocol level deadlocks. They have two physical networks having one network for request traffic and one network for reply traffic which completely avoids traffic contention and deadlock. As mentioned earlier, the load on the reply network is high and results in poor load balancing and fails to maximize the system performance. The throughput effectiveness is really important in the case of GPGPUs because GPGPUs are throughput critical and not latency critical[8]. Therefore, it is important to

design a bandwidth-efficient NoCs for GPGPUs considering their asymmetric traffic requirements and throughput criticality.

Therefore, this study evaluates and analyzes the impact of different MC placements in the network, different routing algorithms and the need for physically partitioned networks. In this work, we analyzed the traffic patterns in GPGPUs. Then, motivated by existing work [1,2] on MC placement and VC monopolization, we expanded their work by proposing better MC controller placements, VC partitioning and reducing the link utilization and power consumption. We also studied the effects of monopolization on XY-YX routing under diverse MC placements. Our simulation results show that the performance of our proposed scheme is increased by up to 14% in terms of IPC and 21% reduction in latency compared to the baseline NoC.

This report is organized as follows: Chapter 2 discusses the related work in the field of NoCs in CPUs and GPUs. Chapter 3 discusses the background on NoCs in GPGPUs and motivation from existing works for this work Chapter 4 talks about the problems associated with existing work and how they are addressed with newly proposed schemes. Chapter 5 discusses the performance evaluation of schemes proposed in Section 4. Chapter 6 discusses the limitation of this work and future scope. Chapter 7 ends with conclusion of the proposed scheme.

Chapter 2: Related Work

While a large body of related research has been dedicated to improving NoCs in the CPU domain, only a handful of works [2,8,9,15,20] have explored the impact of NoC design in GPGPUs. Several aspects of NoCs, such as topology [21,22,23], routing [24,25,26,27], flow control [28,29,30], power consumption [34,35], and router architecture [31,32,33] were improved a lot in CPU domain. However, due to considerably different requirements, traffic patterns in GPGPUs, there is need for effective NoC designs in GPGPUs.

William J. Dally and Brian Towels [17] proposed the concept of using on-chip interconnection networks instead of ad-hoc point-to-point wiring structures. The system modules communicate among each other by sending packets resulting in high performance circuits and reduced area on chip. They also proposed the router architecture and flow control mechanisms to route the packets effectively. William J. Dally [18] proposed the concept of virtual channel flow control where each physical network channel is divided into several small queues called as virtual channels rather than a single queue. This decoupling, due to separate virtual channels allows active messages to pass blocked messages and thus avoiding head of the line blocking. They showed that the virtual channel flow control can increase network throughput by a factor of 4. Seitz et al. [19] proposed a deadlock free routing algorithm by using virtual channels. They proposed a general approach to prevent deadlocks in interconnection networks with the help of Channel dependency graphs. The deadlocks can be avoided by restricting the routing from few set of nodes to eliminate the cycle formation in the network.

Bakhoda et al. [15] evaluated the impact of NoCs in GPGPUs by designing a simulator combining NoCs and GPU architecture. A “checker-board” NoC with oblivious

routing [8] is proposed by replacing some of the full-routers with half routers with limited connectivity and placing L2 cache at these half routers maintaining a minimum hop count. The cost reduction by employing asymmetric designs and removing the unused links and routers in the network based on traffic patterns is discussed in [20]. A direct all-to-all overlay (DA2Mesh) network [9] is proposed to minimize the traffic the traffic contention across the memory controllers by using multiple dedicated narrow networks enabling higher frequency to increase effective bandwidth. A state-of-the-art router [8] has been proposed with multiple injection ports to accelerate injection of the packets into the network. Jang et al. [2] proposed a concept of VC monopolization where all the VCs can be fully monopolized by either request traffic or reply traffic. Our work explores the potential improvements to the VC monopolization scheme by combining with the new MC placements and overcome the problems with VC monopolization. We mainly focused on potential bottleneck of contention around links of memory controller and the max channel load to improve network performance.

Chapter 3: Background & Motivation

In this section, we discuss the high level design of NoCs in GPGPUs, traffic patterns in GPGPUs, Memory controller placements and the concept of VC monopolization.

3.1 GPGPU Network-on-Chip

A GPGPU consists of many SIMT cores called Streaming Multiprocessors (SMs), each contains 16 scalar processors[SPs]. These SMs are connected to the Memory modules called Memory Controllers (MCs). A Memory Controller(MC) node consists of a L2 cache bank and a memory controller that interfaces with off-chip GDDR memory. Each SM or MC node is connected to a router through a network interface. This network Interface is responsible for sending and receiving of packets from SMs to MC and vice versa. High-level architecture of latest NVIDIA Tesla GPU with SMs and MCs (L2 cache) is shown in Figure 2.

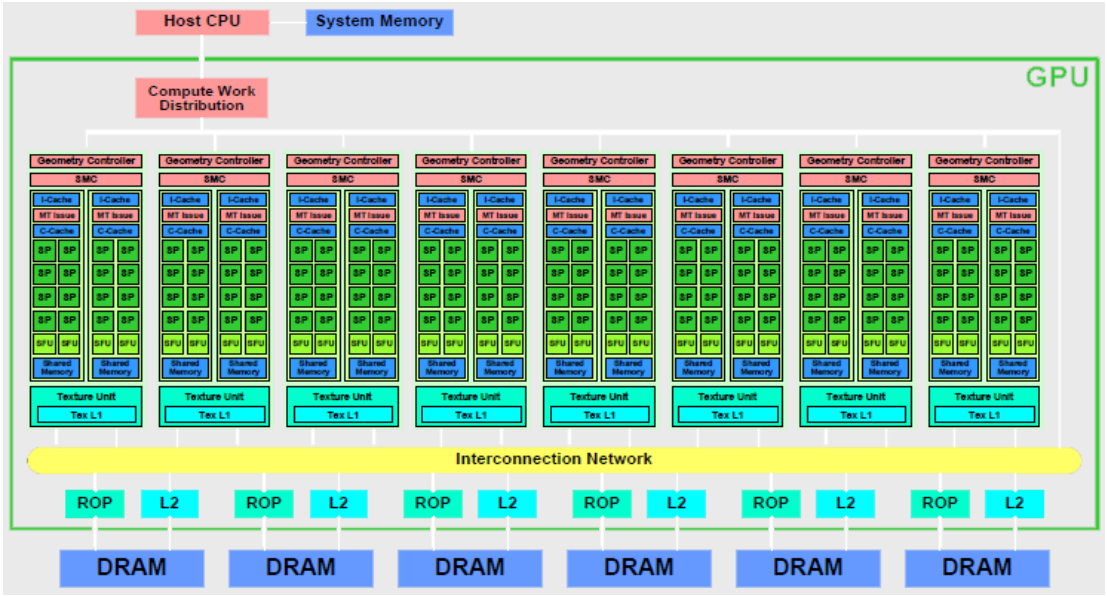

Figure 2. Architecture of NVIDIA Tesla GPU (Source: NVIDIA)

Each SM has access to 16KB low latency highly-banked per-core shared memory, global texture memory with a per-core constant cache, a global constant memory with a per-core constant cache, a private L1 cache and register files. Each MC is associated with a slice of shared L2 cache which assume write-back policy [4] for faster access. In order to access global memory, memory requests must be sent via interconnects to the corresponding MCs, which are physically distributed over the chip. To avoid protocol level deadlock, separate physical networks are used for requests and replies which can be replaced by alternate separate logical networks(VCs). We are considering 2D-mesh and dimension order routing for its simplicity, scalability, and regularity [8]. Figure 3 shows the NoC layout of an 4x4 2D-Mesh.

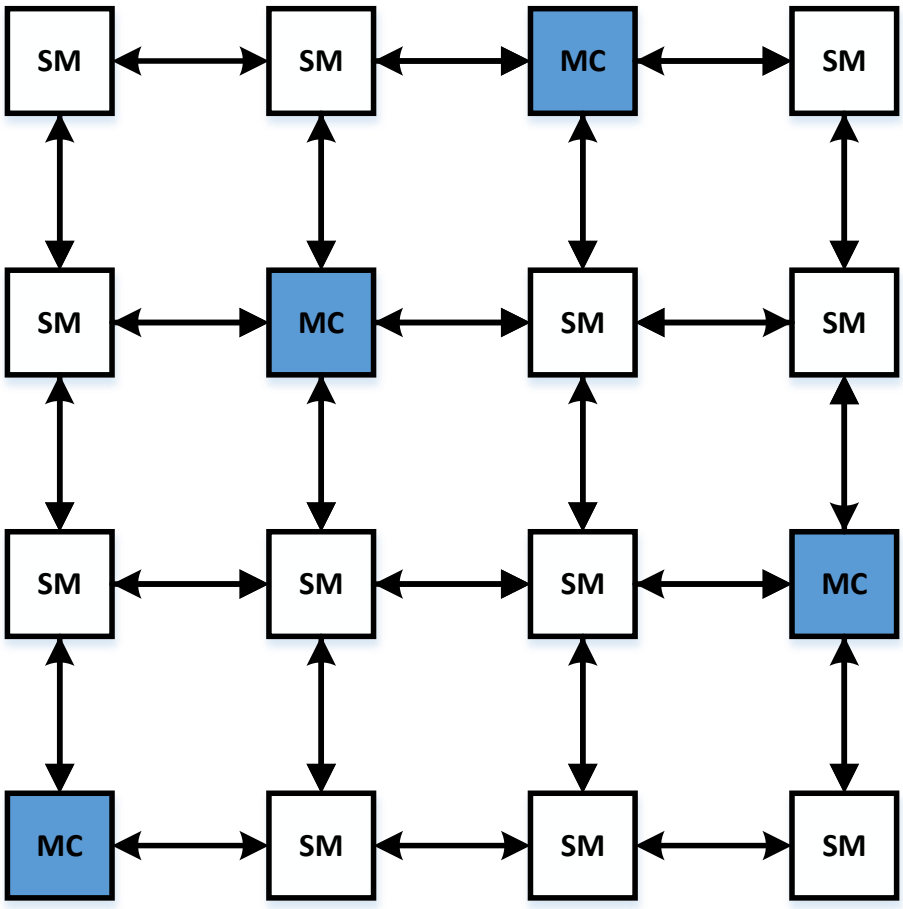

Figure 3. 4x4 2D-Mesh with 4 MCs

The baseline NoC router consists of 5 I/O ports to connect to the SM, MC and other routers in the network. The baseline router uses speculation and look-ahead routing to reduce the latency [12]. Each router is input-queued with multiple VCs per input port and uses flit based flow control [13]. To avoid buffer overflows, credit based backpressure mechanism is used [13]. Figure 4 shows the high-level architecture of a baseline router.

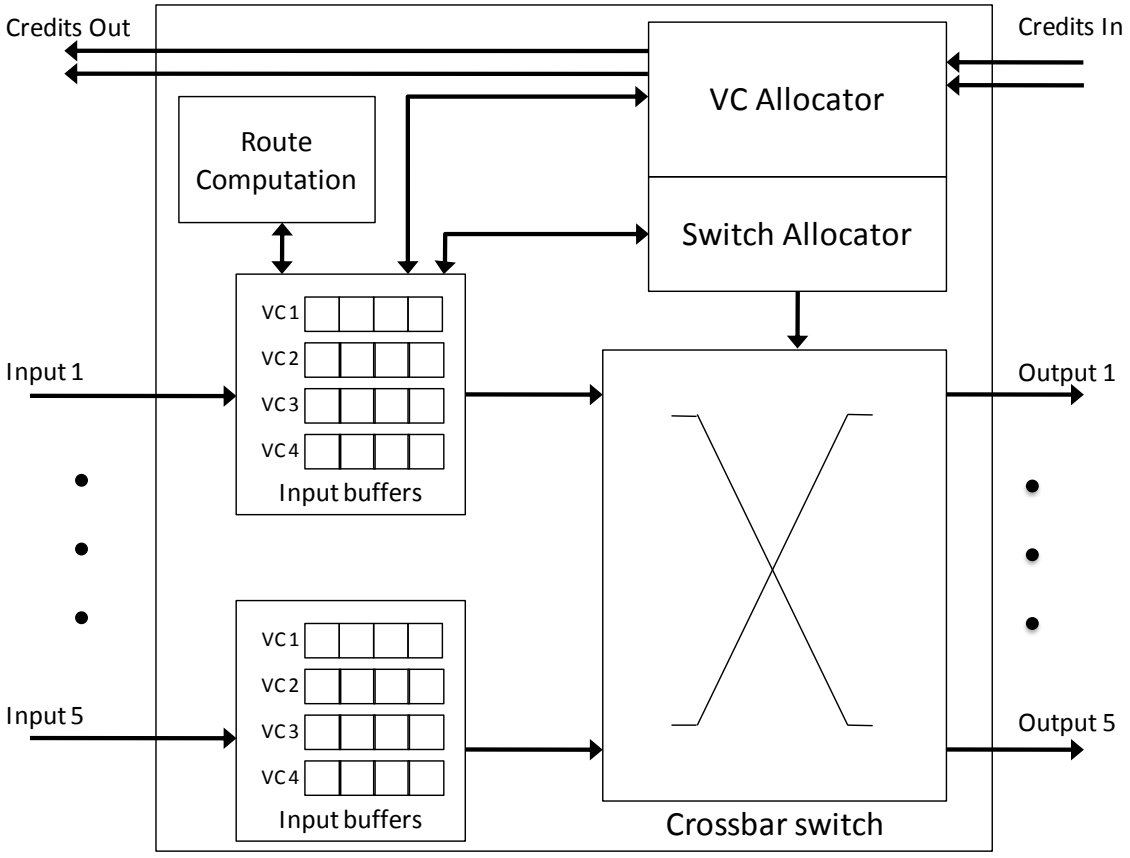

Figure 4. High-level architecture of baseline VC Router

We employed Dimension-order routing (DOR) due to its simplicity and ability to monopolize VCs. In Dimension-order routing, all messages from node A to node B will always traverse the same path. With DOR, a packet traverses the network dimension-by-

dimension, reaching the ordinate matching to its destination before switching to next dimension. There are other routing algorithms such as adaptive routing [13] and oblivious routing [13] which were not considered because of the critical path delays [1] and requirement of more virtual channels.

3.2 Traffic Patterns in GPGPU NoCs

There are many streaming multiprocessors and only few memory controllers in the GPGPU architecture. Further, we have many-to-few request packets from SM cores to MCs and few-to-many reply packets from MCs to SM cores [8]. Figure 5 shows the Many-to-Few-to-Many traffic in GPGPUS.

If a local L1 cache miss happens in any of the SMs, a cache data request is generated by the SM core and sent into the network. The request packet is injected into the request network through injection ports of the SM core. These request packets are forwarded through the routers and eventually reach the destination router connected to its associated MC. If the L2 cache at MC is also missed, MC forwards the request to off-chip GDDR memory. Once the requested data is received by the MC from the off-chip main memory, the network interface formats the reply packet and injects it into the reply network. The reply packet is forwarded through the routers in the network and reach the requested SM core. If we have L2 cache hit at the MC node, off-chip memory access will be skipped and the reply data is directly injected into the reply network.

Read request traffic is of many short packets of cache line address and write request traffic is of few long packets of the entire cache line. Read reply traffic is of many long packets of the entire cache line from MC and write reply traffic is of few short packets of cache line acknowledgments.

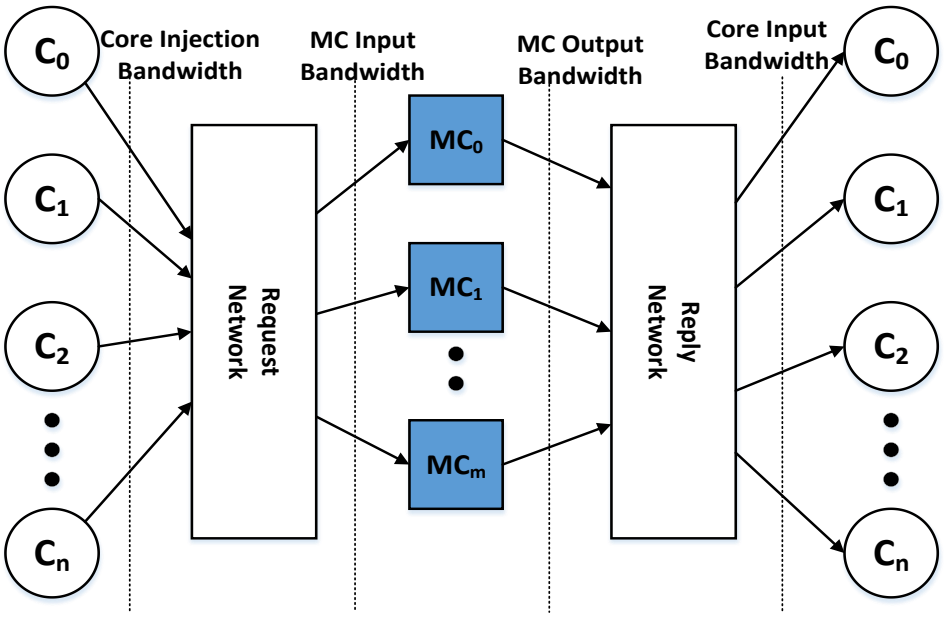

Figure 5. Many-to-Few-to-Many On-chip Traffic

Usually, read requests and write replies are mapped into a single flit whereas, read replies and write requests are mapped into 3-5 flits. Figure 6 shows the distribution of flits among read request, read reply, write request and write reply packets. On average, 61% of the NoC traffic in GPGPUs is from read replies. Therefore, we can conclude that the reply network is much congested than request network and the reply traffic gets congested on the links around MCs.

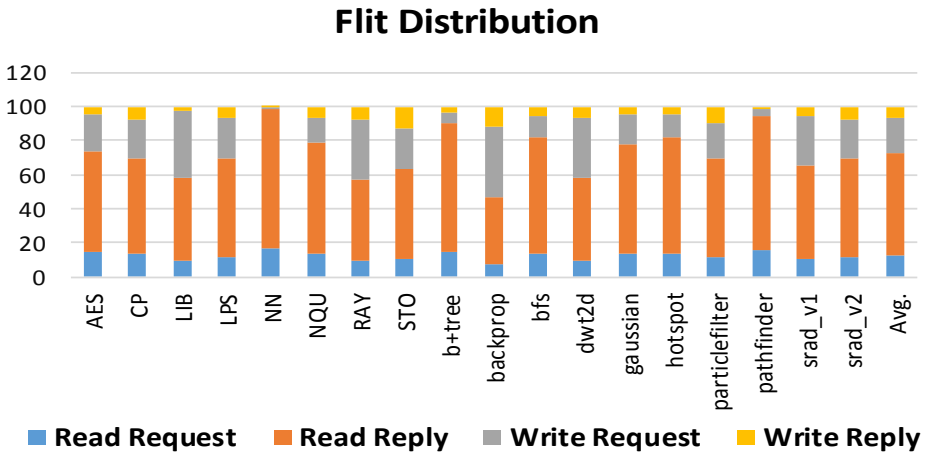

Figure 6. Packet-Type Distribution of GPGPU Benchmarks

3.3 Memory Controller Placement

The MC placement effects the network latency and bandwidth of the interconnect by distributing the load across the network. Dennis et al. [1] evaluated the impact of memory controller location and the influence of processor-memory traffic on on-chip networks. They studied all permutations of memory controllers in a 2D-mesh topology and found a configuration that minimizes maximum channel load. Figure 7 shows different MC placements proposed in their work. Their work does not analyze the effects of MC placement in GPGPUs. Jang et al. [2] have done a quantitative analysis on MC placements in GPGPUs and showed how distributed MC location can improve the NoC efficiency in GPGPUs.

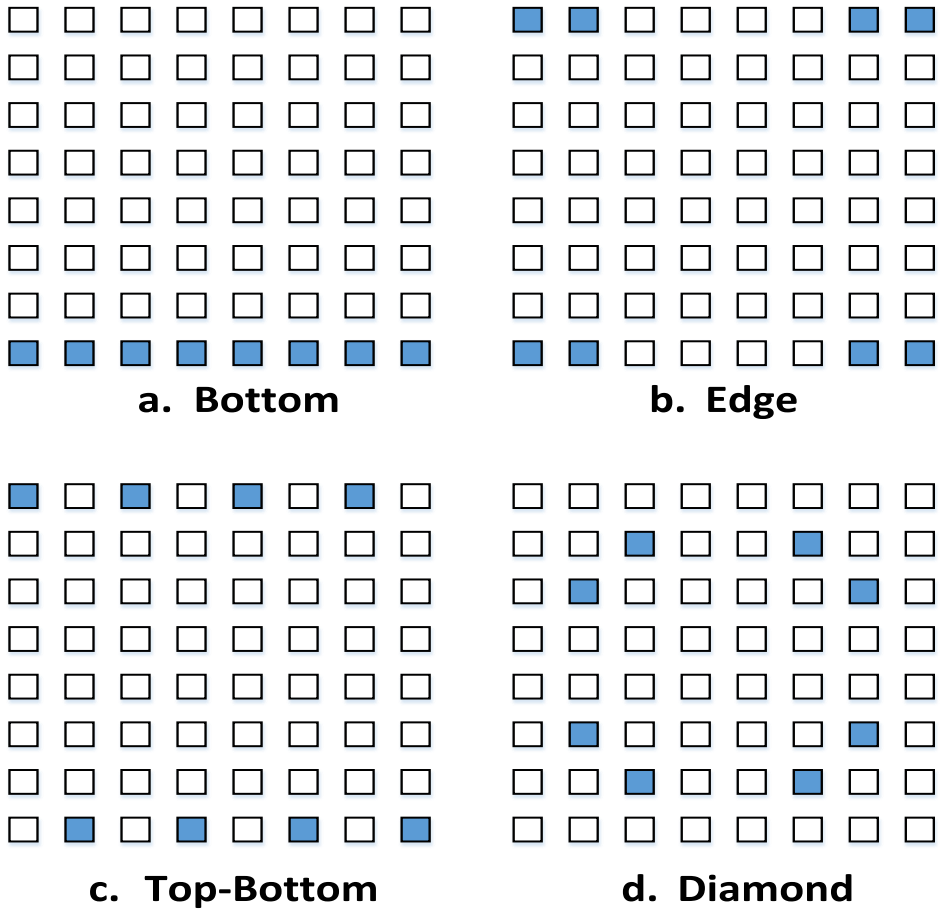

Figure 7. Different MC placements co-located with GPGPU cores (shaded tiles represent MCs)

The effectiveness of MC placement is evaluated by the minimum hop count from cores to MC. Among all other MC placements, Diamond MC placement shown in Figure 7(d) allows minimum number of hops from the cores to MCs. Since we are considering only Dimension order routing, there is only one unique path from each core to give MC irrespective of XY or YX or XY-YX routing. Jang et al. [2] gave an estimate to calculate the number of hops for any MC placement. For a NxN mesh with N number of MCs and N^2 – N cores, the average number of hops from cores to MCs is given by the Manhattan distance between each MC and each SM core:

$$H_{avg} = \frac{H_{verti} + H_{hori}}{[N^2 - N] * N} \quad (1)$$

In the above equation, H_{vert} and H_{hori} are the aggregated hops in vertical and horizontal directions respectively. Among all MC placements, Diamond has the lowest hop count of 7. Bottom, Edge and Top-Bottom have a hop count of 8 which also corresponds to results from prior work [1].

3.4 VC Monopolization

The network resources such as VCs and physical links are shared by both request and reply packets. To avoid the deadlock, reply packets and request packets must not compete for same resources. Prior work [8,9,11] suggested partitioning NoC into two physical subnetworks for different traffic: one for the request and one for the reply. This sort of two physical networks [9] doubles the number of routers and link resources resulting in hardware overhead. The use of single physical network and separating the traffic by using logical subnetworks (VCs) reduce the area overhead and power consumption of NoC, but results in higher link contention and less resource utilization. Therefore, Jang et al. [2] proposed a virtual channel partitioning technique called VC monopolization, where all the VCs can be fully monopolized by either request traffic or reply traffic. Each set of VCs are multiplexed over one physical link. Figure 8 shows the

concept of VC monopolization compared to Non monopolized virtual channel flow control.

In case of single physical network, each input port at the router has 2 VCs per port. One VC is used for request traffic and other VC is used for reply traffic. In case of VC monopolization, if there is no mixed traffic on the link, i.e. each unidirectional link is composed of only either request traffic or only reply traffic then, both VCs can be monopolized by either request traffic or reply traffic, providing more buffer resources.

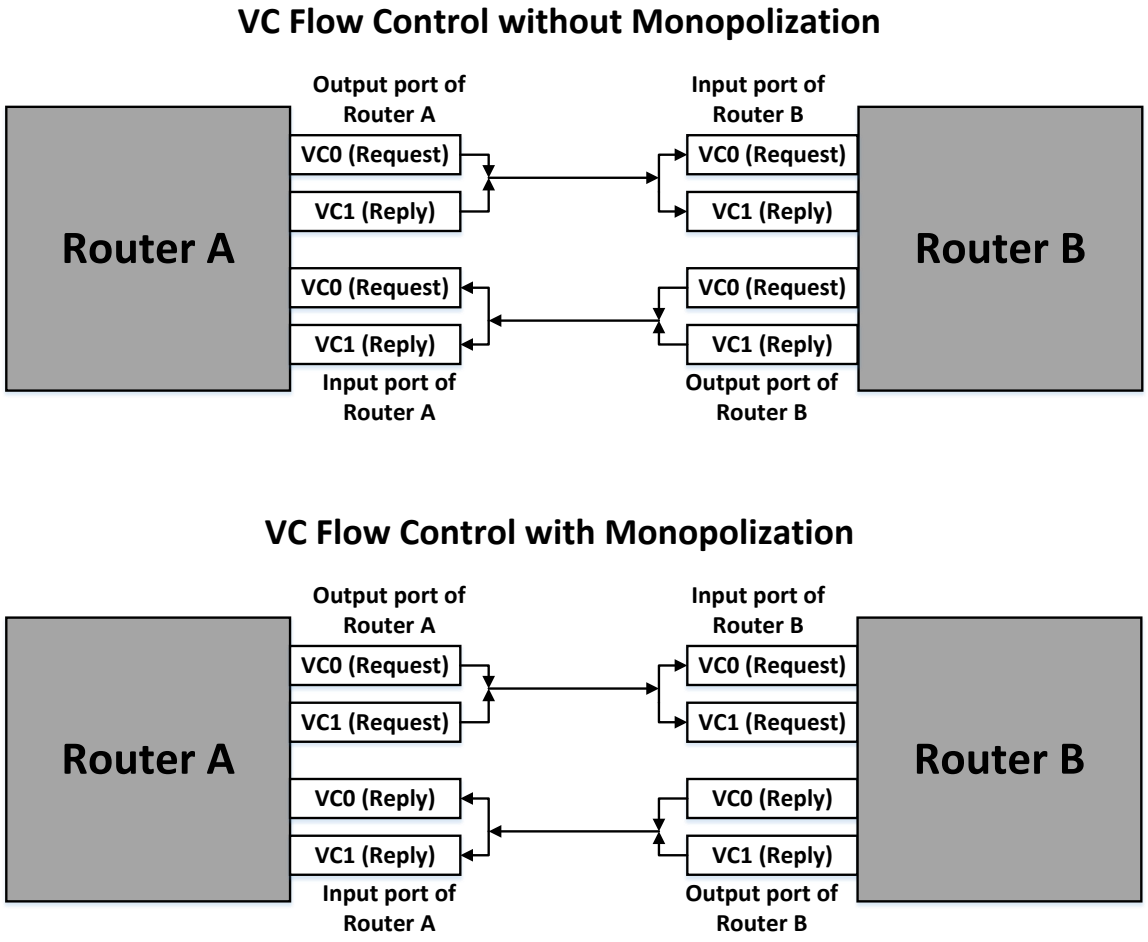

Figure 8. VC Monopolization by separating request and reply traffic

In the case of Bottom, MC placement, where all MCs are located at the bottom, request and reply traffic take separate paths and are not overlapped on any of the links with dimension order routing. Figure 9 shows the VC monopolization on a 4x4 2D-mesh with bottom MC placement and YX routing. Among all the dimension order routing, YX routing with Bottom MC placement and full monopolization outperforms all other MC placements with 80% improvement [2]. Therefore, there is no need to split a network into two to prevent deadlocks. However, VC monopolization is not feasible when we have mixed traffic on the links.

The problem associated with this approach is high link contention on the links between MCs in the request network and the links traversing out the MCs in the reply network. The thickness of the links indicates the heaver traffic on the links around MCs. These links are bottleneck to the performance and lead to poor load balancing. We will study more about this drawback in the later sections.

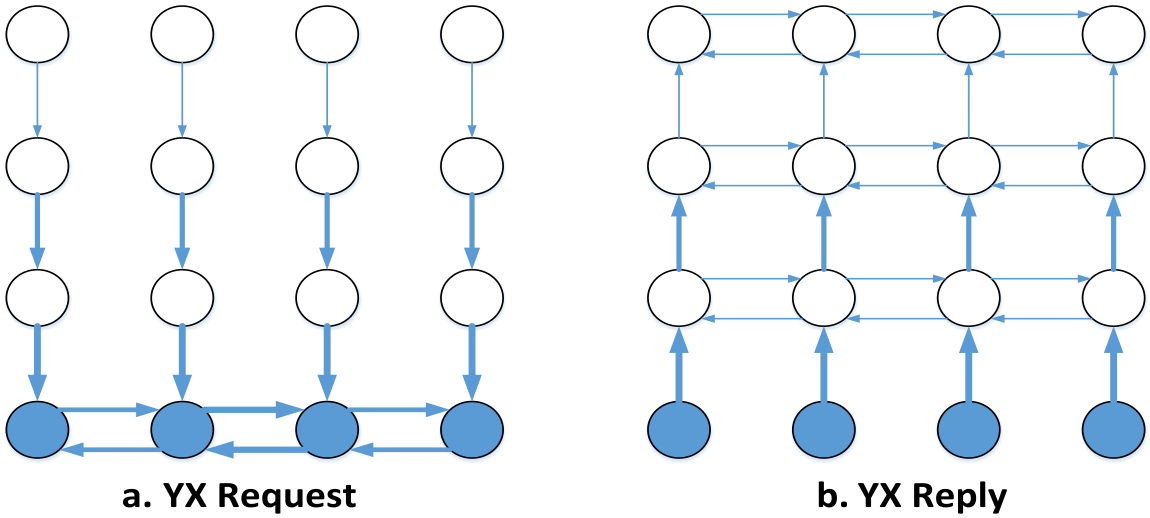

Figure 9. VC Monopolization in Bottom MC Placement (request(a) and reply(b) take different paths, No mixed traffic on links)

Chapter 4: Designing High-Performance NoCs in GPGPUs

With the basis of asymmetric traffic patterns in GPGPUs, we analyze the maximum channel load associated with VC monopolization from the prior work [2]. Based on this analysis and existing MC placement schemes, we propose better MC placements along with XY-YX routing and VC partitioning schemes to achieve higher bandwidth and reduce the contention over the links.

4.1 Maximum Channel Load associated with VC Monopolization

Jang et al. [2] proposed a VC monopolization scheme in a 2D-mesh with Bottom placement as shown in Figure 10. In their scheme, request and reply traffic are not overlapped with dimension order routing. Thus, all the VCs can be fully monopolized by either request or reply traffic. Their scheme is not feasible for other MC placements if there is mixed traffic on the links. The drawback of this approach is YX routing in VC monopolization scheme increases network contention due to the high volume of request traffic between MCs and YX routing also increases network contention on the links going outward from MCs. We used link utilization as a metric to measure contention on the links. The number on each link represents the number of nodes the link is serving i.e. number 4 on link represents that particular link is serving traffic to/from 4 other nodes. Figure 10 shows the link utilization of the VC monopolization scheme. The link utilization around the MCs is very high compared to other links which result in poor load balancing and less network bandwidth.

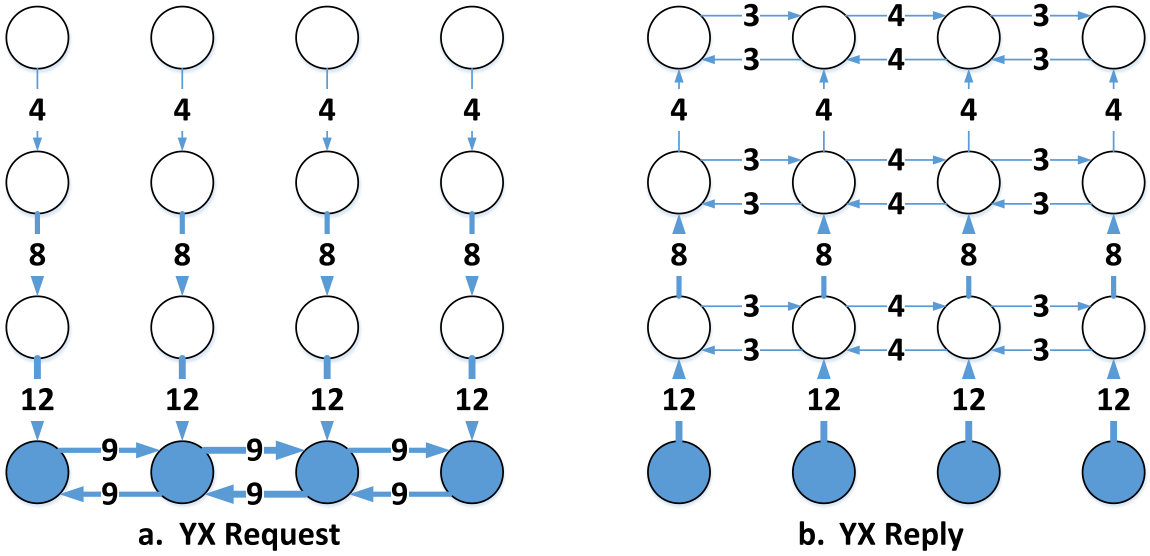

Figure 10. Link Utilization of Bottom MC placement with VC Monopolization

Figure 10(a) shows that the link contention is increased as the request traffic is approaching MCs and in between MCs as well. Figure 10(b) shows that the link contention is high on the links going outward MCs. One of the abstract metric for NoC performance is the Maximum Channel Load which is determined by the most congested link for a given traffic pattern. The maximum load channel will limit the overall network bandwidth. Therefore, VC monopolization with Bottom MC placement results in a Maximum channel load of 12 and hop count of 8.

To address the above stated issues and improve the NoC performance, we came up with new MC placements and better VC partitioning schemes combined to reduce the maximum channel load, hop count and contention of links around MCs. We have used XY-YX (one of the dimension order routing) routing in our proposed schemes to take advantage of monopolizing few links with no mixed traffic. In XY-YX routing, request subnetwork routes packets in X dimension first and then, shifts to Y dimension whereas, reply network routes packets in Y dimension first and then shifts to X dimension. This

sort of routing helps to achieve significant performance improvement because heavy traffic between MCs is distributed among links in other directions rather than on one single link providing different routing paths for request and reply packets.

The advantage of XY-YX routing is the links with mixed traffic can have a separate virtual channel for each packet type: one for the request and one for the reply. The links which do not have any mixed traffic can still be monopolized for higher buffer utilization. In the case of our MC placement schemes and XY-YX routing, we have scope to partition VCs such that all the vertical links can be fully monopolized with either request or reply traffic. Few horizontal links can also be monopolized depending on the MC location and mixed request and reply traffic.

4.2 VC partitioning with various MC Placement Schemes

As discussed in Section 3, one way of improving NoC performance is to reduce traffic congestion by spreading process-memory traffic and balancing the load by spreading the MCs across the NoC. Prior work [1] on MC placements showed the performance improvements from MC placements in CMPs but, not in GPGPUs. Therefore, we came up with 4 additional MC placement schemes, namely Diagonal, 8-Queen, Step and Zig-zag. The VC partitioning schemes across the links with proposed MC placement schemes are studied by calculating the max channel load and the average hop count for each MC placement. The advantage of these MC placements over existing MC placement schemes can be further studied with VC partitioning schemes discussed in later sections.

4.2.1 Diagonal MC Placement

We propose a new MC placement scheme where all the MCs are placed diagonally among the SMs. Diagonal MC placement with XY-YX routing has an average hop count of 7 which is similar to Diamond but, the performance gain is through the

combination of hop count and reduced maximum channel load. Figure 11 shows that the max channel load on the XY request network and YX reply network. The max channel load in case of Diamond MC placement is 9 and is better than that of VC monopolization with Bottom MC placement which has a max channel load of 12. The disadvantage with this scheme is the MCs located in the corners have only two links to distribute the traffic across the other SMs. Out of these two links, one link is serving heavier load compared to the other link which is serving only three other cores co-located in the same row.

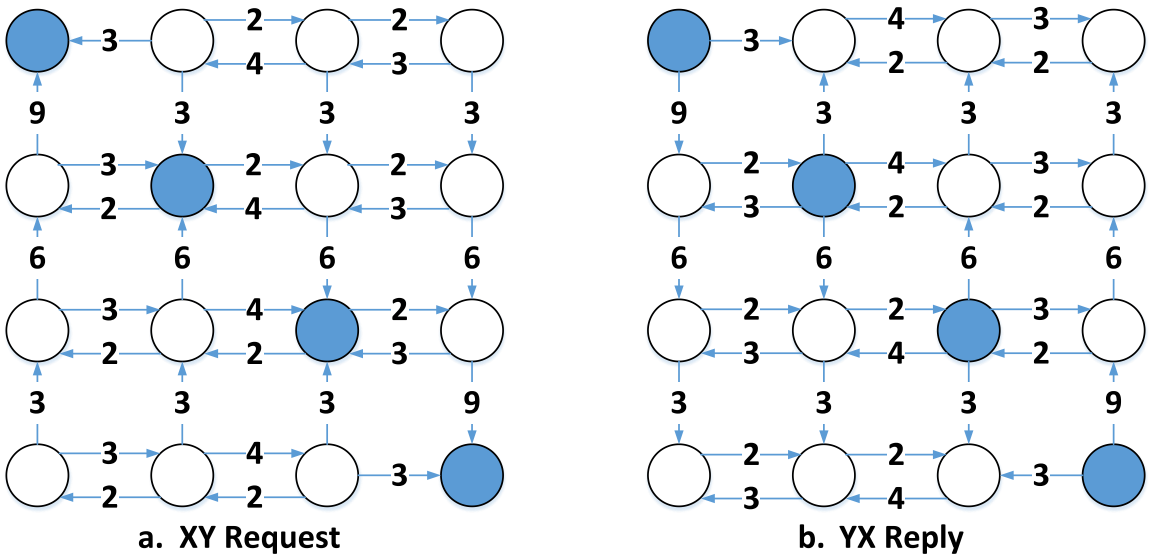

Figure 11. Diagonal MC placement with XY-YX routing

4.2.2 8-Queen MC Placement

To avoid contention on MCs located in the corner, we came up with an 8-queen MC placement where no two MCs will be in the same row or same column. In this MC placement, you can consider any solution of 8-queen other than the ones which have a queen in the corner nodes. This particular placement ensures that four queens are always placed on the edges and provides a scope for monopolizing few horizontal links as well. 8-queen MC placement has an average hop count of 7 which is also similar to Diagonal

and Diamond but eliminates the MCs in the corners. Figure 12 shows that the max channel load on the XY request network and YX reply network. The max channel load in case of 8-queen MC placement is 9 and is better than that of VC monopolization with Bottom MC placement and same as of Diagonal MC placement. The disadvantage with this scheme is the MCs located in the top row and bottom row have only two links to distribute the traffic. Out of these two links, one link is serving heavier load compared to the other link which is serving only three other cores co-located in the same row.

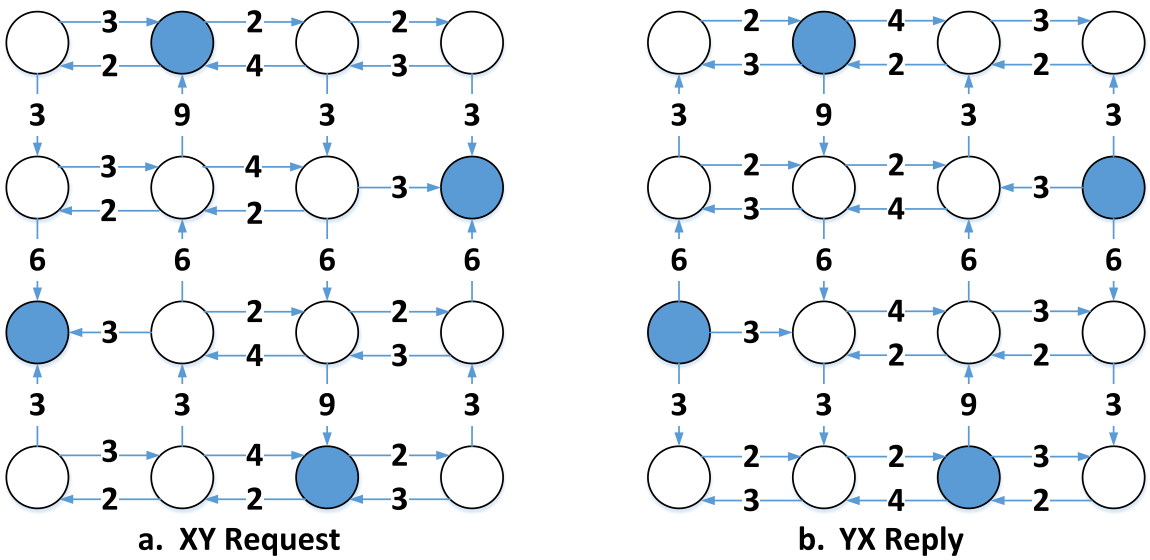

Figure 12. 8-Queen MC placement with XY-YX routing

4.2.3 Zig-zag MC Placement

Eliminating MCs at the corners is not sufficient enough, the MCs on top and bottom links should also be taken care of to reduce the maximum channel load on the network. In the case of 8-queen, though we are avoiding MCs at the corners, the link contention is high at the two MCs located in top and bottom row of the mesh network. To reduce the contention on the links around MCs in the top and bottom rows, we came up with the zig-zag MC placement where MCs are alternatively placed in the middle two

rows of the mesh network. This MC placement will restrict the MCs in the corners and in the top and bottom rows. Zig-zag MC placement has an average hop count of 6 which is better than all existing MC placement schemes. Figure 13 shows the Max channel load on the XY request network and YX reply network. The max channel load on Zig-zag MC placement is 6 on vertical links and 8 on horizontal links which is better than Bottom with VC monopolization, Diagonal, and 8-queen with XY-YX routing.

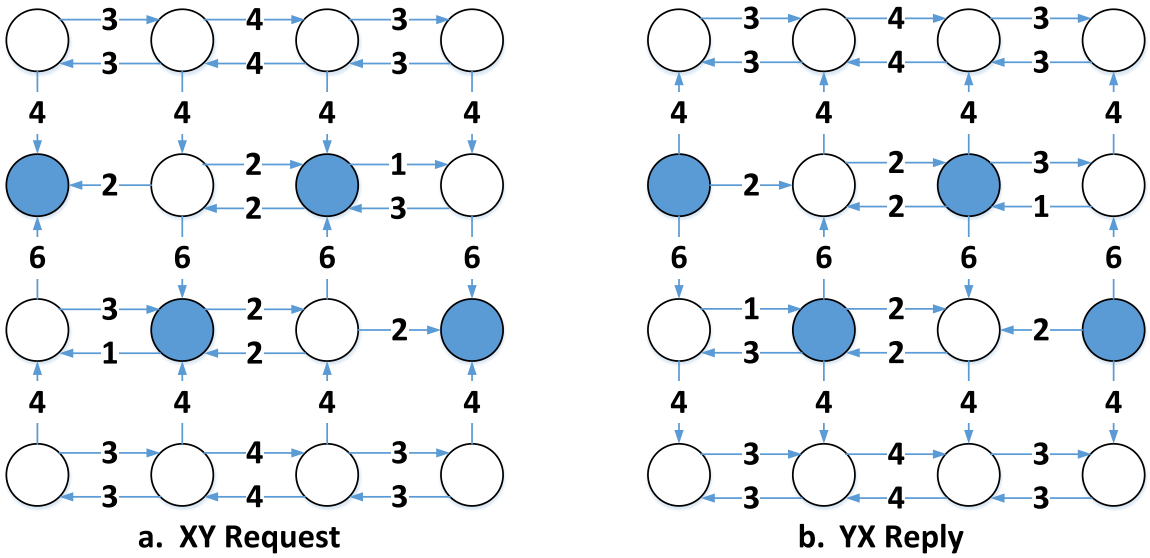

Figure 13. Zig-zag MC placement with XY-YX routing

4.2.4 Step MC Placement

Another similar MC placement scheme is Step MC placement where half of the memory controllers are placed to the left on the lower middle row and remaining half of the memory controllers are placed in the right of the upper middle row in a mesh network. This MC placement will also restrict the MCs in the corners and in the top and bottom rows. Step MC placement has an average hop count of 6 and max channel load of 6 on virtual links and 8 on horizontal links which is similar to Zig-zag MC placement. Figure 14 shows the link contention with Step MC placement with XY-YX routing.

Though the max channel load is similar to that of Zig-zag MC placement, Zig-zag MC placement performs better than the Step MC placement because of having dedicated VCs for request and reply traffic. Two MCs co-located in the same row will increase the reply traffic on the links outgoing from MCs because reply traffic is always higher than request traffic.

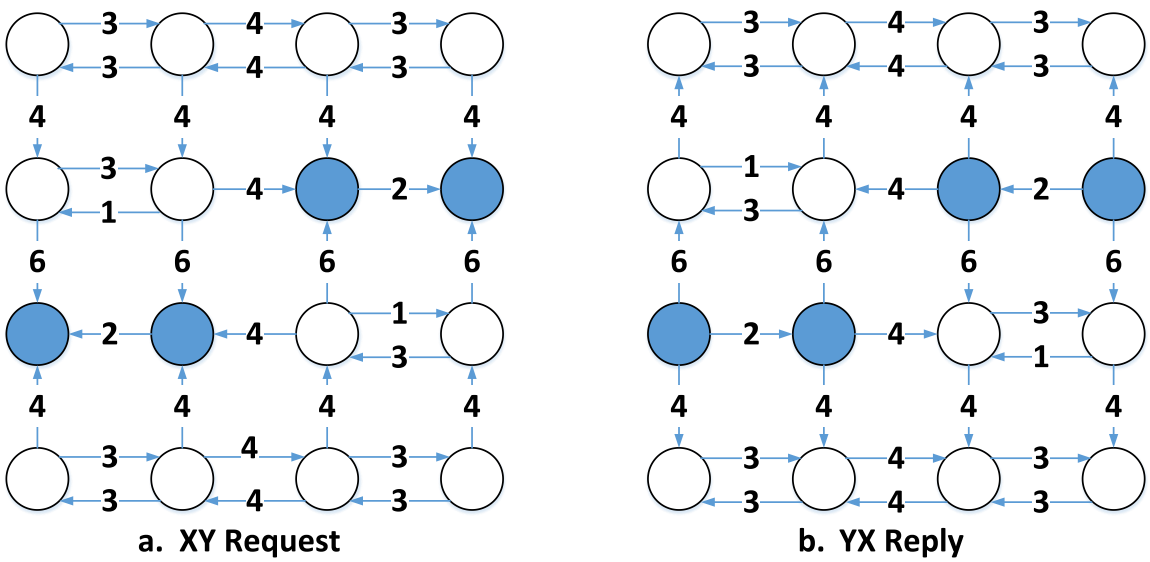

Figure 14. Step MC placement with XY-YX routing

Among all the proposed MC placement and VC partitioning schemes, the top performer is Zig-zag MC placement with a reduced average hop count of 6 and Max channel load of 6 on vertical links and 8 on horizontal links. Step also has similar max channel load and hop count as Zig-zag MC placement but dedicated VCs in the middle horizontal links in Zig-zag help in improving performance. Diagonal and 8-Queen have a hop count of 7 which is similar to Diamond. All the four newly proposed MC placements have reduced maximum channel load when compared to Bottom MC placement with VC Monopolization. Figure 15 shows the Diagonal, 8-queens, Step, and zig-zag MC placement schemes in an 8x8 2D-Mesh.

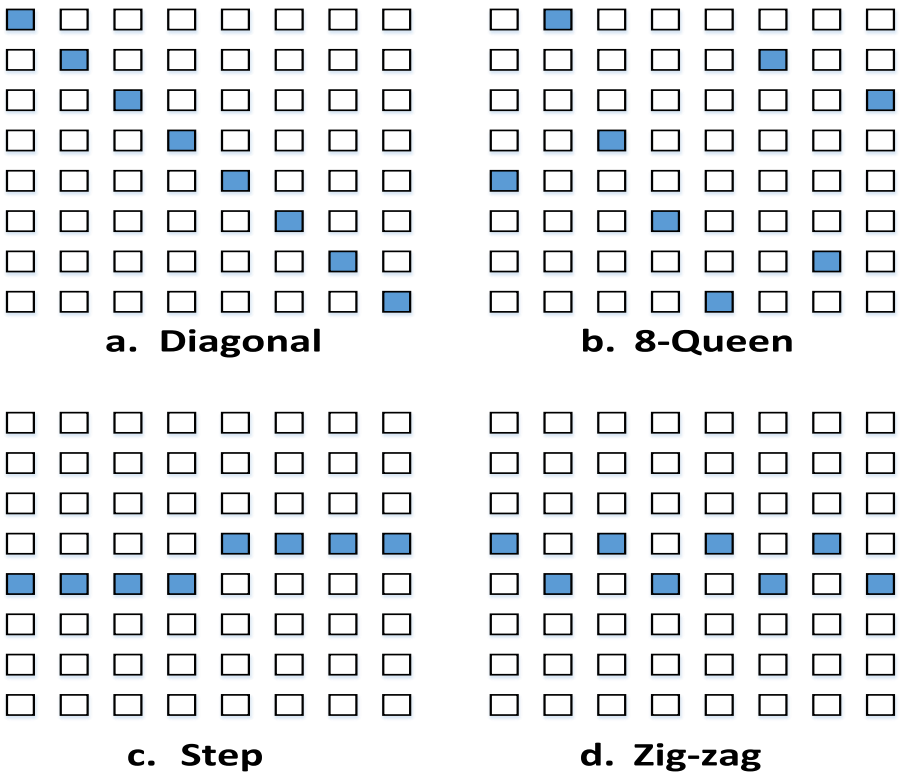

Figure 15. Proposed MC placements [shaded tiles represent MCs]

Chapter 5: Performance Evaluation

In this section, we evaluate schemes proposed in section 4 with the aim of developing a high-performance NoC optimized for GPGPUs. We also analyze the simulation results across a variety of GPGPU benchmarks.

5.1 Methodology

We used GPGPU-Sim [15] simulator to implement the proposed schemes and NoC designs. We used NVIDIA's GTX480 as our target architecture. We used an 8x8 2D-mesh network to connect SMs, caches, and MCs. The baseline NoC is built with a single physical network with two separate VCs for handling request and reply traffic. Table 5.1 shows the system configuration of baseline NoC. We evaluate our schemes with ISPASS [11] and Rodinia [14] benchmarks. Each benchmark is structured to span a range of parallelism and compute patterns.

System Parameters	**Details**
Shader Core	56 cores, 1400 MHz, SIMT width = 8
Memory controller	8 MCs, 924 MHz
Virtual Channels	2 VCs per port per Uni-directional link
VC Buffer size	4 flits per VC
MC Placement	Diamond
Cache	L1 Inst.Cache- 2 KB, L1 Data Cache- 16KB L2 cache- 64KB, Shared Memory- 48KB

Table 1. System Configuration of Baseline NoC

5.2 Performance Analysis

The baseline NoC consists of a single physical network and two logical networks (VCs) with Diamond MC placement as shown in Figure 6(d) in an 8x8 2D mesh. We compared our proposed schemes with Baseline NoC and Bottom MC placement with VC monopolization scheme proposed in [2]. Bottom MC placement causes high network

congestion near the MCs and increases the maximum channel load. To avoid this problem, MCs are sparsely distributed across the network. With distributed MC placement, request and reply traffic are spread across multiple locations on the NoC instead of converging to the bottom row. Among all the existing MC placements proposed in earlier works [1], Diamond MC placement is the winner with an average hop count of 7. The average hop count of existing MC placements along with newly proposed MC placement schemes are listed in Table 2.

MC Placement	**Average Hop Count**
Bottom	8
Edge	8
Top-Bottom	8
Diamond	7
Diagonal	7
8-queen	7
Step	6
Zig-zag	6

Table 2. Average hop count of various MC placements

The existing Bottom MC placement with YX routing and VC monopolization as shown in Figure 10 has a maximum channel load of 12 on the links around MCs. The maximum channel load has been reduced significantly with the new MC placements discussed in section 4. The maximum channel loads on Horizontal and vertical links are shown in the Table 5.3.

MC Placement	**Max. Channel Load**
Bottom with YX and VC monopolization	12
Top-Bottom with XY-YX	10
Diagonal with XY-YX	9
8-queen with XY-YX	9
Zig-zag	6 (8 on Horizontal Links)

Table 3. Maximum Channel Load on Various MC placements

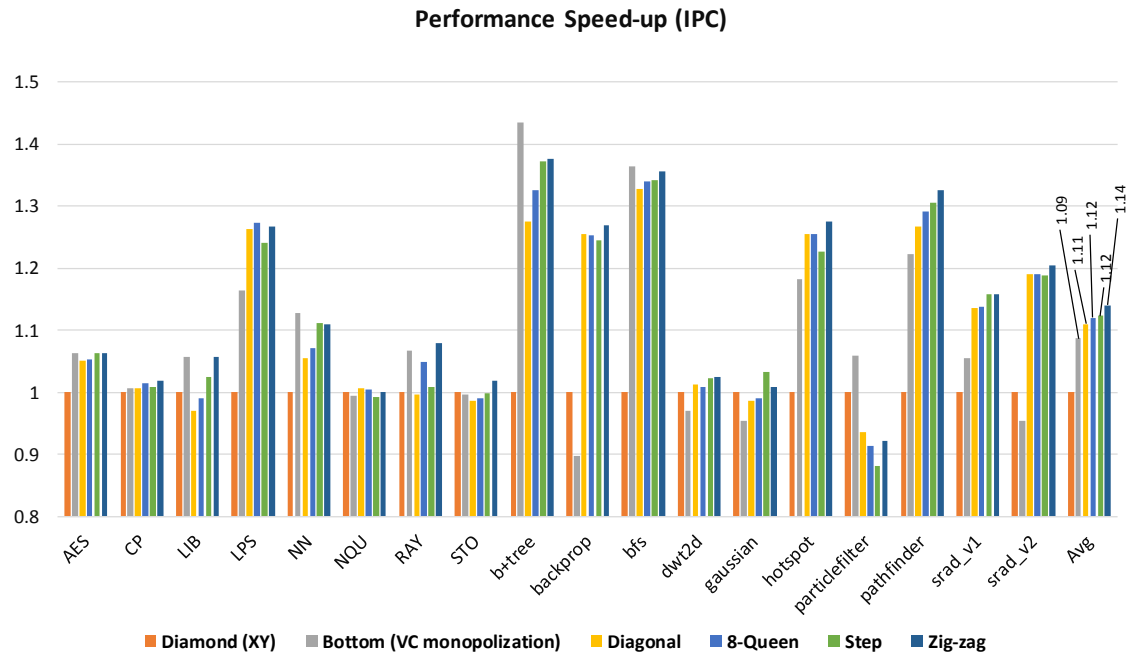

Figure 16. Speed-up (IPC) with different MC placements and VC partitioning schemes

The existing VC monopolization scheme [2] with Bottom MC placement as shown in Figure 10 has a performance improvement of 8% speed-up in terms of IPC compared to baseline NoC. Figure 16 shows the performance in terms of Instructions per Cycle (IPC) across VC monopolization with Bottom MC placement [2], and newly proposed MC placement schemes normalized to baseline NoC.

From the Figure, we can see that the newly proposed MC placement schemes along with XY-YX routing have a significant performance improvement. This is because the distributed request and reply traffic are spread across multiple locations of the on-chip network. Compared to the baseline NoC, the average performance speed-up in terms of IPC is 11%, 12%, 12.5% and 14.1% for the Diagonal, 8-Queen, Zig-zag and Step MC placements respectively.

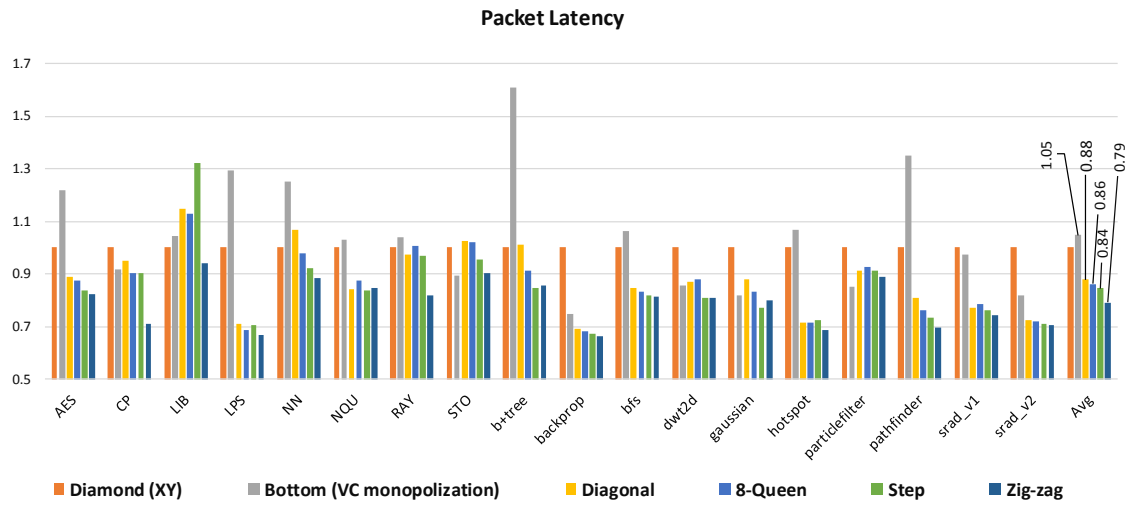

Figure 17. Packet Latency with different MC placements normalized to Baseline NoC

Figure 17 shows the performance in terms of Packet Latency across VC monopolization with Bottom MC placement [2], and newly proposed MC placement schemes normalized to baseline NoC. Packet Latency is given by the sum of Network Latency and Serialization latency associated with storing the packet in the buffer. Network Latency is given by the time for packet to traverse network starting from head flit arriving at the input port of source to the tail flit departing the output port of destination. The Packet latency is reduced by 13%, 14.8%, 16.6% and 21% compared to baseline NoC for the Diagonal, 8-Queen, Zig-zag and Step MC placements respectively.

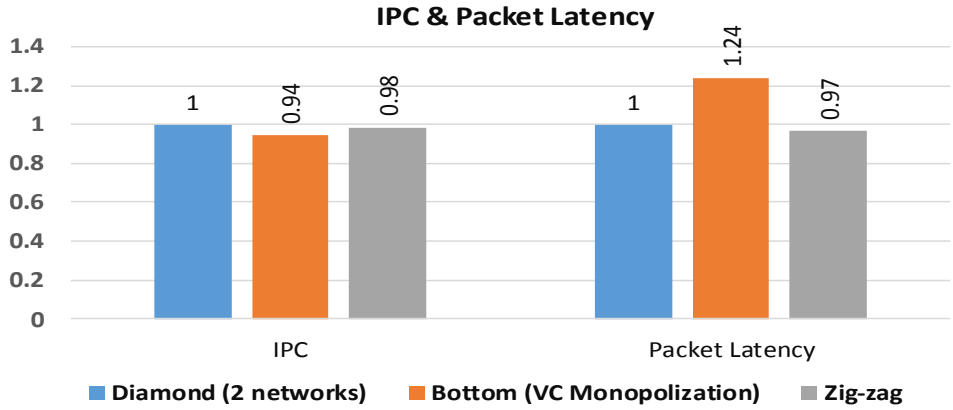

Figure 18. IPC & Packet Latency compared to Diamond with two networks

Figure 18 shows the average speed-up of IPC and reduced latency of Zig-zag MC placement and Bottom MC placement with VC monopolization compared to Diamond MC placement with two dedicated physical networks for request and reply. From the figure, we observe that Zig-zag MC placement degrades system performance (IPC) by only 2% whereas, Bottom MC placement degrades system performance by 6% when compared to having a dedicated physical request and reply networks. In terms of packet latency, zig-zag outperforms Bottom MC placement with VC monopolization and Diamond MC placement with 22% and 3% respectively.

Overall, Zig-zag MC placement has a 14% performance speedup in terms of IPC and 21% reduced latency among all the existing MC placement schemes because of reduced maximum channel load on the links surrounding MCs and minimum hop count. This proves the performance effectiveness of the MC placement schemes and VC partitioning scheme in XY-YX routing described in earlier sections.

Chapter 6: Limitations and Future Work

This study considers only Dimension-order routing across the NoC because of its simplicity and ability to partition VCs, however, Adaptive routing has been improved a lot in recent years. We ran our results on a limited set of benchmarks due to limited resource availability. The entire study has been based on the link contention, max channel load and MC placements to improve the throughput in GPGPUs. This study can be further improved by studying the NoC power optimizations using DSENT simulator [16] due to the reduced average hop count and reduced channel load. We can also extend this study to a higher dimension NoC along with increased set of VCs and asymmetric VC partitioning (allocating more VCs to reply traffic). The additional VCs used to avoid protocol deadlock can affect the critical path of the router which needs to be taken care of. They are several other NoC parameters such as the type of VC allocators, in priority and out priority at the router inputs, link widths etc. which can affect the NoC performance. A detailed survey can be done with the effects of all NoC parameters to improve system throughput of a GPGPU.

Chapter 7: Conclusion

In this work, we analyzed the traffic patterns and network characteristic in GPGPUs under a wide range of benchmarks. We proposed new MC placement schemes like Diagonal,8-queen, Step and Zig-zag which have reduced average hop count compared to existing MC placement schemes. We also analyzed how maximum channel load and link contention around MCs can be reduced by using VC partitioning schemes in XY-YX routing across proposed MC placements. We showed the improved system performance based on VC partitioning in XY-YX routing across proposed MC placement schemes. This work can be further improved by studying how various NoC parameters affect the system performance of GPGPUs.

Bibliography

[1] Abts, Dennis, et al. "Achieving predictable performance through better memory controller placement in many-core CMPs." *ACM SIGARCH Computer Architecture News* 37.3 (2009): 451-461.

[2] Jang, Hyunjun, et al. "Bandwidth-efficient on-chip interconnect designs for GPGPUs." *Proceedings of the 52nd Annual Design Automation Conference*. ACM, 2015.

[3] Kayıran, Onur, et al. "Neither more nor less: optimizing thread-level parallelism for GPGPUs." *Proceedings of the 22nd international conference on Parallel architectures and compilation techniques*. IEEE Press, 2013.

[4] Owens, John D., et al. "Research challenges for on-chip interconnection networks." *IEEE micro* 27.5 (2007): 96-108.

[5] Dally, William J., and Brian Towles. "Route packets, not wires: on-chip interconnection networks." *Design Automation Conference, 2001. Proceedings*. IEEE, 2001.

[6] Benini, Luca, and Giovanni De Micheli. "Networks on chip: a new paradigm for systems on chip design." *Design, Automation and Test in Europe Conference and Exhibition, 2002. Proceedings*. IEEE, 2002.

[7] Dally, William J. *Virtual-channel flow control*. Vol. 18. No. 2SI. ACM, 1990.

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

[9] Kim, Hanjoon, et al. "Providing cost-effective on-chip network bandwidth in GPGPUs." *Computer Design (ICCD), 2012 IEEE 30th International Conference on*. IEEE, 2012.

[10] Yuan, George L., Ali Bakhoda, and Tor M. Aamodt. "Complexity effective memory access scheduling for many-core accelerator architectures." *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*. ACM, 2009.

[11] Bakhoda, Ali, et al. "Analyzing CUDA workloads using a detailed GPU simulator." *Performance Analysis of Systems and Software, 2009. ISPASS 2009. IEEE International Symposium on* (pp. 163-174). IEEE, 2009.

[12] Kumar, Amit, Li-Shiuan Peh, and Niraj K. Jha. "Token flow control." *Proceedings of the 41st annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Computer Society, 2008.

[13] Enright, Natalie, and Li-shiuan Peh. *On-chip networks*. Morgan & Claypool Publishers, 2009.

[14] Che, Shuai, et al. "Rodinia: A benchmark suite for heterogeneous computing." *Workload Characterization, 2009. IISWC 2009. IEEE International Symposium on*. IEEE, 2009.

[15] Bakhoda, Ali, et al. "Analyzing CUDA workloads using a detailed GPU simulator." *Performance Analysis of Systems and Software, 2009. ISPASS 2009. IEEE International Symposium on*. IEEE, 2009.

[16] Sun, Chen, et al. "DSENT-a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling." *Networks on Chip (NoCS), 2012 Sixth IEEE/ACM International Symposium on*. IEEE, 2012.

[17] Dally, William J., and Brian Towles. "Route packets, not wires: on-chip interconnection networks." *Design Automation Conference, 2001. Proceedings*. IEEE, 2001.

[18] Dally, William J. "Virtual-channel flow control." *IEEE Transactions on Parallel and Distributed systems* 3.2 (1992): 194-205.

[19] Dally, William J., and Charles L. Seitz. "Deadlock-free message routing in multiprocessor interconnection networks." *IEEE Transactions on computers*100.5 (1987): 547-553.

[20] Ziabari, Amir Kavyan, et al. "Asymmetric NoC Architectures for GPU Systems." Proceedings of the 9th International Symposium on Networks-on-Chip. ACM, 2015.

[21] Balfour, James, and William J. Dally. "Design tradeoffs for tiled CMP on-chip networks." Proceedings of the 20th annual international conference on Supercomputing. ACM, 2006.

[22] Grot, Boris, et al. "Express cube topologies for on-chip interconnects." 2009 IEEE 15th International Symposium on High Performance Computer Architecture. IEEE, 2009.

[23] Kim, John, James Balfour, and William Dally. "Flattened butterfly topology for on-chip networks." Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture. IEEE Computer Society, 2007.

[24] Gratz, Paul, Boris Grot, and Stephen W. Keckler. "Regional congestion awareness for load balance in networks-on-chip." 2008 IEEE 14th International Symposium on High Performance Computer Architecture. IEEE, 2008.

[25] Hayenga, Mitchell, Natalie Enright Jerger, and Mikko Lipasti. "Scarab: A single cycle adaptive routing and bufferless network." Proceedings of the 42nd annual IEEE/ACM international symposium on microarchitecture. ACM, 2009.

[26] Kinsy, Michel A., et al. Application-aware deadlock-free oblivious routing. Vol. 37. No. 3. ACM, 2009.

[27] Singh, Arjun, et al. "GOAL: a load-balanced adaptive routing algorithm for torus networks." ACM SIGARCH Computer Architecture News 31.2 (2003): 194-205.

[28] Chen, Lizhong, and Timothy M. Pinkston. "Worm-bubble flow control." High Performance Computer Architecture (HPCA2013), 2013 IEEE 19th International Symposium on. IEEE, 2013.

[29] Michelogiannakis, George, and William J. Dally. "Elastic buffer flow control for on-chip networks." IEEE Transactions on Computers 62.2 (2013): 295-309.

[30] Peh, Li-Shiuan, and William J. Dally. "Flit-reservation flow control." High-Performance Computer Architecture, 2000. HPCA-6. Proceedings. Sixth International Symposium on. IEEE, 2000.

[31] Jeloka, Supreet, et al. "Hi-Rise: A high-radix switch for 3D integration with single-cycle arbitration." 2014 47th Annual IEEE/ACM International Symposium on Microarchitecture. IEEE, 2014.

[32] Kim, John, et al. "Microarchitecture of a high-radix router." ACM SIGARCH Computer Architecture News. Vol. 33. No. 2. IEEE Computer Society, 2005.

[33] Mullins, Robert, Andrew West, and Simon Moore. "Low-latency virtual-channel routers for on-chip networks." ACM SIGARCH Computer Architecture News. Vol. 32. No. 2. IEEE Computer Society, 2004.

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

[35] Demir, Yigit, and Nikos Hardavellas. "SLaC: Stage laser control for a flattened butterfly network." 2016 IEEE International Symposium on High Performance Computer Architecture (HPCA). IEEE, 2016.