Parallel Forwarding for Efficient Bandwidth Utilization in Networks-on-Chip

Elham Momenzadeh[1], Mehdi Modarressi[2(✉)], Abbas Mazloumi[2], and Masoud Daneshtalab[3,4]

[1] School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran
elham.momenzade@gmail.com
[2] Department of Electrical and Computer Engineering, University of Tehran, Tehran, Iran
{modarressi,a.mazloumi}@ut.ac.ir
[3] Mälardalen University (MDH), Västerås, Sweden
[4] Royal Institute of Technology (KTH), Stockholm, Sweden
masdan@kth.se

Abstract. Networks-on-chip (NoC) provide a scalable and power-efficient communication infrastructure for different computing chips, ranging from fully customized multi/many-processor systems-on-chip (MPSoCs) to general-purpose chip multiprocessors (CMPs). A common aspect in almost all NoC workloads is the varying size of data transmitted by each transaction: while large data blocks are transferred as multiple-flit packets, a part of the traffic consists of short data segment (control data) that does not even fill a single flit. In conventional NoCs, switch allocator assigns/grants a switch output (and the link connected to it) to a single flit at each cycle, even if the flit is shorter than the link bit-width. In this paper, we propose a novel NoC architecture that enables routers to simultaneously send two short flits on the same link, effectively utilizing the link bandwidth that otherwise would be wasted. To this end, new crossbar, virtual channel (VC), and switch allocator architectures are presented to support parallel short packet forwarding on NoC links. Simulation results using synthetic and realistic workloads show that the proposed architecture improves the NoC performance by up to 24%.

Keywords: Network-on-Chip · Heterogeneous packet size · Bandwidth utilization

1 Introduction

Networks-on-chip (NoC) are widely known as the most promising solution to handle inter-core communication in multi- and many-core architectures. NoCs provide a power-efficient infrastructure with scalable bandwidth for on-chip communication. As the core count and workload complexity of chip multiprocessors (CMP) and multi/many-processor systems-on-chip (MPSoC) increase, the rate and complexity of on-chip communication raise considerably. Consequently, there is always a growing demand for NoCs with higher throughput and lower latency.

J. Knoop et al. (Eds.): ARCS 2017, LNCS 10172, pp. 152–163, 2017.
DOI: 10.1007/978-3-319-54999-6_12

NoC bit width (flit size) is a first-order design parameter that highly affects the maximum network bandwidth and packet latency. This parameter determines the bit-width of all NoC datapath components (i.e. link, buffer, and crossbar). As a result, in addition to its impact on performance, bit-width also plays an important role in determining the total NoC implementation cost and power consumption.

Performance metrics always favor enlarging bit-width (as long as the cost constraint allows), because wider links decrease packet serialization overhead, thereby enhance both the speed and throughput of networks.

Recent NoC designs and commercial implementations use links as wide as 128 [1, 2], 144 [3], 160 [4], 256 [5], and 512 bits [6] to maximize performance with respect to area constraints. However, the message size, that is the amount of data transmitted at each network transaction, varies significantly in realistic workloads [7]. For example, in a typical CMP workload, the traffic consists of long data and short control packets. Data packets composed of multiple flits to transfer a cache block, while control packets transfer request and coherency messages that contain a memory/IO address plus a few control bits. Whereas the former benefits from larger bit widths, the latter cannot even fill half of the bit width at each transfer [8].

In some recent studies, it has been shown that a considerable portion of traffic in CMP workload is the short request and coherency packets [7, 9, 10]. For example, it has been shown that more than 78% of the packets in the PARSEC suite programs [11] are short control packets (request or coherency), whereas the remaining packets are long and contain a full 64B cache line [9]. Very different packet sizes (from 8-bit control packets to data packets with kilobits of data) are also reported for multimedia and telecommunication workloads implemented on application-specific NoCs [12]. As mentioned before, NoCs enlarge bit-width to reduce data serialization overhead of time-critical data packets, but this results in considerable bandwidth waste and resource underutilization when sending short packets: A short control flit uses part of the bit-width, leaving the remaining bits idle and the link underutilized. Buffers are also become underutilized in conventional NoCs, because those buffer slots that keep control packets have many bits zero-padded. However, conventional switch allocators allocate the switch output (and the corresponding downstream link) to a single flit, regardless its bit-width usage.

In this paper, we propose a novel architecture that enables routers to transfer and store two short flits through each port in parallel. In this architecture, if two or more short flits request for an output port, the switch allocator grants the port to two flits: the second flit uses the otherwise idle bit-width of the link to go downstream in parallel with the first flit. The input port also supports receiving and buffering two short flits simultaneously.

These scheme decreases the switch allocation failure rate and hence, part of the unnecessary short flit blocking latency is eliminated.

As a quantitative motivation on the potential impact of our proposed parallel forwarding on performance. Table 1 shows the percentage of switch allocations with at least one loser under two representative workloads: a workload with high injection rate from the ISPASS GPU benchmark suite [13] and a workload with light traffic load from the PARSEC CMP suite [14]. The table also shows in what percentage of the total switch allocation failures a short flit is blocked by another short flit.

Table 1. Switch allocation failure analysis

Workload	Total switch allocation failures (%)	Percentage of total failures with two short flits (%)
PARSEC: Ferret (0.08 flit/node/cycle)	16.32	36.12
GPU: BFS (0.3 flit/node/cycle)	38.08	30.93

As the table shows, the proposed parallel short packet forwarding can potentially reduce switch allocation failure rate (which is 16% and 38% in Table 1) by up to 36%. This architecture allows designers to increase bit width in favor of long data packets and mitigate resource underutilization by parallel short packet transfer/storage.

Since control transfers account for a considerable amount of on-chip traffic in CMP workloads, many flits can take advantage of the proposed mechanism and so, the performance and resource utilization of the NoC increases considerably.

Several prior works proposed to use physically separate sub-networks to handle each traffic class (data and control) appropriately [6] or to reduce power [8].

For example, Intel Xeon Phi uses a 512-bit wide ring for data packets and very narrower sub-networks for control and address packets [6]. However, multiple sub-networks have a higher cumulative area than a single network. This can potentially increase the implementation cost and power consumption of the NoC.

Our mechanism implements different sub-networks into the same NoC fabric. It can be considered as a polymorphic NoC: long packets see links and buffers as single n-bit structures, whereas these components act as two n/2-bit parallel structures from the short packets' perspective.

In the next sections, we first explore the related work, introduce the proposed NoC architecture and then show it can reduce NoC latency by up to 24% and throughput by 30%, on average.

2 Related Work

Several hybrid network-on-chip designs can be found in the literature that partition the NoC into multiple parts and optimize each part for a specific traffic class.

Using physically separated NoCs for data and control packets is also proposed in many related work. The authors in [15] show the potential benefits of using multiple physicals sub-networks for control and data packets.

In [16], a NoC is partitioned into two packet-switched and circuit-switched sub-networks using time-division multiplexing (TDM). The packet-switched sub-network carries request packets, whereas the circuit-switched part is used to make shortcut paths for data packets. Each request packet makes circuit for its corresponding data packet while traveling towards the destination. Proactive Resource Allocation (PRA) NoC exploits the distinct characteristics of data and control packet to increase performance [17]. In PRA, most short request packets use

conventional packet switching, but multi-flit data packets are provided by pre-allocated paths, on which they are forwarded with low per-hop latency and power consumption.

Cache Coherent NoC (CCNoC) architecture is another hybrid architecture presented in [8]. It uses two different sub-networks for control and data to reduce the power consumption. As cache coherency protocols produce a group of write and read-request messages, managing cache coherency is done more efficiently by using dedicated sub-networks, in terms of both performance and power consumption. As control packets are smaller, lower bit width (64 bits) is applied for the request sub-network. Consequently, power consumption decreases while performance gets no impact. Response packets convey several cache blocks and so, require a higher bit width (112 to 128 bits). They showed that in addition to power-efficiency, using a heterogeneous structure results in a better performance than a unified NoC. As another insightful study in this field, [7] investigates the effect of bit width on performance and scalability of NoCs and concludes that the flit size should be set to the smallest packet type's size.

The above works focus on different aspects of workloads with mixed packet types. To the best of our knowledge, our proposed work is the first method that focuses on the underutilized NoC resources when forwarding short flits and modifies routers to allow parallel transfer of such short flits.

3 The Proposed NoC Architecture

3.1 NoC Packet Size

As our method targets CMP workloads, we consider two different kinds of packets in the network: long data and short control packets. Control packets are either memory and I/O requests or coherency messages that consist of a memory or I/O address along with a few control flags. Data packets are sent in response to a request and transfer a cache or memory block to a remote core. As the payload of these packets is a cache block, which can be as large as 32-128 bytes in a conventional cache (e.g. 64-bytes for Intel Xeon Phi [6] and ARM Cortex A15 [18]), they are long and should be fragmented into multiple flits. Carrying a memory address (which is 40-bit wide in ARM Cortex A15 [18], for example), a control packet would fill half of a 128-bit flit (40 bit memory address as payload, 10 bit destination address for network routing in a 1024-node network, and the remaining 14 bits for control, routing, and error recovery data), while a data packet requires five 128-bit flits (4 payload and one header). As an off-chip example, the HyperTransport protocol, which is implemented in modern AMD processors, also uses 512-bit packet for data and 64-bit packet for control transfer [19]. Therefore, in this paper, we use 128-bit links (flits), 64-bit control messages (that are sent as a single 128-bit flit in a conventional NoC with 64 zero-padded bits) and five-flit data packets.

3.2 Proposed Router Architecture

In a conventional architecture, as mentioned, each datapath element handles a single flit at each cycle. In this work, we propose to transfer and store two short control packet in parallel to use the idle bit width of the links, crossbar switches, and buffers that otherwise

would be wasted (filled by zero-padded null data). To this end, several router components must be modified: switch allocator to detect short flits and allocate a 128-bit link to two requesting ones, crossbar switch and links to transfer two short flits in parallel (in addition to the baseline one long flit), and virtual channels to accept and store two short flits simultaneously. This architecture is depicted in Fig. 1.

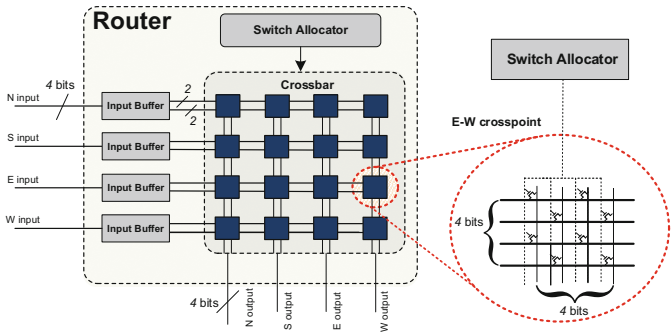


Fig. 1. The proposed router architecture and the internal connections of one crosspoint (East-West crosspoint) of the crossbar. Bit width is set to 4 for the sake of simplicity

Crossbar. In order to send two short flits simultaneously, the crossbar crosspoints should be capable to switch half bit width of inputs and outputs independently. For example, the crossbar should be able to connect the high half (n/2 bit) of input port E to the low half of output port S and the low half of input port E to the high half of output port N. However, if a long data packet is traversing the crossbar, the required connection is established on the full bit-width, just like a conventional NoC.

Figure 1 also shows the internal connections of a crosspoint. The figure shows 4-bit wide links for simplicity. The switches in the orthogonal positions implement the regular connections for full bit-width switching.

The other switches are added in our design to allow half bit width switching. As the figure shows, the new crossbar needs more switches to support half width switching.

Figure 2 shows several sample connections established on the crossbar at a cycle. In this figure, two control flits that come from input port N are connected to output ports S and E. A full width data flit that comes from input port E is connected to output port W. the idle half bit width of output port E is also connected to input port S. The internal connections of two connection points are also depicted in the figure, where the connected switches are identified by a circle.

Links. Two short flits should be able to pass a link at the same cycle. However, there is no need to add any logic to links to manage this parallel transfer, because short flit concatenation is done by crossbar switch. The two short flits will be directed to the right VCs assigned to them at the upstream router through the multiplexer at the downstream input port (Fig. 3). The multiplexers are set by upstream router, just like what a conventional router does.

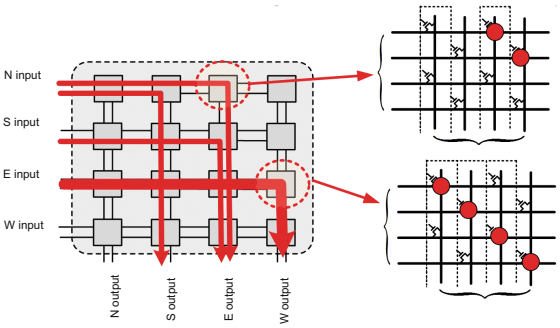


Fig. 2. Several full width and half width connections on the proposed crossbar and the internal connections of two sample switches. The switches identified by a red circle are turned on. (Color figure online)

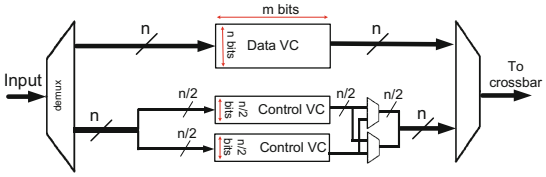


Fig. 3. The structure of the input unit with one data VC and two half width control VCs

Input VCs. NoCs that are used in CMPs often use different VCs for request, data, and coherency messages. The main advantage of using different VCs for each traffic class is that we can assign priority to packets based on packet's VC and order memory transactions to avoid protocol deadlocks. So, we consider two VCs per port and assign them to data and control (coherency or request) packets.

In our design, the data VC has m-entry n-bit wide buffer, as in a baseline router. The control VC consumes the same buffering space, but is horizontally partitioned to get two m-entry n/2-bit buffers (Fig. 3). Each narrow buffer has its own control logic to load/store flits simultaneously. As Fig. 3 shows, each input port has a single n-bit line to crossbar switch input. Switch allocator configures the multiplexers of this line to connect either one long flit or two short flits to the crossbar based on its allocation decision.

Switch Allocator. This component should distinguish short and long flits and grants each output link to at most one long or two short packets (if any). It should also allow a crossbar input to be shared by two short flits by appropriately setting the select line of the multiplexers between the input units and crossbar (see Fig. 3).

VC Allocator. VC allocator selects one of the control or data VCs for a packet based on its type. If there are more than one control VCs, a VC is selected for a requesting packet in a round robin fashion.

Please note that the VC allocation unit considers each narrow half-width VC as an independent VC. The demultiplexer in front of the input unit is capable to send two half-width flits to two different half-width control VCs in parallel (apart from its basic functionality that sends a full-width flit to a data VC).

4 Experimental Results

4.1 Experimental Environment

We use a cycle-accurate NoC simulator, BookSim [20], to simulate our architecture. We have tested the proposed NoC architecture under the uniform synthetic traffic pattern, as well as several traffic traces from the ISPASS GPU [13] and PARSEC benchmark suites. PARSEC traffic obtained from the Netrace library [14]. The GPU workload is the traffic between shader cores and memory modules extracted by GPGPUSim [13].

We use the mesh topology with wormhole-switched routers and 128-bit links (max flit size = 128). The routers are 3-stage pipelined (look-ahead routing + VC allocation, switch allocation, crossbar traversal + link traversal) and the routing algorithm is deterministic XY.

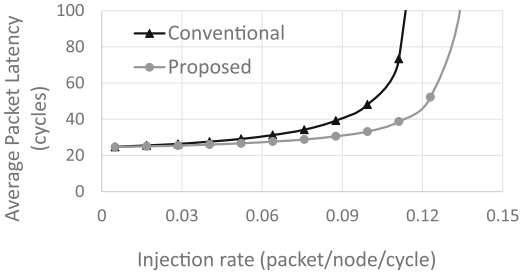
The network has two message classes that is a common configuration for CMPs to provide different levels of priority for response (data), and control (request and coherency) messages and resolve memory protocol deadlocks. A single virtual channel is considered for each message class. The data virtual channel is 128-bit wide and 8-flit deep. The control virtual channel that keeps short packets is partitioned horizontally and is arranged as two parallel 64-bit 8-flit buffers.

To evaluate our method, we compare each test-case with a conventional packet-switched network (referred to as Conventional in the graphs) that features all the above-mentioned architectural parameters, except that it does not have parallel short packet forwarding and partitioned control VC.

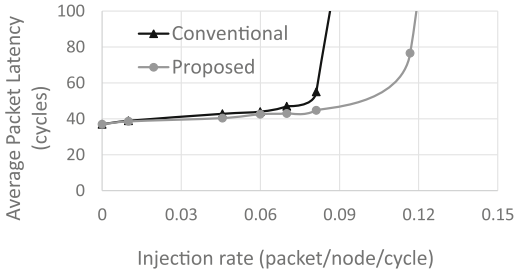
4.2 Performance Evaluation

Synthetic Traffic. First, we use a uniform traffic pattern to evaluate the network performance in different injection rates. The traffic is composed of 50% short (one 64-bit flit) and 50% long (five 128-bit) packets. Figure 4 shows the average packet latency for different injection rates. We consider 4×4 and 8×8 mesh networks with two half-width VCs for control and one full-width 128-bit VC for data packets.

As illustrated in the figure, our approach outperforms the baseline under most of the traffic injection rates. Furthermore, it pushes the saturation point by 22% for the 4×4 and 30% for the 8×8 NoCs.



(a)



(b)

Fig. 4. Average packet latency of the proposed and conventional NoCs in (a) 4×4 mesh and (b) 8×8 mesh

Under low traffic, the arbitration failure rate is low, so few flits benefit from parallel packet transfer that is used to resolve arbitration failures. Therefore, the latency approaches to the baseline latency. As the injection rate increases, however, the probability of arbitration failure increase that in turn, provides more opportunity for our proposed parallel packet transfer to improve performance. Consequently, the difference between the performance of the proposed NoC and the baseline increases under higher injection rates.

Realistic Workloads. Next, we evaluate the NoCs under four PARSEC and four GPU workloads. The experiments are done on an 8×8 mesh network for PARSEC and 5×5 for GPU.

Figure 5 shows the performance comparison results. The request and coherency packets have one 64-bit flit and data (response) packets have five 128-bit flits. In the GPU benchmarks some data packets have two 128-bit flits (together with the 5-flit packets). As the figure shows, performance is improved by 13%, on average for PARSEC and 24% for GPU. Again, the main source of better performance of our method is its ability to effectively use idle link bandwidth to remove many unnecessary control packet blocking situations. The GPU programs have considerably higher traffic loads than PARSEC, which translates to more efficiency of simultaneous packet forwarding.

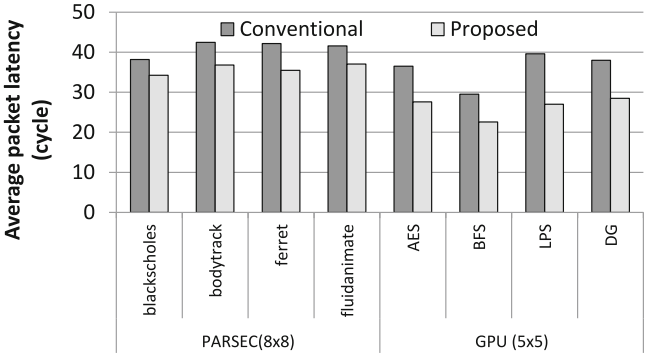


Fig. 5. Average packet latency comparison for realistic workloads

Cost Evaluation. The area overhead of the proposed architecture over the baseline is evaluated by synthesizing the VHDL description of our proposed router by a commercial synthesis tool in 45 nm technology. The amount of area overhead highly depends on the number of depth of virtual channels, but for configuration described earlier in this section the proposed NoC architecture increase the area of a baseline packet-switched NoC by 8%.

The area of control-path components of the proposed router, i.e. switch and VC allocators, are increased, but their total area has insignificant contribution to the entire router area (less than 7%). In the data-path side, the area of the buffers in the proposed router is roughly the same as a baseline conventional router with two VCs. However, the main source of area overhead in our design includes the extra multiplexers at the input port and additional crosspoint switches for the crossbar. Please note that the number of crossbar input and output ports, as well as the bit width of each port, which determines the crossbar layout and has the first-order effect on its area footprint, is the same as the baseline, but the crosspoints are doubled. Our synthesis shows that the modifications increase the crossbar's area from 22,900 um^2 to 24,500 um^2 (the total area of the modified router is 62,000 um^2).

Synthesis results in 45 nm technology also show that in the proposed router, the delay of route computation, switch allocation, VC allocation (two VCs), and crossbar traversal pipeline stages are 63 ps, 380 ps, 435 ps, and 254 ps, respectively. VC allocator has often the longest router pipeline latency, but our simple VC allocation scheme, where the message class determines the VC, leads to a simple and fast VC allocator logic.

As the results show, the latency of all stages is below 500 ps and so, the router can work at 2 Ghz, which is high enough as the working frequency of a high performance NoC.

Comparison with CCNoC. We also compare the proposed method with CCNoC [8]. The network parameters are the same as the previous experiments, but CCNoC has two physically separate sub-networks (128-bit data, 64-bit control) and uses a single VC per sub-network.

Figure 6 compares the average packet latency of the proposed NoC with CCNoC. The figure also compares the performance of CCNoC with a scaled-up version of the proposed NoC that has the same area as CCNoC. Our area analysis shows that by increasing the bit-width of the proposed NoC to 192 bits, it has a close area (within 5%) to an equivalent CCNoC. To simulate this bit-width, three short messages can pass a link simultaneously. Long (5-flit) packets also pass the wider links in four consecutive cycles (1.5 flits per cycle).

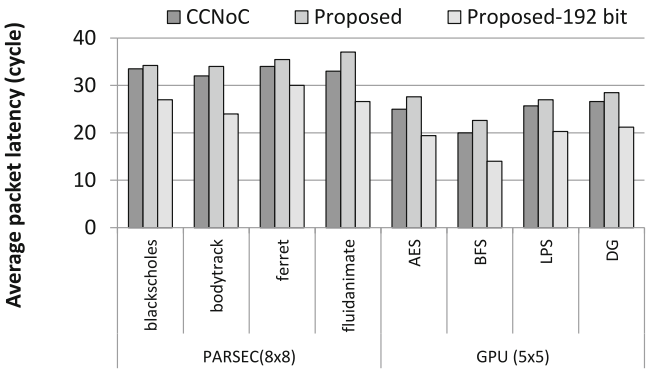


Fig. 6. Average packet latency comparison with CCNoC

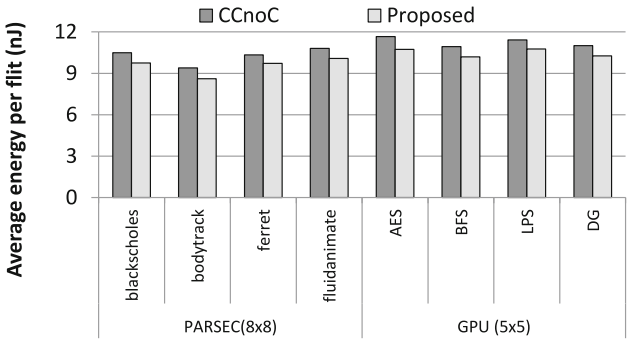


Fig. 7. Energy per flit (J) comparison with CCNoC

This configuration (represented by the bars marked as Proposed 192-bit in Fig. 6) actually shows the performance that the proposed parallel short packet forwarding would offer if the extra area overhead of CCNoC is invested to increase the bit-width of the proposed NoC.

As Fig. 6 shows, although the baseline proposed NoC suffers from an average performance loss of 7.5%, its area-normalized version can improve the performance by up to 21% over CCNoC.

The traffic load of many CMP applications is somewhat light and places between the zero load and saturation points. In CCNoC, this light traffic is further divided into

two lighter traffic loads. Consequently, the resources will be left underutilized and the performance is close to our proposal that utilizes the unused bandwidth of a single network to manage both control and data traffic.

Figure 7 compares the energy consumption of CCNoC with the proposed NoC. The results are obtained through the Dsent power library [21] and show that our NoC has 9% less energy usage, on average, mainly due to the less static power it wastes.

5 Conclusion

In this paper, we proposed a method to support parallel transfer and storage of short control flits in modern NoCs. In these NoCs, a large portion of bandwidth is wasted because a considerable part of packets consist of short control packets that are by far narrower than the link and buffer bit width. By the proposed parallel short flit sending the idle bit width is utilized to effectively reduce unnecessary control packet blocking latency. We showed that the proposed mechanism can be a more power and area-efficient alternative of the multiple physical sub-network schemes that has been used in some recent research and commercial NoC designs. One can consider the proposed NoC a polymorphic NoC architecture that integrates a wide data and a narrow control NoCs and has different bit widths from the point of view of different packet classes. The experimental results under a set of realistic and synthetic benchmarks revealed that this architecture can significantly reduce packet latency and improve throughput of NoCs.

References

1. Gratz, P., Kim, C., Sankaralingam, K., Hanson, H., Shivakumar, P., Keckler, S.W., Burger, D.: On-chip interconnection networks of the TRIPS chip. IEEE Micro **27**(5), 41–50 (2007)
2. Kumary, A., Kunduz, p., Singhx, A., Pehy, L.S., Jha, N.: A 4.6 Tbits/s 3.6 GHz single-cycle NoC router with a novel switch allocator in 65 nm CMOS. In: 2007 25th International Conference on Computer Design, Lake Tahoe, CA, pp. 63–70 (2007)
3. Howard, J., Dighe, S., Vangal, S.R., Ruhl, G., Borkar, N., Jain, S., Erraguntla, V., Konow, M., Riepen, M., Gries, M., Droege, G., Lund-Larsen, T., Steibl, S., Borkar, S., De, V.K., Van der Wijngaart, R.F.: A 48-core IA-32 processor in 45 nm CMOS using on-die message-passing and DVFS for performance and power scaling. IEEE J. Solid State Circ. **46**(1), 173–183 (2011)
4. Wentzlaff, D., Griffin, P., Hoffmann, H., Bao, L., Edwards, B., Ramey, C., Mattina, M., Miao, C., Brown III, J.F., Agarwal, A.: On-chip interconnection architecture of the tile processor. IEEE Micro **27**(5), 15–31 (2007)
5. Rotem, E., Naveh, A., Ananthakrishnan, A., Weissmann, E., Rajwan, D.: Power-management architecture of the Intel microarchitecture code-named Sandy Bridge. IEEE Micro **32**(2), 20–27 (2012)
6. Overview of Intel Xeon Phi Coprocessor. https://software.intel.com
7. Lee, J., Nicopoulos, C., Park, S.J., Swaminathan, M., Kim, J.: Do we need wide flits in networks-on-chip? In: ISVLSI, Natal, pp. 2–7 (2013)
8. Volos, S., Seiculescu, C., Grot, B., Pour, N.K., Falsafi, B., De Micheli, G.: CCNoC: specializing on-chip interconnects for energy efficiency in cache-coherent servers. In: Sixth International Symposium on Networks-on-Chip, Copenhagen, pp. 67–74 (2012)

9. Ma, S., Jerger, N.E., Wang, Z.: Whole packet forwarding: efficient design of fully adaptive routing algorithms for networks-on-chip. In: HPCA, New Orleans, pp. 1–12 (2012)
10. Badr, M., Jerger, N.E.: SynFull: synthetic traffic models capturing cache coherent behavior. In: 2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA), Minneapolis, MN, pp. 109–120 (2014)
11. Bienia, C., Kumar, S., Singh, J.P., Li, K.: The PARSEC benchmark suite: characterization and architectural implications. In: 17th International Conference on Parallel Architectures and Compilation Techniques, pp. 72–81. ACM, New York (2008)
12. Modarressi, M., Tavakkol, A., Sarbazi-Azad, H.: Application-aware topology reconfiguration for on-chip networks. IEEE Trans. Very Large Scale Integr. Circ. **19**(11), 2010–2022 (2011)
13. Bakhoda, A., Yuan, G.L., Fung, W.W.L., Wong, H., Aamodt T.M.: Analyzing CUDA workloads using a detailed GPU simulator. In: ISPASS, Boston, MA, pp. 163–174 (2009)
14. Hestness, J., Grot, B., Keckler, S.W.: Netrace: dependency-driven trace-based network-on-chip simulation. In: The Third International Workshop on Network on Chip Architectures (NoCArc 2010), pp. 31–36. ACM, New York (2010)
15. Yoon, Y.J., Concer, N., Petracca, M., Carloni, L.P.: Virtual channels and multiple physical networks: two alternatives to improve NoC performance. IEEE Trans. Comput. Aided Des. Integr. Circ. Syst. **32**(12), 1906–1919 (2013)
16. Mazloumi, A., Modarressi, M.: A hybrid packet/circuit-switched router to accelerate memory access in NoC-based chip multiprocessors. In: Design, Automation and Test in Europe *C*onference (DATE 2015), pp. 908–911 (2015)
17. Lotfi-Kamran, P., Modarressi, M., Sarbazi-Azad, H.: Near ideal network-on-chip for servers. In: 23rd IEEE Symposium on High Performance Computer Architecture (HPCA 2017), TX, USA (2017)
18. Cortex-A15 Technical Reference Manual. https://www.arm.com
19. HyperTransport Technology. https://www.amd.com
20. BookSim 2.0. https://nocs.stanford.edu/cgi-bin/trac.cgi/wiki/Re
21. Sun, C., Chen, C.H.O., Kurian, G., Wei, L., Miller, J., Agarwal, A., Peh, L.S., Stojanovic, V.: DSENT - a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling. In: NOCS, Copenhagen, pp. 201–210 (2012)