Heterogeneous NoC Router Architecture

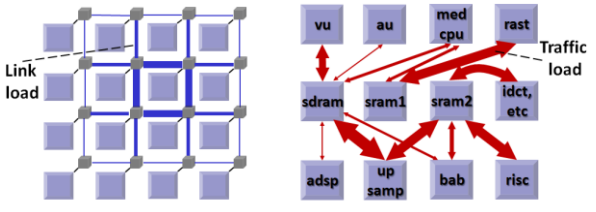
Yaniv Ben-Itzhak, Israel Cidon, Avinoam Kolodny, Michael Shabun, and Nir Shmuel

Abstract—We introduce a novel heterogeneous NoC router architecture, supporting different link bandwidths and different number of virtual channels (VCs) per unidirectional port. The NoC router is based on shared-buffer architecture and has the advantages of ingress and egress bandwidth decoupling, and better performance as compared with input-buffer router architecture. We present the challenges facing the design of such heterogeneous NoC router, and describe how this router architecture addresses them. We introduce and formally prove a novel approach that reduces the number of required middle shared-buffers without affecting the performance of the router. In comparison with an optimal input-buffer homogeneous router, our NoC router improves saturation throughput by 6%-47% for standard traffic patterns. The router achieves significant run-time improvement for NoC-based CMP running PARSEC benchmarks. It offers better scalability, area, and power reduction of ~15%-~60%, for NoC based CMPs of size 4x4 up to 16x16, as compared with optimal input-buffer homogeneous and heterogeneous routers.

Index Terms—Interconnection architectures, Network connectivity chips, Routers

1 Introduction

SOC and CMP designs use Networks-on-Chip (NoC) to support a variety of inter-module communication bandwidth and latency requirements. In the case of SoC, the traffic and timing requirements are typically known at design time, and certain specific links are designated to carry large data streams. In CMPs, the requirements depend on the characteristics of executed software. In both cases, NoC performance requirements are usually heterogeneous in terms of particular module-to-module bandwidth and delay, as traffic is usually distributed unevenly across the chip. Fig. 1 presents two well-known examples, where heterogeneous loads are common in SoCs. The first, uniform traffic pattern over a mesh NoC [1, 2] (Fig. 1(a)) exhibits higher loads over NoC links in the center of the chip as compared to NoC links in the periphery. The second, MPEG4 core communication pattern [3] (Fig. 1(b)) exemplifies traffic requirements that are heterogeneous between different modules.



(a) Uniform Traffic Pattern over Mesh NoC **(b) MPEG4 core communication pattern [3]**

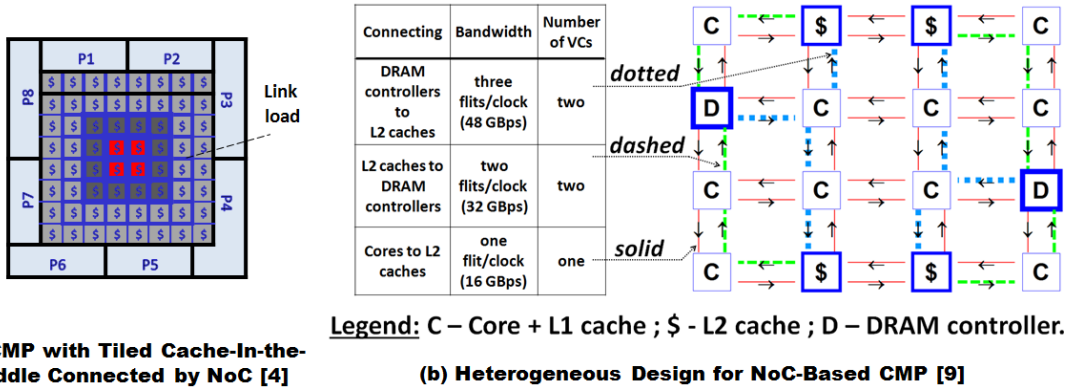
Fig. 1. Heterogeneous traffic loads examples (The link thickness corresponds to its load).

Fig. 2 demonstrates two examples for heterogeneous loads in different CMP architectures. The first, CMP with a tiled cache in the middle architecture (which employs banked DNUCA connected by a NoC [4] (Fig. 2(a)) presents higher loads over NoC links in the center of the banked cache since these banks are accessed the most. The second, a 4x4 NoC-based CMP, which consists of cores with L1-cache, L2 shared-caches and DRAM controllers (Fig. 2(b)). The NoC exhibits three different link loads; hence the optimal heterogeneous NoC (in terms of run-time) consists of three types of links: (1) Links connecting the DRAM controllers and the L2 caches which handle read misses; (2) Links connecting the L2 caches to the DRAM controllers which handle block replacements during miss in the L2 caches; (3) Links connecting the cores to the L2 caches, which handle L1 miss [9].

Other typical sources of traffic heterogeneity are traffic destined to external memory, bus or network interfaces, or to specialized internal computation units. As traffic requirements are generally heterogeneous, it is also expected that the optimal NoC designed to support these requirements should also be heterogeneous in terms of link capacities and number of virtual channels (VCs) for each unidirectional port.

Guz et al. [5] present delay analysis for a NoC router with different link bandwidths, and introduce the necessity for a heterogeneous NoC. However, to the best of our knowledge, previous NoC router architecture proposals, (e.g. [6, 7, 8]) are limited to homogeneous NoCs, assuming that homogeneity simplifies the design of the router's components (e.g., switch/VC allocators, crossbar, etc.). Trading-off efficiency for simplicity and regularity, homogeneous NoCs waste performance, power and area as compared with heterogeneous NoCs [9]. We investigate heterogeneous NoCs, to achieve better efficiency of network resources, despite the added complexity associated with irregularity of network components.

- *Yaniv Ben-Itzhak is with the Electrical Engineering Department, Technion – Israel Institue of Technology. E-mail: yanivbi@tx.technion.ac.il.*
- *Israel Cidon is with the Electrical Engineering Department, Technion – Israel Institue of Technology. E-mail: cidon@ee.technion.ac.il.*
- *Avinoam Kolodny is with the Electrical Engineering Department, Technion – Israel Institue of Technology. E-mail: kolodny@ee.technion.ac.il.*
- *Michael Shabun is with the Electrical Engineering Department, Technion – Israel Institue of Technology. E-mail: m.shabun@gmail.com*
- *Nir Shmuel is with the Electrical Engineering Department, Technion – Israel Institue of Technology. E-mail: nir.shmuel123@gmail.com*



(a) CMP with Tiled Cache-In-the-Middle Connected by NoC [4]

(b) Heterogeneous Design for NoC-Based CMP [9]

Fig. 2. Rationale for using Heterogeneous NoC-based CMP.

For instance, the optimized heterogeneous NoC presented in Fig. 2(b) reduces the area as compared with a homogeneous NoC, for the same run-time [9]. Hence, heterogeneous NoC can better utilize the NoC resources (i.e., total link bandwidth and total number of virtual channels) and consequently achieve better performance, lower power consumption and smaller area.

In this paper, we propose a novel heterogeneous NoC router architecture. We exploit the shared-buffer technique (also known as Space-Time-Space (STS) [10]) in order to handle the router's port bandwidth heterogeneity. Unlike homogeneous NoC routers, which usually schedule a single flit per clock cycle for each output port, a heterogeneous NoC router is required to schedule a different number of flits (for each ingress and egress port) per clock cycle. It should take into account the different egress link bandwidth while maintaining the flits order in each packet, such that the number of flits destined to a given output port o cannot exceed the bandwidth of egress link o. We also present a novel approach that reduces the number of required middle shared-buffers without affecting the router performance.

The paper is structured as follows: Section 2 surveys related works on heterogeneous NoCs, and shared-buffer routers. Section 3 presents the architectural challenges and the general concepts of the router. Section 4 presents the architectural options for heterogeneous NoC routers, and justifies the shared-buffer architecture as our choice. Section 5 presents the detailed structure of the router, and describes a novel approach to reduce the number of shared-buffers. Section 6 presents the advantages and disadvantages of the proposed architecture. Section 7 presents implementation details of the router. Section 8 presents a detailed performance, area and power evaluation, and demonstrates the architecture scalability. Section 9 discusses different issues and presents future direction of this work. Section 10 concluders the paper.

2 Related Work

Previous works partly address the heterogeneous NoC problem. QNoC [5] has introduced the importance of different link capacities in NoCs; however, the authors don't present a router architecture to support their statement. Other works focus on non-uniform number of VCs (with uniform link capacities), e.g. [11, 12]; irregular topologies composed of homogeneous NoC routers, e.g., [13, 14]; or NoCs composed of a small set of predefined routers with fixed bandwidth and a fixed number of VCs for all ports [2, 15, 16, 17], which focus on design and placement rather on the router architecture. Mishra et al. in [2] present a NoC architecture with limited heterogeneous capability (i.e., ability to choose between a small router (link width of 128 bit) and a big router (link width of 256bit)). The authors in [9] have introduced an optimal design methodology for heterogeneous NoC in terms of both different link capacities and number of VCs. However, to the best of our knowledge, NoC router architecture to support such heterogeneity (in terms of link capacity and number of VCs) has not been published.

Several previous works present a homogeneous NoC router, which is based on shared-buffer architecture. Ramanujam et al. in [18] present a shared-buffer NoC router, which is based on STS architecture. Another work by Tran et al. [19] presents a NoC router architecture with shared-queues (which is not STS architecture). However, these previous works don't target heterogeneity of the router in terms of link capacity and number of VCs per unidirectional ports.

3 Architectural Challenges of a Heterogeneous Router

Two main modifications are required in a NoC router to support different link capacities and different number of VCs for each unidirectional link. The first modification is to allocate and arbitrate flits to a different number of VCs for each port. This is relatively straightforward. The second modification, which is much more complex, is to match the different ingress and egress bandwidths of unidirectional ports. The router needs to utilize a high bandwidth egress link by concurrently transmitting flits from several low bandwidth ingress links (over different VCs) and to distribute flits from a single high bandwidth ingress link (and from different VCs) to several low bandwidth egress links.

There are two main options to implement different bandwidth links: The first option is to use a different frequency for each unidirectional port, while using the same link widths. The second option is to use different link

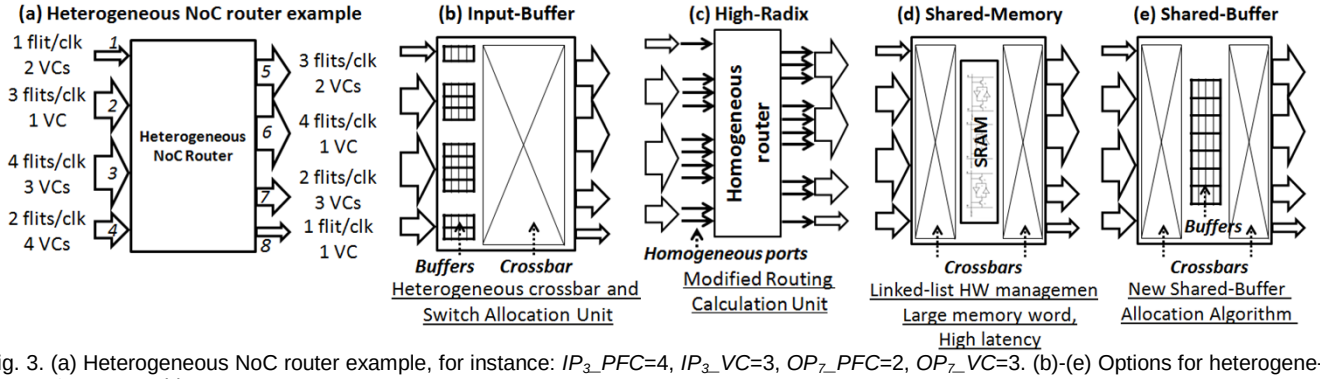


Fig. 3. (a) Heterogeneous NoC router example, for instance: IP_3_PFC=4, IP_3_VC=3, OP_7_PFC=2, OP_7_VC=3. (b)-(e) Options for heterogeneous NoC router architecture.

widths, while keeping the same frequency for all unidirectional ports. The first option offers an ideal rate matching between the unidirectional ports, since each port uses its own frequency, which is determined according to the bandwidth of its connected link. However, this approach is less practical as it requires several predefined clock domains within a single router, which limits the bandwidth heterogeneity. It also requires the router to synchronize between the different clocks domains of the ports. Finally, the maximum link frequency is limited by the silicon technology.

We present a heterogeneous NoC router architecture based on the second option; i.e., different link widths while keeping the router's frequency fixed. Nevertheless, our proposed architecture can be used also for links, which are implemented using the first option. Serial-to-parallel converters can be used in order to store ingress flits to different input buffer slots at each link clock cycle; and parallel-to-serial converters can be used in order to transmit several flits in TDM fashion according to the link frequency.

We define the number of maximum concurrent transferred flits at each clock cycle for each unidirectional input or output port i as the *Input* or *Output Port – Parallel Flits per Cycle i* (IP_i_PFC or OP_i_PFC), respectively. Moreover, each unidirectional (ingress or egress) port j has a different number of VCs, which is noted by *Input* or *Output Port VC j* (IP_j_VC or OP_j_VC). For instance, the link width of egress link o, $LinkWidth_o$, is determined by the number of flits which can be concurrently transmitted (at each clock cycle) over the link, OP_o_PFC.

$$LinkWidth_o = OP_o\_PFC \cdot FlitSize \quad (1)$$

Hence, the bandwidth of egress link o, BW_o, is:

$$BW_o = LinkWidth_o \cdot Frequency \quad (2)$$

Furthermore, a variable number of flits are being received and transmitted at each clock cycle over different input and output ports. Therefore, the router's logic (i.e., VC allocator, switch allocator, VC arbiter, etc…) should support splitting ingress flits from an input port to several output ports according to the flits' destinations. It should also support merging of flits from different input ports that are destined to the same output port.

4 The Heterogeneous NoC Router Architecture Options

In this section, we present several alternatives for a router in a heterogeneous NoC (see Fig. 3(a) for example). Input-buffer (Fig. 3(b)); High-radix (Fig. 3(c)); Shared-memory (Fig. 3(d)); and Shared-buffer (Fig. 3(e)). We present how one can utilize these router architectures to support heterogeneity, and compare them. Note that the choice of routing algorithm might affect the heterogeneous design (in terms of assigning bandwidth and number of VCs per unidirectional-port), but not the architecture choice. Therefore, any routing scheme (include adaptive routing) can be used with any of the following router architectures.

4.1. Input-Buffer Heterogeneous NoC Router Architecture

The input-buffer NoC router architecture is commonly used [6, 7]. This router architecture, illustrated in Fig. 3(b), can be modified to support heterogeneity. A different number of arbiters for each output port is required in order to allow transmission of a different number of concurrent flits per clock cycle. In addition, in order to support the connectivity of input and output ports, the crossbar should consist of different input and output link widths. It should allow combining flits from several low bandwidth input ports into a high bandwidth output port, and vice versa.

Each input port receives a different number of concurrent flits at each clock cycle. Therefore, the input-buffers of each input port should have different write and read rates (i.e., the number of written and read flits at each clock cycle respectively). It is clear that the write rate of the input-buffers must be equal to its maximum received number of concurrent flits (IP_PFC). However, the read rate of each input-buffer can vary between two extreme options.

The first option is to use input-buffers with a read rate that is equal to their IP_PFC. This option uses input-buffers with the minimal read rate such that the input-buffers won't be saturated. However, this option results in low burst handling capability if the flits are destined to an output port with higher egress link bandwidth. Consider the following case: a large number of flits stored in a

Table 1. Comparison between heterogeneous NoC router alternatives (The cell darkness corresponds to its disadvantage)

Criteria	**Input-Buffer** Read Rate = Input port bandwidth	**Input-Buffer** Read Rate = Maximum output port bandwidth	**High-Radix**	**Shared-Memory**	**Shared-Buffer (STS)**
Latency	High, due to low burst handling capability (-)	Low (+)	High, due to limited bandwidth per packet (-)	High, due to shared memory word size and linked-list management (-)	Low (+)
Throughput	Low, due to low burst handling capability (-)	Medium	Low/Medium, depends on number of concurrent packets (-)	High (+)	High (+)
Required Resources	Low buffer and crossbar resources	Very high buffer and crossbar resources	High-radix crossbar. Modified routing calculation unit.	Linked-list hardware management over-head. High BW shared memory.	Additional crossbar

given input port, are destined to an output port with a higher egress link bandwidth. In this case, the router won't be able to fully utilize the given output port, since the read-rate of the corresponding input-port is lower than the bandwidth of the given output port.

The second option, which targets the worst-case scenario, uses for every input-buffer a read rate of the maximum bandwidth output port. This option achieves full utilization of all output ports at the expense of higher buffer and crossbar costs as compared to the first option. Since the read rate of all input-buffers is determined by the highest bandwidth output port, the total read rate of the input-buffers can't be simultaneously utilized (unless all output ports have the same bandwidth).

In summary, the heterogeneous input-buffer router architecture is inefficient for decoupling the different ingress and egress link bandwidths, unless large buffer and crossbar resources are used.

4.2. High-Radix Heterogeneous NoC Router Architecture

High-radix router (i.e., router with many homogeneous ports) [20] is another option for heterogeneous NoCs. Fig. 3(c) illustrates how to exploit a high-radix homogenous NoC router architecture for heterogeneous NoCs. In order to maintain the heterogeneity, one can allocate a different number of ports for each direction, which results in a different aggregate bandwidth. The heterogeneity benefit of such router is its total throughput, since several packets can be transmitted towards the same direction through different ports. However, each packet can be transmitted over a single port of the router and therefore only a single flit of each packet can be transmitted at each clock cycle. Hence, the packet latency may be longer as compared with the heterogeneous link approach presented in Section 3, which allows concurrent transmission of flits from the same packet at each clock cycle.

4.3. Shared Memory Heterogeneous NoC Router Architecture

In shared-memory routers [10, 21] (Fig. 3(d)), all input and output ports have access to a common memory. In every clock cycle, all input ports can store their incoming flits and all output ports can retrieve their outgoing flits. Since the flits are read from a shared-memory to the output ports, the total read-rate of the shared-memory is determined by the total output ports' bandwidth. Similarly, the total write-rate of the shared-memory is determined by the total input ports' bandwidth. Therefore, it seems that shared-memory router architecture can offer better rate matching between input ports and output ports as compared to input-buffer router architecture. However, we show that in spite of these promising capabilities, this router might not be a good choice for heterogeneous NoCs due to high internal latency and high hardware overhead.

Linked-list based shared-memory (SRAM) NoC router has been presented in [22]. All flits that are destined to the same output port are linked together in the same logical queue. However, linked-list implementation requires five memory access cycles for each read or write memory operation [10], which limits the maximum ingress and egress bandwidth of the router. Moreover, the hardware overhead for managing linked lists in a shared-memory is too costly for NoC routers [23].

Heterogeneous shared-memory based router should have high bandwidth in order to store all flits destined to the same output port (where each input port receives a different number of concurrent flits at each clock cycle). SRAMs are known to have relatively low read and write bandwidth. Hence, there are two main solutions to overcome it and achieve high read and write bandwidth. The first solution is the usage of a large word-size such that the total input ports' bandwidth is supported. Since the memory word should be built from several ingress flits (destined to the same output port), it is necessary to wait for a sufficient number of flits to utilize the word size. This adds latency and buffers. Alternately, one can use a non-full SRAM word (i.e., contains less flits) in order to reduce latency, at the expense of lower actual memory bandwidth and usage. Another solution is using a multi-port SRAM, such that flits from each input port are written to each logical queue. However, the SRAM layout size increases quadratically with the number of ports, leading to higher latency and higher power consumption.

4.4. Shared-Buffer Heterogeneous NoC Router Architecture

Shared-buffer routers (also known as Space-Time-Space (STS) routers [10]) consist of separate buffers which are shared among all input and output ports via two crossbars (Fig. 3(e)). The shared-buffers store the ingress flits from all the input ports. Each shared-buffer stores a single flit in each cell. Hence, unlike the shared-memory router architecture, the shared-buffer router architecture eliminates the need to manage linked-lists and to wait for sufficient number of flits to utilize the memory word. Therefore, the shared-buffer router achieves lower latency and higher throughput than the shared-memory router. Therefore, shared-buffer architecture is our choice for heterogeneous NoC routers.

In the sequel, we describe our proposed shared-buffer heterogeneous NoC router architecture in detail, and show that our proposed router architecture offers simple ingress and egress bandwidths decoupling, and better performance. We also point out the router architecture consideration, its advantages and disadvantages.

4.5. Comparison Summary

Table 1 summarizes the comparison between the different architecture options for the heterogeneous NoC router. The input-buffer router architecture has low capability in decoupling the different ingress and egress bandwidths. High-radix router architecture is simple, but offers high packet latency and low throughput. Shared-memory router architecture offers better ingress and egress bandwidth decoupling but incurs high latency. Moreover, linked-lists implementation has significant memory operations overhead and it is infeasible for NoC router.

On the other hand, shared-buffer router architecture offers superior ingress and egress bandwidth decoupling, lower latency and higher throughput, as we present in the sequel.

5 The Shared-Buffer Heterogeneous Router Architecture

5.1. Overview of the Shared-Buffer Heterogeneous Router

Fig. 4 illustrates the definitions and architecture of the heterogeneous NoC router approach. Each cell in the shared-buffers corresponds to a different time-slot. The time-slot of the cell determines the number of clock cycles until the flit stored in this cell will be transmitted to the output port via the second crossbar. The destination of each incoming flit is first inspected. Next, the incoming flits are "time-stamped" (i.e., assigned a time-slot), which determines departure time of the flit from the router. The flits are then assigned to one of the shared-buffers, given that there are no timing conflicts with other flits already stored in this shared-buffer; such that, the number of flits destined to a specific output port j at each time-slot do not exceed its OP_j_PFC. The assigned flits are transmitted to the shared-buffers via the first crossbar. Finally, the flits in the shared-buffers progress (to the right in Fig. 4)

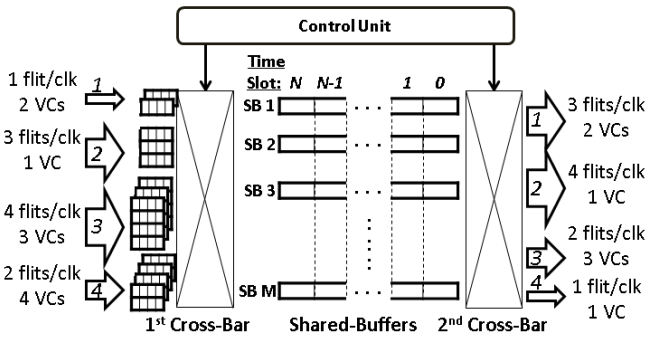


Fig. 4. The architecture of the shared-buffer heterogeneous NoC router. Demonstrated for: IP_3_PFC=4, IP_3_VC=3,OP_3_PFC=2, OP_3_VC=3.

at each clock cycle and the flits at the head of the shared-buffers are transmitted to their corresponding output ports.

Unlike previous shared-buffer router architecture proposals (e.g., [18]), our shared-buffer router architecture should also take into account the heterogeneity of the input and output ports while maintaining the order of the flits within a packet. Flits can be concurrently transmitted over several VCs. At each cycle, one flit can be read from and written to each shared-buffer. Also, several flits can be written at each cycle to a shared-buffer into different time-slots (defined as *shared-buffer write speed-up* in section 5.3).

The shared-buffer heterogeneous router uses credit-based flit-level flow control. Flow-control is applied on a flit-by-flit basis, advancing each flit from an input queue towards its assigned time-compatible shared-buffer and ultimately to the egress link. Flits are time stamped and placed into a shared-buffer only when the router has credits for the corresponding output port and assigned VC.

There are some straightforward advantages of this router architecture, which can be indicated at this point:

1) The shared-buffers decouple input and output port bandwidths, as any flit can acquire any shared-buffer.
2) The shared-buffers offer better buffer utilization as they are shared among all ports.
3) The shared-buffers provide path diversity between the input and output ports within a router. Hence, the router can tolerate defective shared-buffers (by disabling allocation of flits to that defective shared-buffer).

5.2. The Shared-Buffer Heterogeneous NoC Router Pipeline

Fig. 5(a) presents the router data-flow and pipeline of the router. First, the incoming flits are written into the input-buffers in the Buffer Write stage. The input-buffers are segmented according to the number of VCs of each input port. When a head flit arrives at a VC, the Route Computation stage determines the output port of the flit based on its coordinates.

In the next stage, the head-flits continue to the Virtual Channel Allocation stage, which arbitrates for free virtual channels at the input of the next-hop router by managing

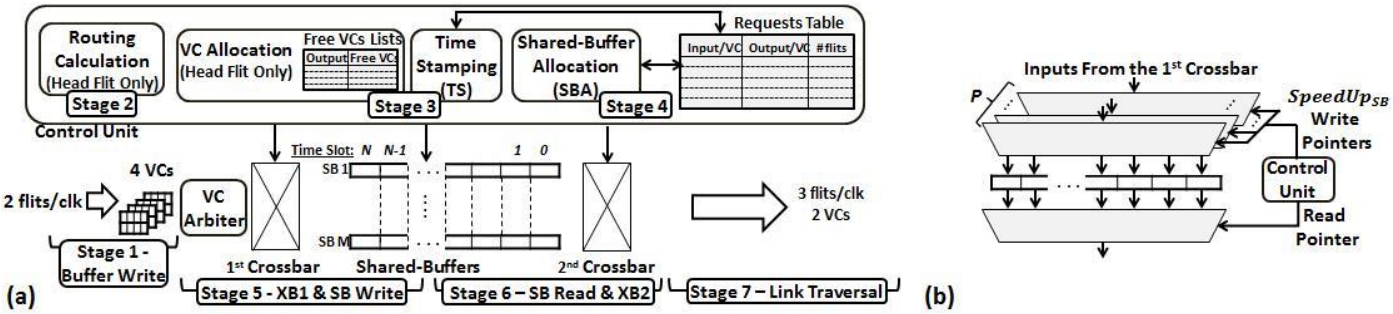


Fig. 5. (a) Data flow of the shared-buffer heterogeneous. (b) $SpeedUp_{SB}$ *SP-PIFO queue* scheme.

a free VCs list for each output port.

The Time-Stamping (*TS*) and Shared-Buffers Allocation (*SBA*) stages are responsible for assigning the ingress flits into the shared-buffers by resolving two types of conflicts: *arrival conflict* (by the *SBA* stage) and *departure conflict* (by the *TS* stage) [18], as explained below.

The router holds a request table that indicates the number of ingress flits stored in the input-buffers for each input port and VC. The *TS* stage assigns time-slots in a cyclical fashion. It starts from an initial input port and chooses a winning VC for each input port (according to the VC arbitration. e.g., winner-takes-all, round-robin). For each winning VC, the *TS* unit assigns the earliest possible departure times for at most IP_i_PFC flits from input port *i*. The assigned departure time of each flit can be different, while maintaining flits order. For instance, in any case that $IP_i\_PFC > OP_j\_PFC$ (assuming in-port *i* send flits to out-port *j*), the *TS* spreads the ingress flits over several time slots according to IP_i_PFC and OP_j_PFC. Finally, in order to preserve fairness between the input ports, the initial input port is incremented in each clock cycle. Moreover, in order to avoid the case that all shared-buffer slots are occupied by a single flow, the *TS* unit reserves a minimum number of shared-buffer slots for each input port.

The router architecture allows several flits (of the same packet or different packets), which are destined to the same output port, to be transmitted concurrently (over a single VC or multiple VCs, respectively) according to the egress link bandwidth (*OP_PFC*). Therefore, the *TS* unit should take into account the bandwidth of the requested output port when assigning the time-slots. Hence, the unit assigns a given time-slot t to an ingress flit destined to a given output port o only if the number of flits already stored in the shared-buffers with the same time-slot t and the same destination o is lower than OP_o_PFC. This is known as resolving the *departure conflict* [24].

In the *SBA* stage, the flits that were assigned to a time-slot in the previous stage are assigned to a specific shared-buffer. The *SBA* is responsible to maintain the order of flits, which belong to the same packet. In cases where several flits of the same packet have been assigned the same time-slot, this unit assigns the first available shared-buffer (i.e., with the lowest index) to the first flit and so forth. Moreover, this unit should take into account the write constraint of the shared-buffers; i.e., how many flits can be written simultaneously to each shared-buffer at each clock cycle (explained in section 5.3). This is also known as resolving the *arrival conflict*. Flits which didn't succeed to get shared-buffer allocation due to conflicts re-enter the *TS* stage.

Next, the flits proceed to the fifth pipeline stage. They traverse the first crossbar (*XB1*) and written to their assigned shared-buffer (*SB Write*) in their assigned time-slot. All the flits stored in time slot *i* are moved into slot *i*-1 (Fig. 4). Note that, the ingres flits are stored in input-buffers till the *XB1* stage (i.e., during *RC*, *VCA* and *TS* and *SBA* pipeline stages).

The flits stored in time-slot 0 are read from the shared-buffers (*SB Read*) and traverse the second crossbar (*XB2*), each flit towards its corresponding output port. The number of flits which traverse towards each output port is determined according to the egress link's bandwidth (*OP_PFC*). Finally, in the link traversal stage, the flits are transmitted through the egress link towards the downstream router.

This architecture uses more pipeline stages as compared with input-buffer architecture. Hence, the credit round trip time is increased. Nevertheless, we discuss how this drawback can be mitigated in section 6.3; and we demonstrate that this architecture outperforms input-buffer routers, due to its lower internal blocking, in section 8.

5.3. Reducing the Number of Required Shared-Buffers

We present a novel approach to reduce the number of shared-buffers, which are required to guarantee conflict-free router (i.e., no *arrival* and *departure conflicts* occur).

The *TS* and *SBA* stages are used to resolve the *arrival* and *departure conflicts* [18]. Previous work assumed that only a single flit can be written to each shared-buffer at each cycle (e.g., [18]); hence, an ingress flit might have *arrival conflict* with at most $\sum_i IP_i\_PFC - 1$ other flits. In addition, it has a *departure conflict* with all other flits that have the same time-slot; hence, at most $\sum_i OP_i\_PFC - 1$ other flits. Therefore, by the pigeonhole principle [24], both conflicts can always be resolved if the number of shared-buffers, SB, satisfies:

$$SB \geq \sum_i IP_i\_PFC + \sum_i OP_i\_PFC - 1 \qquad (3)$$

However, as opposed to homogeneous NoC routers (in which $IP_{i_{PFC}} = 1$ and $OP_{i_{PFC}} = 1, \forall i$), heterogeneous NoC routers consists of $IP_i\_PFC \geq 1$ and $OP_i\_PFC \geq 1$. There-

fore, the limitation of equation (3) for a conflict-free router may be impractical in terms of power and area considerations, even for heterogeneous NoC router with relatively low ingress and egress bandwidths. Therefore, we present a novel approach that reduces the required number of shared-buffers in equation (3).

Definitions:

1) *shared-buffer write speed-up:* The number of flits that can be written at each clock cycle to each shared-buffer into different time-slots. Also noted by $SpeedUp_{SB}$ (≥ 1).

2) $SpeedUp_{SB}$ *SP-PIFO queue:* Up to $SpeedUp_{SB}$ arriving flits are placed at (or, "pushed-in" to) arbitrary free locations in the queue (through $SpeedUp_{SB}$ write-ports); the relative ordering of flits in the queue is preserved; and flits depart from the head of the queue (Fig. 5(b)).

In contrast with previous works, we use *shared-buffer write speed-up* that can be higher than one. Therefore, we use $SpeedUp_{SB}$ *SP-PIFO queues* for the shared-buffers. As mentioned in section 5.2, the *SBA* stage can allocate several incoming flits (up to *shared-buffer write speed-up*), which have different time slots, into the same shared-buffer.

Theorem 1: A heterogeneous shared-buffer router with a given *shared-buffer write speed-up*, $SpeedUp_{SB}$, is conflict-free if the number of shared-buffers, SB, satisfies:

$$SB \geq \left\lceil \frac{\sum_i IP_i\_PFC - SpeedUp_{SB}}{SpeedUp_{SB}} \right\rceil + \sum_i OP_i\_PFC \qquad (4)$$

Proof: An ingress flit can be stored together with $SpeedUp_{SB} - 1$ other flits in the same shared-buffer. Therefore, an ingress flit might have *arrival conflict* with at most $\sum_i IP_i\_PFC - 1 - (SpeedUp_{SB} - 1)$ other ingress flits, which can be written into $\left\lceil \frac{\sum_i IP_i\_PFC - SpeedUp_{SB}}{SpeedUp_{SB}} \right\rceil$ different shared-buffers. Furthermore, the ingress flit has *departure conflict* with at most $\sum_i OP_i\_PFC - 1$ other flits. By the pigeonhole principle, we get that the conflicts can always be solved if the condition in equation (4) holds. ■

Motivation: The motivation for speeding-up the shared-buffer write is as follows. The required number of shared-buffers decreases as the $SpeedUp_{SB}$ increases, which in turn requires to increase the first crossbar and the write-ports number at each shared-buffer (Fig. 5). However, the power and area overhead introduced by these additional resources is much lower compared with adding more buffers. Buffers are known to be the largest NoC's power and area consumers [25, 26, 27, 28, 29, 30]. The router's area and power can be reduced by increasing the $SpeedUp_{SB}$; thus using fewer shared-buffers (4).

Our evaluation for equation (4) shows that the highest marginal buffer savings is achieved by increasing the *shared-buffer write speed-up* from one to two. For instance, Fig. 6 presents that the required number of shared-buffers of a given router with $\sum_i IP_i\_PFC = \sum_i OP_i\_PFC = 8$ [flits/clock] is reduced by 27% when increasing from one to two, then by 7% from two to three, etc. The largest buffer saving is 47% for *shared-buffer writ e speed-up* equal to

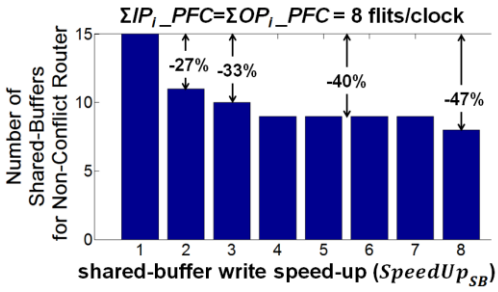


Fig. 6. Required shared-buffers for conflict-free router (and buffer savings) versus *shared-buffer write speed-up* .

eight. When the condition in (4) does not hold, conflicts between ingress flits might occur. However, the number of conflicts (and the probability for such conflicts) is reduced as the *shared-buffer write speed-up* is increased. Finally, in order to allow full utilization of all egress links, one can bound the number of shared-buffers, *SB*, to:

$$SB \geq \sum_i OP_i\_PFC \qquad (5)$$

In summary, if the condition described in (5) holds, the router may achieve full utilization of its entire egress links (when no conflicts occur). If the condition in (4) holds, the router is also conflict-free. Furthermore, when $SpeedUp_{SB} = \sum_i IP_i\_PFC$ and $SB = \sum_i OP_i\_PFC$, the router can mimic an output-buffer router by assigning OP_i_PFC shared-buffers to each output port *i*.

6 Advantages and Disadvantages of the Heterogeneous Shared-Buffer Router

6.1. The Performance Advantage

Our shared-buffer router architecture deals better with output port blocking compared to the heterogeneous input-buffer router architecture presented in section 4.1. The shared-buffer router copes with blocking by scheduling ingress flits over several clock-cycles, such that the flits are assigned to the same time-slot. However, input-buffer router copes with blocking at each clock-cycle separately; i.e., ingress flits can be transmitted together towards the output port only if they were scheduled at the same clock-cycle. In case that the read rate of the input-buffers is equal to their *IP_PFC*, we might get low burst handling capability (particularly, in cases the flits are destined to an output port with higher egress link bandwidth). The router won't be able to fully utilize the given output port, since the read-rate of the corresponding input-port is lower than the bandwidth of the given output port. Therefore, in order to increase the egress link utilizations, the read-rate of the input queues should be equal to the maximum egress link bandwidth (i.e., $\max_i\{OP_i\_PFC\}$), which results in higher buffer and crossbar costs. Since the read rate of all input-buffers is determined by the highest bandwidth output port, the total read rate of the input-buffers can't be simultaneously utilized (unless all output ports have the same bandwidth).

In contrast, our proposed router architecture can fully utilize the output ports even for read-rate equals to the corresponding ingress link bandwidth (*IP_PFC*). Conse-

quently, our proposed heterogeneous router architecture can resolve blocking while achieving full egress bandwidth and offers better performance as compared to input-buffer router architecture. Section 8 presents several evaluations that demonstrate this advantage.

6.2. Packet Length and Link Bandwidth Dependency Disadvantage

Our proposed router architecture accomplishes bandwidths heterogeneity by transmitting several concurrent flits per clock cycle. However, this approach might cause throughput degradation in some cases where egress link bandwidth (i.e., *OP_PFC* [flits/clock]) is not a multiple of the length of the packet (in flits) transmitted over it. Fig. 7(a) presents an example of a single flow, which consists of eight serial flit packets, is being transmitted over a single link. Fig. 7(b) presents the transmitted flits at each cycle (in the same column) for link bandwidth of four flits/clock. In this case the egress link is fully utilized; thereby the router achieves the maximum throughput. Fig. 7(c) presents the transmitted flits for link bandwidth of three flits/clock. However, in this case, the egress link is not fully utilized since the link bandwidth is not a multiple of the packet length. When the last two flits are transmitted, there is some unutilized link bandwidth, since a new packet cannot enter the egress link before the current packet has not freed its VC. The throughput degradation of egress link *o* is calculated by:

$$\frac{OP_o\_PFC - (L \bmod OP_o\_PFC)}{OP_o\_PFC \cdot \left\lceil L/OP_o\_PFC \right\rceil} \qquad (6)$$

Where, L is the length of the packet in flits. Our simulations validate this result and show that this disadvantage is mitigated when at least two flows can share the link. It is clear that, as the number of VCs increases, the probability for unutilized link bandwidth decreases and therefore throughput degradation becomes less significant.

Fig. 8 presents the end-to-end latency versus the flow's offered load for our proposed heterogeneous NoC router and for an *optimal heterogeneous router*. Fig. 8(a) presents the case where the egress link bandwidth (four flits/clock) is a multiple of the packet length (eight flits); it can be seen that our proposed heterogeneous NoC achieves the same saturation throughput as the *optimal heterogeneous router*. Fig. 8(b) presents the saturation degradation when the egress link bandwidth (three flits/clock) is not a multiple of the packet length (eight flits). According to equation (6), the saturation throughput of our heterogeneous NoC router (42.6Gbps) degrades by 11% as compared to the maximum achieved saturation throughput (48Gbps); it can be seen that our simulation validates this result. Moreover, our simulations show that this disadvantage is negligible (full egress link utilization is achieved) when the link has at least two VCs.

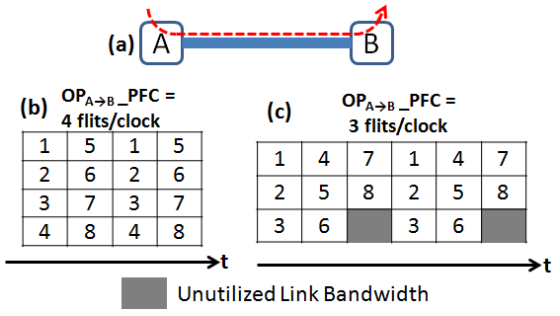


Fig. 7. (a) Throughput degradation example. Transmitted flits per clock cycle, for link bandwidth of: (b) four, (c) three flits/clock.

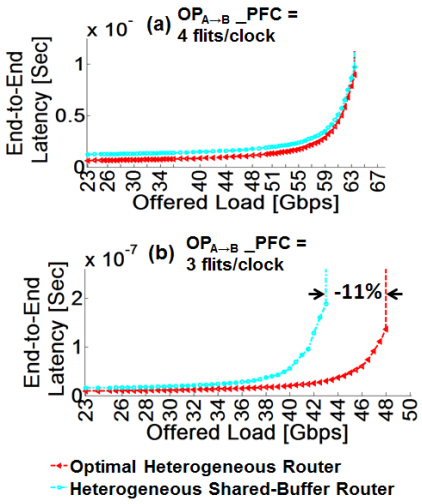


Fig. 8. End-to-end latency versus offered-load, for the example presented in Fig. 7. Packet length=8 flits,

6.3. Pipeline-Length and Input Queue Size

As mentioned in section 5.1, the heterogeneous NOC router uses credit-based flit-level flow control. The sufficient and necessary input queues size for achieving maximum throughput can be calculated by ([1]):

$$Size_{InputQueue} = Peak\ Throughput \cdot RTT \qquad (7)$$

The peak throughput units are [flits/cycle] and RTT is the number of cycles from the credit is sent upstream until the corresponding flit is sent downstream in response. Usually, in homogeneous NoC router, the peak throughput is single flit per clock cycle. Therefore, the input queue size of such routers equals to their RTT. In our proposed architecture, however, several flits can be concurrently transmitted at each clock cycle; thereby, the peak throughput can be greater than a single flit per clock (depends on the link's bandwidth). Furthermore, the RTT of our proposed is a bit much higher as compared to standard input-buffer routers. Thus, the necessary input queue sizes of our proposed architecture are greater as compared to homogeneous NoC routers.

One might argue that the pipeline length of our proposed router architecture results in degraded performance, as compared to input-buffer. However, as explained above in section 6.1, and evaluated next in section 8, our proposed architecture achieves better performance.

The performance of a router is mainly affected by the router's internal latency. It is determined mainly by the pipeline length, which contributes fixed latency, and by

the blocking probability, which contributes variable latency, depending on the router architecture, scheduling algorithm, traffic patterns and loads. While most previous work focuses on reducing the pipeline length, we use a longer pipeline to reduce the blocking probability, such that total internal latency is reduced and therefore the overall performance is better at realistic traffic conditions.

Nevertheless, many known optimizations can be applied to our router pipeline in order to reduce its RTT; for instance, look-ahead routing (LA) [1, 31] , speculative VC allocation [1, 32], pipeline bypassing [28, 33, 34], or speculation [34]. Such pipeline optimizations are not included in our router implementation and evaluation, and yet our evaluations result in better performance and performance efficiency as compared to input-buffer based heterogeneous NoC router (see Section 8)

7 Implementation Details

We implement our architecture using VHDL and synthesize it for TSMC 45nm process by Synopsys Design Compiler. We use registers for the shared-buffers, since the required storage size of each shared-buffer is several hundred bytes, which does not justify the usage of SRAM. Furthermore, we use circular buffers [35] for the shared-buffers in order reduce the power consumed by shifting data in the shared-buffers.

Fig. 9 presents the area evaluations of our implemented architecture. Fig. 9(a) presents the router area versus total ingress and egress bandwidth ($\sum_i IP_i\_PFC$ and $\sum_i OP_i\_PFC$), for the conflict-free output-buffer mimic (i.e., the router parameters are: $SB = \sum_i OP_i\_PFC$, $SpeedUp_{SB} = \sum_i IP_i\_PFC$) and flit size of 32 bits. It can be seen that the area grows linearly with the total ingress and egress bandwidth. Therefore, our proposed architecture is scalable. Fig. 9(b) presents the router area (with $\sum_i IP_i\_PFC = \sum_i OP_i\_PFC = 10$ flits/clock, and frequency of 500MHz) for different bandwidth partitions between the ports. It can be seen that the area differences are within 2%. Hence, the area of our proposed router depends mostly on the total ingress and egress bandwidth rather than on the specific bandwidth partitions between the ports.

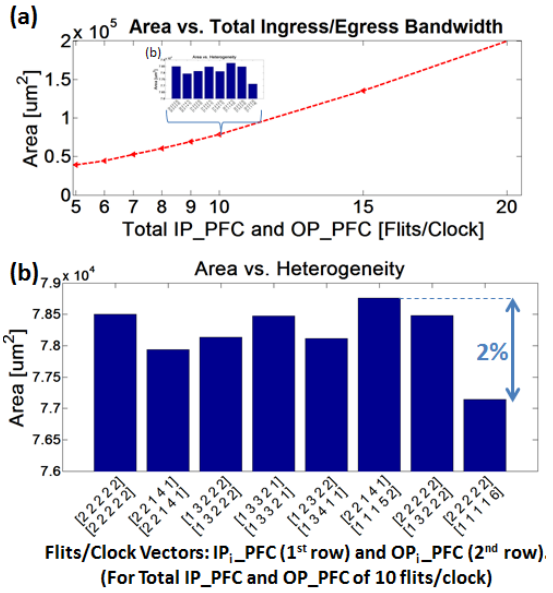


Fig. 9. Area evaluations of the architecture. (a) Area scalability; (b) The area for different *IP_PFC* and *OP_PFC* vectors (for total *IP_PFC and OP_PFC* of 10 flits/clock).

8 Quantitative Results

8.1. Performance Comparison Methodology

To the best of our knowledge, there is no previous detailed architecture for heterogeneous NoC router; hence, we cannot directly compare it to previous heterogeneous router architectures. To that end, we define an *optimal input-buffer heterogeneous router* as a reference architecture (e.g., [2]). This router uses different frequency but equal link width for each unidirectional port. The read-rate of each input-queue is unlimited (i.e., it can achieve full utilization of all egress links), and its internal latency equals zero (i.e., the router is *optimal* in terms of latency). We also define an *optimal input-buffer homogeneous router*, for homogeneous topology (i.e., same link capacity and number of VCs for all unidirectional ports). We fairly compare the routers to the shared-buffer heterogeneous router, by using the same total link capacities and total number of VCs.

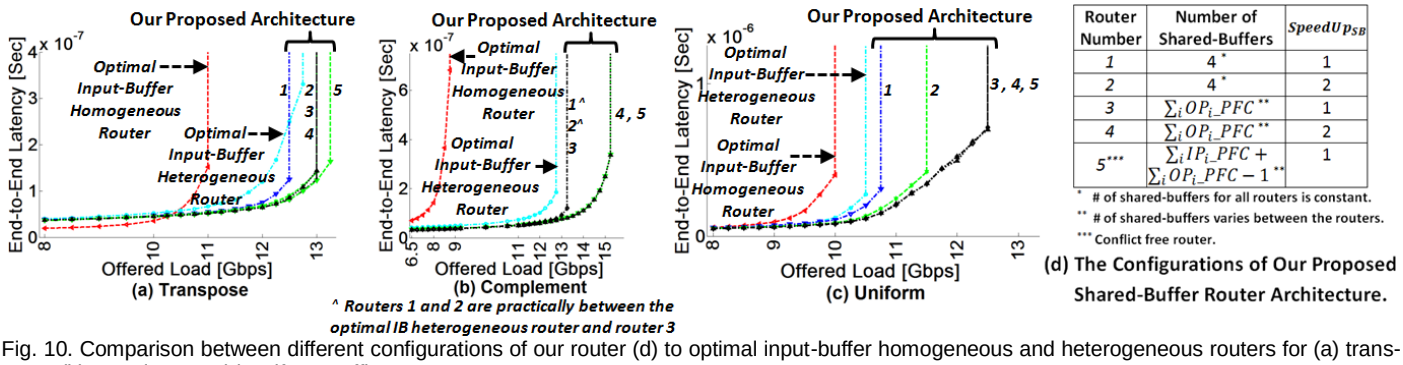
Fig. 10(d) details the configurations of the shared-buffer heterogeneous router we use for our evaluation. For the sake of ultimate performance comparison, we also simulate a conflict-free router (*router 5*), which indicates an upper performance bound, although this router is not a practical design choice.

We simulate the router architecture and the *optimal IB heterogeneous* and *homogeneous routers* using HNOCS [36]. HNOCS is an OMNeT++ [37] based heterogeneous NoC simulator. It supports any heterogeneous NoC in terms of any link capacity and any number of VCs. We simulate deterministic XY routing without loss of generality, frequency of 500MHz (based on the synthesis in section 7), flits of 32 bits and packets of eight flits.

8.2. Evaluation of Standard Traffic Patterns

In this section, we compare the performance of the shared-buffer heterogeneous router, the *optimal IB heterogeneous router*, and the *optimal IB homogeneous router* for a 4x4 mesh NoC. We also demonstrate how the number of shared-buffers and the *shared-buffer write speed-up* can be configured according to the preferred cost-performance ratio. Moreover, we demonstrate that the same performance can be achieved using fewer shared-buffers and a higher *shared-buffer write speed-up* (section 5.3). We use an optimal heterogeneous topology for the heterogeneous routers, based on optimization method presented in [9].

Fig. 10 presents the average end-to-end latency versus offered-load for *transpose*, *complement* and *uniform* traffic patterns. The homogeneous NoC results in poor performance as compared with the heterogeneous ones (saturation throughput is reduced by 6%-47%). The saturation throughput of the shared-buffer heterogeneous routers is higher by 2%-20% as compared with the *optimal IB heterogeneous router*. In general, the performance is improved by increasing the number of shared-buffers and/or the *shared-buffer write speed-up*. Fig. 10(a) and Fig. 10(c)



Router Number	Number of Shared-Buffers	$SpeedUp_{SB}$
1	4 *	1
2	4 *	2
3	$\sum_i OP_i\_PFC$ **	1
4	$\sum_i OP_i\_PFC$ **	2
5***	$\sum_i IP_i\_PFC + \sum_i OP_i\_PFC - 1$ **	1

* # of shared-buffers for all routers is constant.
** # of shared-buffers varies between the routers.
*** Conflict free router.

(d) The Configurations of Our Proposed Shared-Buffer Router Architecture.

Fig. 10. Comparison between different configurations of our router (d) to optimal input-buffer homogeneous and heterogeneous routers for (a) transpose, (b) complement, (c) uniform traffic patterns

demonstrate that the saturation throughput is improved by 5% on average by increasing the *shared-buffer write speed-up* for the router with four shared-buffers (*router 1* versus *router 2* in Fig. 10(d)). Furthermore, Fig. 10(a) and Fig. 10(b) demonstrate that the same saturation throughput is achieved using fewer number of shared-buffers and higher *shared-buffer write speed-up (router 2* versus *router 3)*. *Router 3* results in higher number of shared-buffers as compared to *router 2*, while employing lower *shared-buffer write speed-up*. Hence, this architecture can offer the same performance while reducing both hardware resources and power consumption (see sections 5.3 and 8.3).

8.2.1. Comparison to Homogeneous Shared-Buffer Router ([18])

In this sub-section, we compare the saturation throughput improvement relative to input-buffer router between a homogeneous ([18]) and our heterogeneous shared-buffer router architecture.

Ramanujam et al. in [18] evaluate two homogeneous shared-buffer routers DSB200 and DSB300, consisting of five and 10 shared-buffers, respectively. Therefore, for the sake of fair comparison, Table 2 presents the comparison of *routers 1* and *2*, which utilize approximately the same resources as DSB200 and DSB300 routers. It can be seen that *router 1* results in lower saturation throughput improvement as compared to DSB200, which occurs due to lower number of shared-buffers.

However, by increasing the *shared-buffer write speed-up* for uniform traffic pattern, *router 2* can better utilize its heterogeneous unidirectional ports, and provide a better saturation point as compared to DSB200 and DSB300. On the other hand, the homogeneous architecture ([18]) does not benefit from doubling the shared-buffer resources (DSB200 vs. DSB300). For complement traffic pattern, which results in much higher heterogeneity of loads, our proposed heterogeneous architecture offers much greater saturation throughput improvement.

Table 2. Comparison of saturation throughput improvement relative to input-buffer architecture (SU_{SB} stands for $SpeedUp_{SB}$)

	Homogeneous [18]		Heterogeneous	
	DSB200 5 SBs	DSB300 10 SBs	*router 1* 4 SBs / 1 SU_{SB}	*router 2* 4 SBs / 2 SU_{SB}
Uniform	9%	9%	7%	15%
Complement	8%	9%	47%	47%

8.3. Evaluation of the Shared-Buffer Heterogeneous NoC-Based CMP

In this section, we evaluate the performance of our architecture over a 4x4 NoC-based CMP (Fig. 2(b)), which was presented in [9] (Disclaimer: the core placement of the CMP is not necessarily optimal; the focus of this paper is on router architecture, hence no attempt was made to optimize the placement). We demonstrate that our NoC can achieve better performance (as compared with *optimal IB homogeneous* and *heterogeneous* routers) by adjusting its parameters (i.e., number of shared-buffers and $SpeedUp_{SB}$), see also section 9 for further discussion.

8.3.1. Performance Evaluation of the Shared-Buffer Heterogeneous NoC-Based CMP

In order to evaluate and compare the performance of our architecture, HNOCS is extended to provide functionality of a core with L1-cache, L2 shared-cache and DRAM controller. We assume 64KB L1-cache and 4MB L2 shared-caches with both 64 bytes cache lines and associativity of 64 lines per set. We use PARSEC benchmark suite [38] and extract the L2 cache accesses by Pin tool [39]; then, we use it to build new traces and apply them to our simulation model.

Fig. 11 presents the run-time improvement of the PARSEC benchmarks for the *optimal IB heterogeneous router*, and several configurations of the shared-buffer heterogeneous router (Fig. 10(d)) relative to the *optimal IB homogenous router*. The router with four shared-buffers and *shared-buffer write speed-up* of one (*router 1*) improves the run-time by 35% on average as compared with the *optimal IB homogenous router*. The four shared-buffers with *shared-buffer write speed-up* of two (*router 2*) and the conflict-free router (*router 5*) improve average run-time by 45%.

8.3.2. Area and Power Evaluation of the Shared-Buffer Heterogeneous NoC-Based CMP

In order to evaluate area and power, we use the ORION model [26] and modified it to support our proposed architecture. Despite the fact that area estimation can be obtained from our design synthesis (as presented in section 7), we use ORION for both area and power estimations, in order to preserve estimation uniformity.

Fig. 12(a) presents the total NoC area of the 4x4 NoC-based CMP for the *optimal IB homogenous* and *heterogeneous routers* and for several configurations of our heteroge-

neous router (see Fig. 10(d)). The crossbar and buffers are largest area consumers, respectively. Fig. 12(b) presents the total NoC power consumption of the 4x4 NoC-based CMP. To that end, we integrate ORION model to HNOCS. The links and routers utilizations are derived from HNOCS and used by ORION model to evaluate the total power consumption. We present only the most significant power components, which are the dynamic power of the logic and buffers. It can be seen that the buffers are largest power consumer. *router 1* increases the total NoC area by 8% and 12%, and power consumption by 10% and 38%, as compared to the *optimal IB heterogeneous* and *homogenous routers*, respectively. However, its end-to-end delay does not seem to out-perform the *optimal IB heterogeneous router* (Fig. 10(a)). Therefore, in order to achieve better performance, one should increase the *shared-buffer write speed-up; f*or instance, *router 2,* with *shared-buffer write speed-up* equals two, which offers much better latency (e.g., Fig. 10(c)). However, *router 2* allows more flits to be written into the shared-buffers at each clock cycle. Therefore, *router 2* results in additional 13% of buffers dynamic power relative to *router 1*. Moreover, it requires to duplicate the first cross-bar (see Fig. 5(a)) which results in additional area of 58%. The conflict-free router (*router 5*), which demonstrate the best performance, consumes the largest area, 270% of the *optimal IB heterogeneous* and *homogenous routers*, since it requires $\sum_i IP\_PFC_i + \sum_i OP\_PFC_i - 1$ shared-buffers, which increase both buffers and cross-bars areas. It also has the largest power consumption, 280% of the *optimal IB homogenous router.* We elaborate on the area and power penalties in section 9.

8.3.3. Performance Efficiency Evaluation of the Shared-Buffer Heterogeneous NoC-Based CMP

In this section, we evaluate the performance efficiency, i.e., *Performance/Area Ratio (PAR)* and *Performance/Power Ratio* (*PPR*), of the *optimal IB homogenous* and *heterogeneous routers* and several configurations of our heterogeneous router (see Fig. 10(d)). To that end, we define the performance to be inversely proportional to the benchmark's run-time. Fig. 13(a) and Fig. 13(b) present the *PAR* and the *PPR,* respectively. *Router 1* achieves the highest *PAR* and the next-highest *PPR. Router 2* achieves the highest *PPR* and the next-highest *PAR*. On the other hand, the conflict-free router (*router 5*) achieves the lowest *PAR* and *PPR*.

The 4x4 NoC-based CMP example demonstrates that our proposed shared-buffer architecture can achieve better performance and performance efficiency (as compared to the *optimal input-buffer routers*) by adjusting the shared-buffer router parameters (i.e., number of shared-buffers and *shared-buffer write speed-up*). The highest performance efficiency metrics are achieved by the routers with four shared-buffers and *shared-buffer write speed-up* of one and two (*routers 1* and *2*). *router 1* improves the *PAR* by 41% and *router 2* improves the *PPR* by 22% as compared with *optimal IB homogenous router*. Furthermore, *router 2* achieves the next-highest run-time improvement, which is just 1% lower than the highest improvement on average. On the other hand, the conflict-free router (*router 5*) achieves the highest run-time improvement but the lowest performance efficiency metrics.

8.3.4. The Scalability of the Shared-Buffer Heterogeneous NoC-Based CMP

In this section, we demonstrate the scalability of the shared-buffer heterogeneous router architecture. To that end, we simulate NoC-based CMPs in several sizes, from 4x4 up to 16x16. We compare the *optimal IB homogenous* and *heterogeneous routers* to our routers. Clearly, the size of each CMP imposes different total link capacities and number of VCs. Therefore, in order to adjust to this difference, the number of shared-buffers of the routers is customized for each CMP size as indicated in Fig. 14 and Fig. 15.

Fig. 14 presents a comparison of the performance efficiency metrics; i.e. performance/area and performance/power. In order to evaluate area, we use the ORION model [26] and modified it to support our architecture. To evaluate power, we integrate ORION model to HNOCS. The links and routers utilizations are derived from HNOCS and used by ORION model to evaluate the total power consumption. Fig. 14 demonstrates that the *optimal IB homogeneous router* is not scalable, since

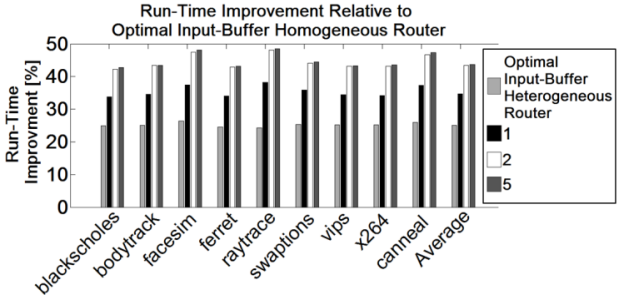


Fig. 11. Improvement of the PARSEC benchmarks run-time execution for several heterogeneous routers (according to Fig. 10(d)) relative to the *optimal IB homogeneous router* (for the same total capacity and VCs).

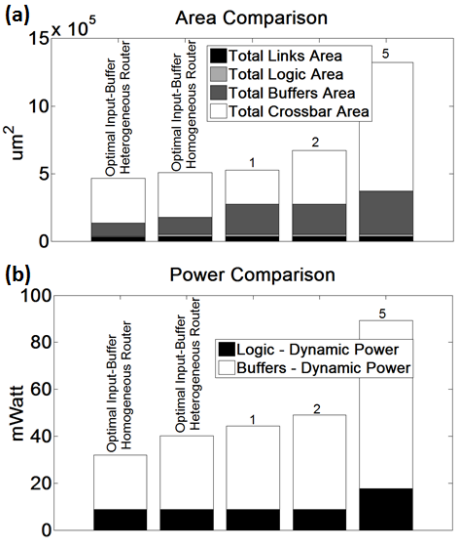


Fig. 12. NoC area and power comparison of 4x4 CMP. For optimal input-buffer homogeneous and heterogeneous routers; and for several shared-buffer heterogeneous routers (according to Fig. 10(d)).

its internal blocking probability increases as traffic loads increases (with CMP size). Therefore, more resources are required in order to achieve reasonable performance as compared with heterogeneous NoC routers (as also explained in [9]). While the *optimal IB heterogeneous router* offers reasonable scalability; the shared-buffer heterogeneous architecture demonstrates better performance efficiency metrics as the CMP size increases.

Fig. 15(a) and Fig. 15(b) present the area and power reduction, respectively, as compared with the *optimal IB homogeneous router* for the same achieved performance of each CMP. Fig. 15(c) presents the area and power reduction ranges achieved by the heterogeneous routers. The highest area and power reduction are achieved by the heterogeneous shared-buffer router with *shared-buffer write speed-up* of one and two, respectively. Increasing the *shared-buffer write speed-up* results in higher area of the first crossbar and logic. On the other hand, it mitigates the router's internal blocking probability; hence, the performance improves which allows further reduction of power consumption for the same achieved performance.

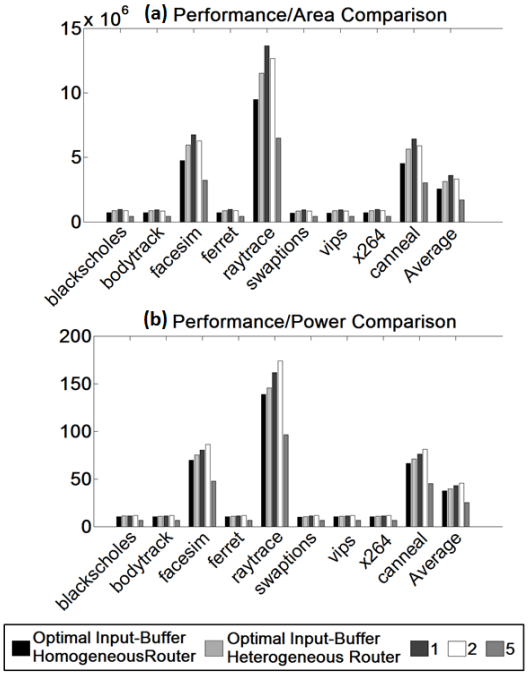


Fig. 13. Performance to area and power comparison of 4x4 CMP running PARSEC benchmarks for optimal input-buffer homogeneous and heterogeneous routers; and for several shared-buffer heterogeneous routers (according to Fig. 10(d)).

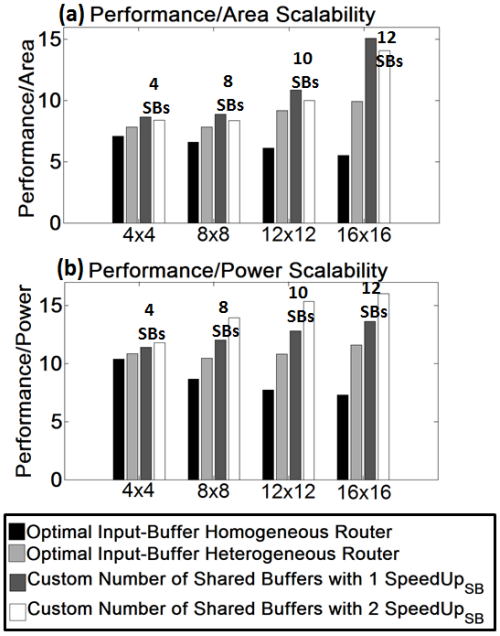


Fig. 14. Scalability demonstration by performance efficiency metrics (for NoC-based CMPs), compared with input-buffer architectures.

9 Discussion and Future Directions

In this paper, we present a new NoC design approach which benefits from system heterogeneity. This approach also introduces a trade-off between design complexity and achieved NoC efficiency. Most previous works choose homogeneous NoCs in order to reduce design time due to regularity of NoC components. Our approach achieves better NoC performance for a given amount of network resources. Such approach may not be practical for today's chips using current design tools, but would be definitely relevant for chips in the future, which will require much higher performance and tighter constraints, using new design automation capabilities.

Parameters Adjustment: In section 8, we demonstrate that the achieved performance of our router architecture depends on the router parameters. In the following, we describe how some guidelines to adjust these parameters.

First, the number of shared-buffers should be equal to the effective egress bandwidth of the router that equals to the maximum number of flits, which can be (or actually) simultaneously transmitted through all egress ports. This condition guarantees maximum egress bandwidth utilization. Second, increasing the *shared-buffer write speed-up,* which reduces its internal blocking, can improve the router latency (as explained in section 5.3). Fig. 10

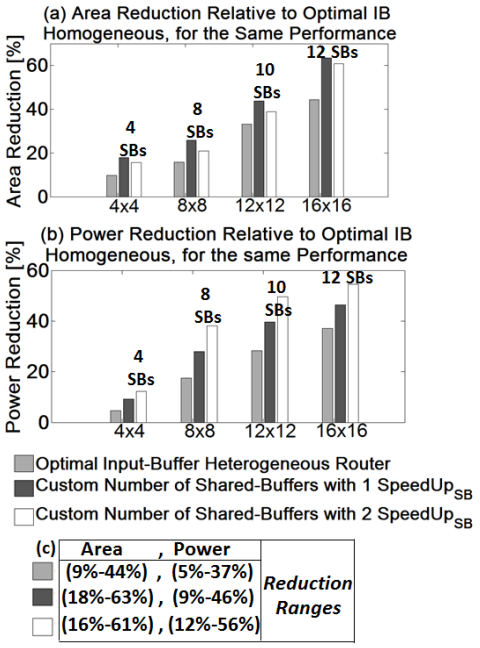


Fig. 15. Area and power reduction as compared with *optimal IB homogeneous router* for the same performance.

demonstrates several routers with sufficient number of shared-buffers, where their latency is improved by increasing the *shared-buffer write speed-up*, e.g., routers 1 and 2 for transpose and uniform traffic patterns, and routers 3 and 4 for complement. In addition, Fig. 11 demonstrates the same trend for routers 1 and 2.

In addition, the depth of VCs and shared buffers also affect the overall performance of the NoC. For given budget of buffers, deeper buffers results in lower number of VCs and vice versa. Such buffers organization has already been researched for input-buffer router [40, 41], and [42]. Rezazad et al. [42] demonstrate that uniform traffic pattern achieves saturation point improvement of maximum 16% among different VC buffers organizations. Whereas, we demonstrate 20%-25% improvement as compared to *optimal IB homogenous router*, before employing such VC buffers optimization. Such optimal VC buffer organization is known to be depended on the given traffic pattern and offered load [40, 41, 42]. Some traffic pattern scenarios favor deeper buffers, while others favor higher number of VCs. Regarding the shared-buffer organization, one should maintain the first guideline aforementioned; i.e. set the number of shared-buffers according to the total effective egress bandwidth of the router. The shared-buffers' depth, on the other hand, has much lower impact on the NoC performance, as we observe through our simulation explorations.

Design Flow: Heterogeneous NoC architecture results in the inclusion of many versions of routers on the same chip. This is a burden the design flow; since cost and time for implementing a chip increase exponentially with the number of design blocks in the chip. However, such architectures may be appropriate in the future, for bigger chips and better automated design flows.

Area and Power Consumption: Our proposed architecture achieves better performance, but consumes more area and power as compared to input-buffer routers. Such increase might seem to be unacceptable, for current chips. Input-buffer homogeneous routers, however, don't utilize all the NoC resources, and might be insufficient for future chips, consisting of hundreds of modules, which impose high loads. On the other hand, our proposed architecture can deliver better performance and deploy NoC resources only where they are necessary. Hence, such area and power penalties would be acceptable for such chips.

Future Work: This paper is the first to introduce such architecture, which can serve as a baseline for better and more efficient future architectures. For instance, a hybrid heterogeneous NoC consisting of both input and shared-buffer routers, depending on the regional load of the NoC. Such hybrid NoC can ease the area and power consumption described above, while still offering superior performance. Another possible direction is to implement *shared-buffer read speed-up,* which can further reduce the number of shared-buffers as described in the above Parameters Adjustment paragraph. Such architecture will further reduce the consumed area of the router while preserving its offered performance.

10 Summary

A novel heterogeneous NoC router architecture, supporting different link bandwidths and number of VCs per unidirectional port is presented. These features are important, since typical traffic in real systems is not uniform in all links. The main advantages of the shared-buffer heterogeneous NoC router are: better ingress and egress bandwidth decoupling, better performance, and ability to configure the router according to the preferred cost-performance ratio. We presented and formally proved a novel approach, which reduces the number of shared-buffers required for a conflict-free router. Reducing number of required shared-buffers also reduces the first and second crossbar sizes and in overall can dramatically reduce area and power consumption. Finally, we demonstrated the performance, hardware resource savings, and scalability of the router for predefined traffic patterns and for NoC-based CMP at various sizes.

References

[1] W. Dally and B. Towles, *Principles and practices of interconnection networks*. Morgan Kaufmann, 2004.

[2] A. Mishra, N. Vijaykrishnan, and C. Das, "A case for heterogeneous on-chip interconnects for CMPs," in *Proceeding of the 38th annual international symposium on Computer architecture*., 2011.

[3] D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini, and G. De Micheli, "Noc synthesis flow for customized domain specific multiprocessor systems-on-chip," *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 2, 2005.

[4] B. Beckmann and D. Wood, "Managing wire delay in large chip-multiprocessor caches," in *Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Computer Society, 2004.

[5] Z. Guz, I. Walter, E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "Efficient link capacity and qos design for network-on-chip," in *Proceedings of the conference on Design, automation and test in Europe*, 2006.

[6] A. Agarwal, C. Iskander, and R. Shankar, "Survey of network on chip (noc) architectures & contributions," *Journal of engineering, Computing and Architecture*, vol. 3, 2009.

[7] E. Salminen, A. Kulmala, and T. Hamalainen, "Survey of network-on-chip proposals," *white paper, OCP-IP*, 2008.

[8] F. Moraes, N. Calazans, A. Mello, L. Möller, and L. Ost, "Hermes: an infrastructure for low area overhead packet-switching networks on chip," *the VLSI journal INTEGRATION*, vol. 38, no. 1, 2004.

[9] Y. Ben-Itzhak, I. Cidon, and A. Kolodny, "Optimizing heterogeneous noc design," in *Proceedings of the International Workshop on System Level Interconnect Prediction*. ACM, 2012.

[10] H. Chao and B. Liu, *High performance switches and routers*. Wiley-IEEE Press, 2007.

[11] T.-C. Huang, U. Y. Ogras, and R. Marculescu, "Virtual channels planning for networks-on-chip," in *Quality Electronic Design, 2007. ISQED'07. 8th International Symposium on*. IEEE, 2007.

[12] A. B. Kahng, B. Lin, K. Samadi, and R. S. Ramanujam, "Trace-driven optimization of networks-on-chip configurations," in *Design Automation Conference (DAC), 2010 47th ACM/IEEE*. IEEE, 2010.

[13] B. Grot, J. Hestness, S. W. Keckler, and O. Mutlu, "Kilo-noc: a heterogeneous network-on-chip architecture for scalability and service guarantees," in *ACM SIGARCH Computer Architecture News*, vol. 39, no. 3. ACM, 2011.

[14] M. P. Véstias and H. C. Neto, "Area and performance optimization of a generic network-on-chip architecture," in *Proceedings of the 19th annual symposium on Integrated circuits and systems design*. ACM, 2006.

[15] A. Bakhoda, J. Kim, and T. Aamodt, "Throughput-effective on-chip networks for manycore accelerators," in *Proceedings of the*

2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture. IEEE Computer Society, 2010.

[16] M. Kreutz, C. Marcon, L. Carro, F. Wagner, and A. Susin, "Design space exploration comparing homogeneous and heterogeneous network-on-chip architectures," in *Proceedings of the 18th annual symposium on Integrated circuits and system design*. ACM, 2005.

[17] H. Zhao, M. Kandemir, W. Ding, and M. J. Irwin, "Exploring heterogeneous noc design space," in *Proceedings of the International Conference on Computer-Aided Design*. IEEE Press, 2010.

[18] R. Ramanujam, V. Soteriou, B. Lin, and L. Peh, "Design of a high-throughput distributed shared-buffer noc router," in *Networks-on-Chip, 2010 Fourth ACM/IEEE International Symposium*, 2010.

[19] A. T. Tran and B. M. Baas, "Roshaq: High-performance on-chip router with shared queues," in *Computer Design (ICCD), 2011 IEEE 29th International Conference on*. IEEE, 2011.

[20] J. Kim, J. Balfour, and W. Dally, "Flattened butterfly topology for on-chip networks," in *Proceedings of the 40th Annual IEEE/ACM International Symposium on Micro architecture,* 2007.

[21] L. Leung and C. Tsui, "Optimal link scheduling on improving best-effort and guaranteed services performance in network-on-chip systems," in *Proceedings of the 43rd annual Design Automation Conference*. ACM, 2006.

[22] M. Neishaburi and Z. Zilic, "Reliability aware noc router architecture using input channel buffer sharing," in *Proceedings of the 19th ACM Great Lakes symposium on VLSI*. ACM, 2009.

[23] M. Neishaburi and Z. Zilic, "Eravc: Enhanced reliability aware noc router," in *12th International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2011.

[24] S. Iyer, R. Zhang, and N. McKeown, "Routers with a single stage of buffering," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, 2002.

[25] H. Wang, L. Peh, and S. Malik, "Power-driven design of router microarchitectures in on-chip networks," in *Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Computer Society, 2003.

[26] A. Kahng, B. Li, L. Peh, and K. Samadi, "Orion 2.0: A fast and accurate noc power and area model for early-stage design space exploration," in *Proceedings of the conference on Design, Automation and Test in Europe*, 2009.

[27] L. Benini and G. De Micheli, "Powering networks on chips: energy-efficient and reliable interconnect design for socs," in *Proceedings of the 14th international symposium on Systems synthesis*. 2001.

[28] A. Kumar, P. Kundu, A. Singhx, L.-S. Peh, and N. K. Jha, "A 4.6 tbits/s 3.6 ghz single-cycle noc router with a novel switch allocator in 65nm cmos," in *Computer Design, 2007. ICCD 2007. 25th International Conference on*. IEEE, 2007.

[29] C. A. Nicopoulos, D. Park, J. Kim, N. Vijaykrishnan, M. S. Yousif, and C. R. Das, "Vichar: A dynamic virtual channel regulator for network-on-chip routers," in *Microarchitecture, 2006. MICRO-39. 39th Annual IEEE/ACM International Symposium on*. IEEE, 2006.

[30] P. Kundu, "On-die interconnects for next generation cmps," in *Workshop on On-and Off-Chip Interconnection Networks for Multicore Systems (OCIN)*, 2006.

[31] M. Galles, "Scalable pipelined interconnect for distributed endpoint routing: The sgi spider chip," in *Hot Interconnects*, 1996.

[32] L.-S. Peh and W. J. Dally, "A delay model and speculative architecture for pipelined routers," in *High-Performance Computer Architecture, 2001. HPCA.*

[33] A. Kumar, L.-S. Peh, and N. K. Jha, "Token flow control," in *Proceedings of the 41st annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Computer Society, 2008.

[34] R. Mullins, A. West, and S. Moore, "Low-latency virtual-channel routers for on-chip networks," *ACM SIGARCH Computer Architecture News*, 2004.

[35] S. Smith *et al.*, "The scientist and engineer's guide to digital signal processing," 1997.

[36] Y. Ben-Itzhak, E. Zahavi, I. Cidon, and A. Kolodny, "Hnocs: Modular open-source simulator for heterogeneous nocs," in *Proceedings of the 12th International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XII)*, 2012.

[37] A. Varga *et al.*, "The OMNeT++ discrete event simulation system," in *Proceedings of the European Simulation Multiconference (ESM'2001)*, 2001.

[38] C. Bienia, S. Kumar, J. Singh, and K. Li, "The PARSEC benchmark suite: Characterization and architectural implications," in *Proceedings of the 17th international conference on Parallel architectures and compilation techniques*. ACM, 2008.

[39] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi, and K. Hazelwood, "Pin: building customized program analysis tools with dynamic instrumentation," in *ACM SIGPLAN Notices*, vol. 40, no. 6. ACM, 2005.

[40] N. Parik, O. Deniz, P. Kim, and Z. Li, "Buffer allocation approaches for virtual channel flow control," 2008.

[41] N. Alzeidi, M. Ould-Khaoua, and L. Mackenzie, "Deep versus parallel buffers in wormhole switched k-ary n-cubes."

[42] M. Rezazad and H. Sarbazi-Azad, "The effect of virtual channel organization on the performance of interconnection networks," in *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*. IEEE, 2005.

Yaniv Ben-Itzhak is a PhD candidate in the Electrical Engineering Faculty at the Technion, Israel Institute of Technology under the supervision of Prof. Israel Cidon and Prof. Avinoam Kolodny. His research interests include Multi-Core Processors (CMP), Network on Chip (NoC), and Interconnect for Data-Centers and HPC. He is a recipient of several awards, including Intel research award and HPI fellowship.

Israel Cidon is a Professor and the former Dean of the Electrical Engineering Faculty at the Technion, Israel Institute of Technology. His research interests include computer networks, distributed systems, Multi-Core Processors and Network-on-Chip. He was a research member and the manager of the Network Architecture and Algorithms group at Watson IBM Research, and received twice the IBM outstanding innovation awards. He was a founding editor for the IEEE/ACM Trans. on Networking and editor for the IEEE Trans. on Communications. He is the co-author of over 170 papers and 28 US patents.

Avinoam Kolodny received his doctorate in microelectronics from Technion, Israel Institute of Technology in 1980. He joined Intel Corporation, and engaged in research and development in the areas of device physics, VLSI circuits, electronic design automation, and organizational development. He has been a member of the Faculty of Electrical Engineering at the Technion since 2000. His current research is focused primarily on interconnects in VLSI systems, at both physical and architectural levels.

Michael Shabun received his B.Sc in Electrical Engineering from the Technion – Israel institute of technology, in 2014. He is a recipient of the best project award 2013 from the VLSI laboratory in the Technion. His interests include computers, networks and networks-on-chip. Currently, he works as an Automation Engineer at Intigua, a virtualization software company.

Nir Shmuel received his B.Sc (cum laude) degree in Electrical Engineering from the Technion, Israel institute of technology, in 2013. His interests include communications, DSP, and electro-magnetics. He is a recipient of the best project award 2013 from the VLSI laboratory in the Technion. Currently, he works in Chip Design at EZchip Technologies Ltd.