International Journal of Innovative Computing, Information and Control
Volume **15**, Number **1**, February **2019**

ICIC International ©2019 ISSN 1349-4198
pp. **305–319**

DESIGN CONCEPT AND MICROARCHITECTURE OF NETWORK-ON-CHIP WITH BEST-EFFORT AND GUARANTEED-THROUGHPUT SERVICES

Faizal Arya Samman[1,*] and Thomas Hollstein[2,3]

[1]Department of Electrical Engineering
Universitas Hasanuddin
Jl. Poros Malino Km. 6, Bontomarannu 92171, South Sulawesi, Indonesia
[*]Corresponding author: faizalas@unhas.ac.id

[2]Faculty of Informatics and Engineering Sciences
Frankfurt University of Applied Sciences
Nibelungenplatz 1, D-60318 Frankfurt am Main, Germany

[3]Department of Computer Systems
Tallinn University of Technology
Akadeemia tee 15A, 12618 Tallinn, Estonia

Received April 2018; revised August 2018

Abstract. *A network-on-chip (NoC) design concept that combines a best-effort (BE) and a guaranteed-throughput (GT) service in a single network platform is presented in this paper. The concept is enabled by a flexible flit-level packet interleaving method. Both BE and GT packets can share communication link in a flexible way, in which flits belonging to the same packet are assigned to the same local identity-tag (ID-tag). The ID-tags attached to every flit of packets will be changed/updated locally at runtime over communication links. The updating process is organized by an ID-tag mapping management unit implemented at every output port of the NoC routers. Compared to other existing multiplexing methods, our local ID-slot-based method provides high flexibility to establish connections with more optimal resources utilization. There is no need for a specific algorithm for finding a conflict-free scheduling as commonly used in the time-division multiple access-based methods that use time slots allocation technique. Communication channels can be shared effectively by both packets, where routing conflicts are simply managed using the proposed local ID-management method. Simulation results show that the BT and GT packets can be interleaved safely in the NoC and meet the expected bandwidth for each GT streams. From a selected test traffic scenario, all flits of both BE and GT streams can be routed and accepted correctly at each destination node without data loses.*
Keywords: Network-on-chip, Multicore processor, Best-effort communication, Guaranteed-throughput communication

1. **Introduction.** The new era of multi core or many core processor systems will come soon. The number of processing elements in a multi core platform will be more than 16 or even more than 100 cores in the forthcoming years. In this era, inter-core communication becomes a very crucial issue. Traditional bus systems cannot be a feasible solution to communicate the massive number of cores. In bus systems, higher number of cores will introduce worse bottleneck performance problem. Meanwhile, direct point-to-point communication among the cores is also not a plausible solution, since the communication routes will dominate the multi core system. Network-on-chip (NoC) will seem to be a promising solution. There are many network topologies that can be adopted such as

DOI: 10.24507/ijicic.15.01.305

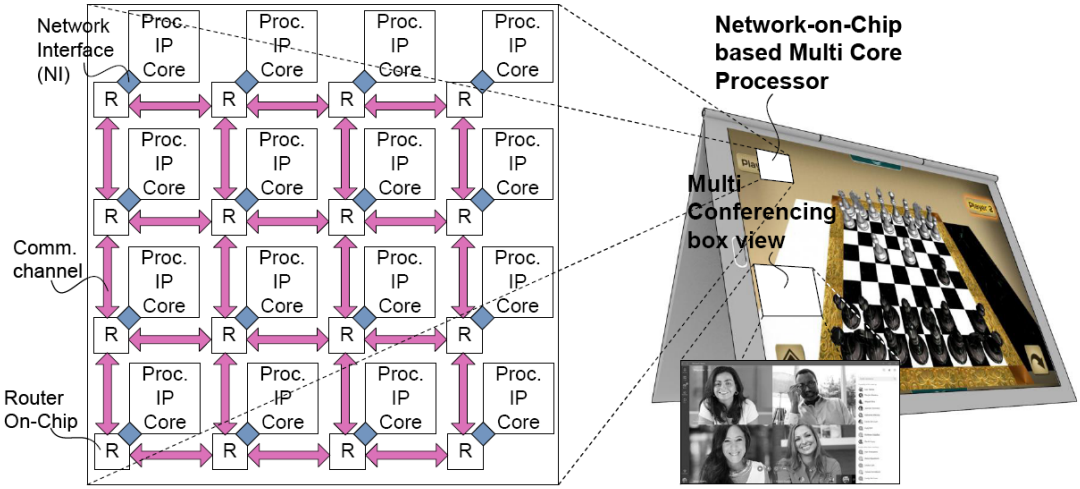


FIGURE 1. Modern Gadget with NoC-based multi core processor

tree-based, torus and mesh. Among them mesh network topology will be the most choice. This topology provides good bandwidth scale, where the increased number of cores will be followed by increased number of bandwidth availability. An example of multi core processor system (16 cores) in a mesh-based NoC topology is depicted Figure 1.

A modern application in the future will demand a high performance computation. In the NoC-based multi core platform, high performance and high quality communication infrastructure is also an important point beside the computing core itself. User satisfaction over software application (at application layer) can be met through inter-thread data throughput quality at network layer [1]. Therefore, quality-of-service (QoS) in the NoC will become an important issue. QoS can be implemented on several NoC communication layers. In physical layer, some problems such as crosstalk problem [2] and bit error problems [3, 4] become important issues. In routing and network layers, QoS can be applied by bundling application traffic into different classes of service [5]. To meet QoS requirements, i.e., global end-to-end minimum data rate or maximum time of end transmission, a resource allocation management is then applied. This paper will discuss the QoS in the routing and network layer with special attention to guarantee the minimum bandwidth requirement.

A modern electronic gadget, running multiple applications, is presented in Figure 1. The gadget computing power is supported by the NoC-based multi core processor system. The gadget can run more than one application such as game, teleconferencing via social media application and/or any other software applications. Teleconferencing is an example of multimedia applications that demands high data communication bandwidth with a relatively constant throughput. Hence, a particular service to guarantee the throughput of the data streaming is required. Otherwise, the teleconferencing cannot be run with acceptable inter-frame transition to display good quality video. In contrast to the GT service, another data communication service called best effort (BE) does not require the guaranty of the end-to-end constant throughput or bandwidth [6]. Other applications demanding non-guaranteed service can use this communication service. Combining both communication routing services in an NoC-based multi core platform is an interesting issue. This paper covers the issue with a specific feature of a simple way to combine BE and GT services in a single routing core with smaller buffer size.

The remaining sections of this paper are described as follows. Section 2 presents the related works and summarized contribution of this paper in the field of virtual circuit configuration methods for NoC. Section 3 presents the design concept. In that section, the

runtime circuit configuration, packet format and the routing mechanism are explained. Section 4 presents the NoC architecture implemented based on the concept. The routing protocols and communication services for both the BE and GT packets/streams are presented in Section 5. Section 6 presents the simulation results and their analysis. Section 7 concludes the paper contents.

2. **Related Works and Contribution.** QoS in NoC routing layer can be implemented using time-division multiple access (TDMA) method [7, 8, 9, 10], space-division multiple access (SDMA) [11], and code-division multiple access (CDMA) [12]. In the SDMA concept, the multiplexing is made based on the fact that NoC links are physically designed with a set of wires. The SDMA NoC router allocates a subset of wires to a given virtual circuit. The more wires (the larger the subset of wires) are allocated for a packet, the more the bandwidth (BW) it reserves. The concept of the CDMA NoC is implemented by introducing orthogonal spreading codes. The link can be shared by conflicting packets in which every bit of the packets is encoded and accumulated by a CDMA transmitter and carried by the spreading codes to the next router. However, combining BE and GT routing services in both SDMA and CDMA NoC routers requires complex efforts.

We propose a new concept of combining BE and GT routing services namely ID-tag-division multiple access (IDMA) method. The main difference of our IDMA method and the TDMA is the way to allocate BW. In TDMA method, packets/streams requesting more BW can reserve more time slots [13]. In the IDMA method, packets/streams requesting more BW can reserve more BW account from BW accumulator unit. Thus, our proposed IDMA method can assign more accurate BW space for each packet stream. Our proposed IDMA method does not require application mapping (time slot allocation) before application running to set up a virtual circuit configuration as needed by the TDMA method. Our proposed IDMA method establishes it at runtime. Hence, it does not add computing delay due to the pre-application computing.

Our NoC applies the concept of combining BE and GT service using a flit-level packet interleaving. The flit-level interleaving enables the implementation of arbitration scheme with fair input selection. The fair arbitration is difficult to apply in NoCs that do not use such flit-level interleaving, e.g., in [14]. Since different flits from different packets/streams can be interleaved on the same buffer, then buffer size can be set to a relatively small number of data slots.

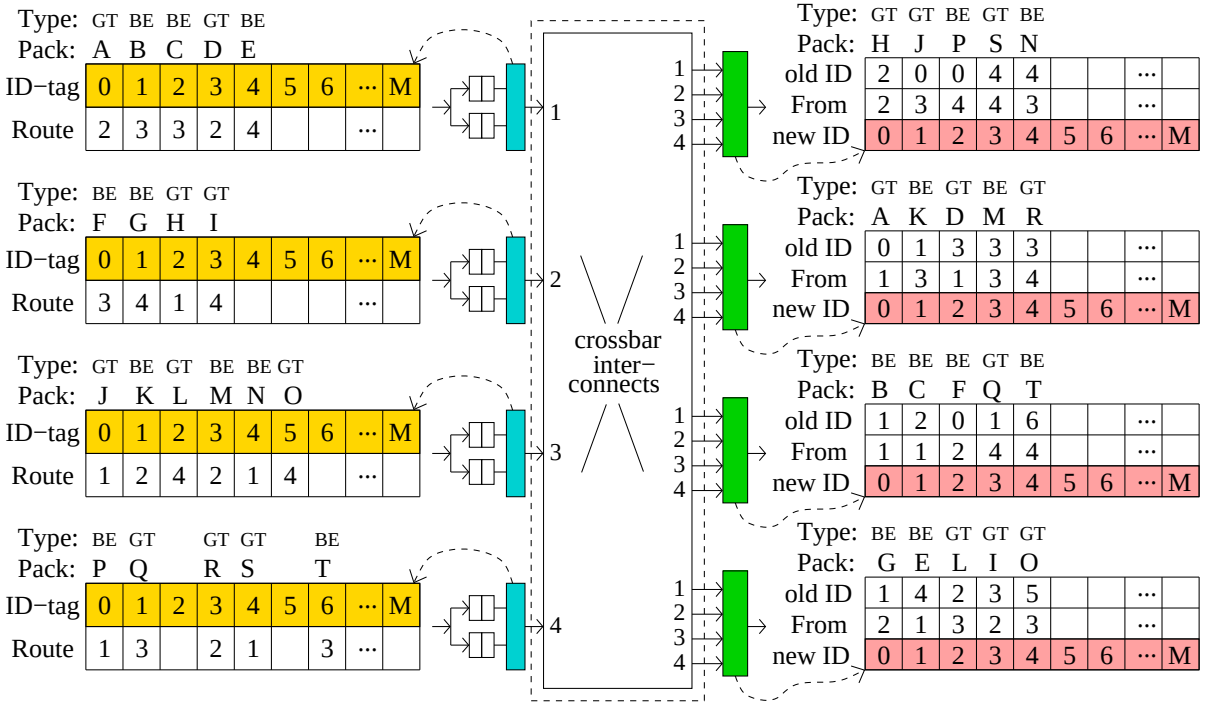
An NoC with some virtual channel buffers with a specific QoS level is presented in [15], which is implemented with a large number of first-in first-out (FIFO) buffers. Our proposed NoC provides only two FIFO buffers, i.e., one for BE packets and the other one for GT packets or streams, where each of them is built with only 2 data slots. Hence, the logic area of our NoC will be potentially and significantly lower.

Another NoC that combined the BE service using packet switching and the GT service using circuit switching on a two-layer NoC platform is presented in [16]. The work uses two-layer NoC that can induce a serious crosstalk problem and increase integration complexity. Our NoC combines the BE and GT service in a single layer NoC. Hence, it can surely result in a smaller logic area and lower static power dissipation.

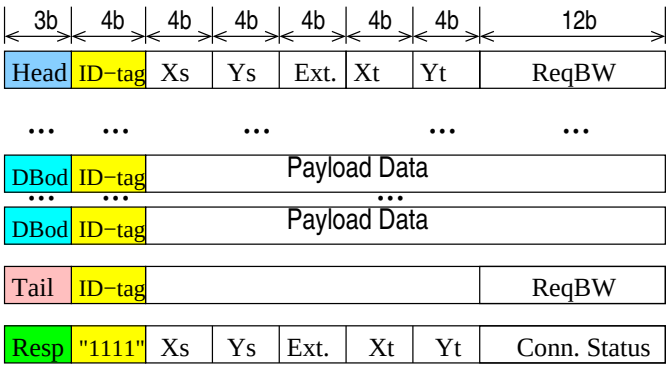
3. **The NoC Design Concept.** In this section, we will explain a few important aspects for the NoC design with combined BE and GT services. Subsection 3.1 explains the concept of the virtual circuit configuration based on the dynamic local ID management. Subsection 3.2 describes the packet format used to perform the BE and GT packets/streams. Subsection 3.3 explains the routing mechanism for each flit type of the BE and GT packets/streams.

3.1. **The local dynamic ID-based virtual circuit configuration.** The concept of the virtual circuit configuration (VCC) in the NoC is made based on the local dynamic ID-tag management. Each time a flit is routed to an advance router, and its ID-tag, which is attached on each flit is updated. Figure 2(a) illustrates the VCC concept applied to a 4-I/O port router. On the left side, we can see four programmable routing tables, where each of them is located at each input port. On the right side, it seems four reservable ID slot tables, where each of them is located at each output port.

Let us see for example the routing table at the upper left side in Figure 2(a). A packet with ID-tag number k is allocated to ID slot k. Hence, the routing table slots 0, 1, 2, 3 and 4, which are allocated for each packet A, B, C, D and E with ID-tag 0, 1, 2, 3 and 4, respectively. All packets are routed to the input port number 1. At each slot, there is an associated route data or routing direction. The routing mechanism will be explained in Subsection 3.3. Packet C for example is routed to output port number 3. On the left



(a) The packet switch concept



(b) The packet format

FIGURE 2. Concept of the locally Organized ID-based routing and its packet format

hand side of the figure, we can see that packet C appears at output port number 3. It uses a new ID-tag/slot number 1. Previously from input port number 1, it uses ID-tag/slot number 2. Both information is used to index the new ID-tag.

All flits belonging to the same group of packet C will be routed with the same ID-tag. The ID-tag update and reallocation are managed by an ID-management unit. Therefore, different packets can be interleaved at flit level on the same link. For M number of ID slot, the link can interleave M number of packet flows. However, it does not mean that we need M register slots for each FIFO buffer to apply the IDMA concept. By using this concept, we can even design the NoC router with only two register slots per FIFO buffer.

3.2. **Packet format for the BE and GT routing services.** The format of packets routed in the NoC is shown in Figure 2(b). The GT packet consists of four types of flits, i.e., header flit (Head), payload data flit (DBod), tail flit (Tail) and response flit (Resp). The BE packet consists of only two flit types, i.e., header flit (Head) and payload data flits (DBod).

Header flits carry the source address (Xs, Ys) and destination address (Xt, Yt) information as well as the expected or requested bandwidth (ReqBW) of a GT packet. The word length for the requested BW can be set arbitrarily, namely 8 until 12 bits. In 2(b), 12-bit word length for the ReqBW is used. Hence, the minimum and maximum decimal-coded digital BW values that can be used by packets/streams are 0 and $2^{12} - 1 = 4095$, respectively. If the maximum bandwidth of each link is $B_{\max}$ in flit per cycle unit or mega byte per second (MBps) and an individual packet/stream h is injected with data rate of $B_{inj}(h)$ also in flit per cycle unit or mega byte per second (MBps), then the ReqBW value for the packet is

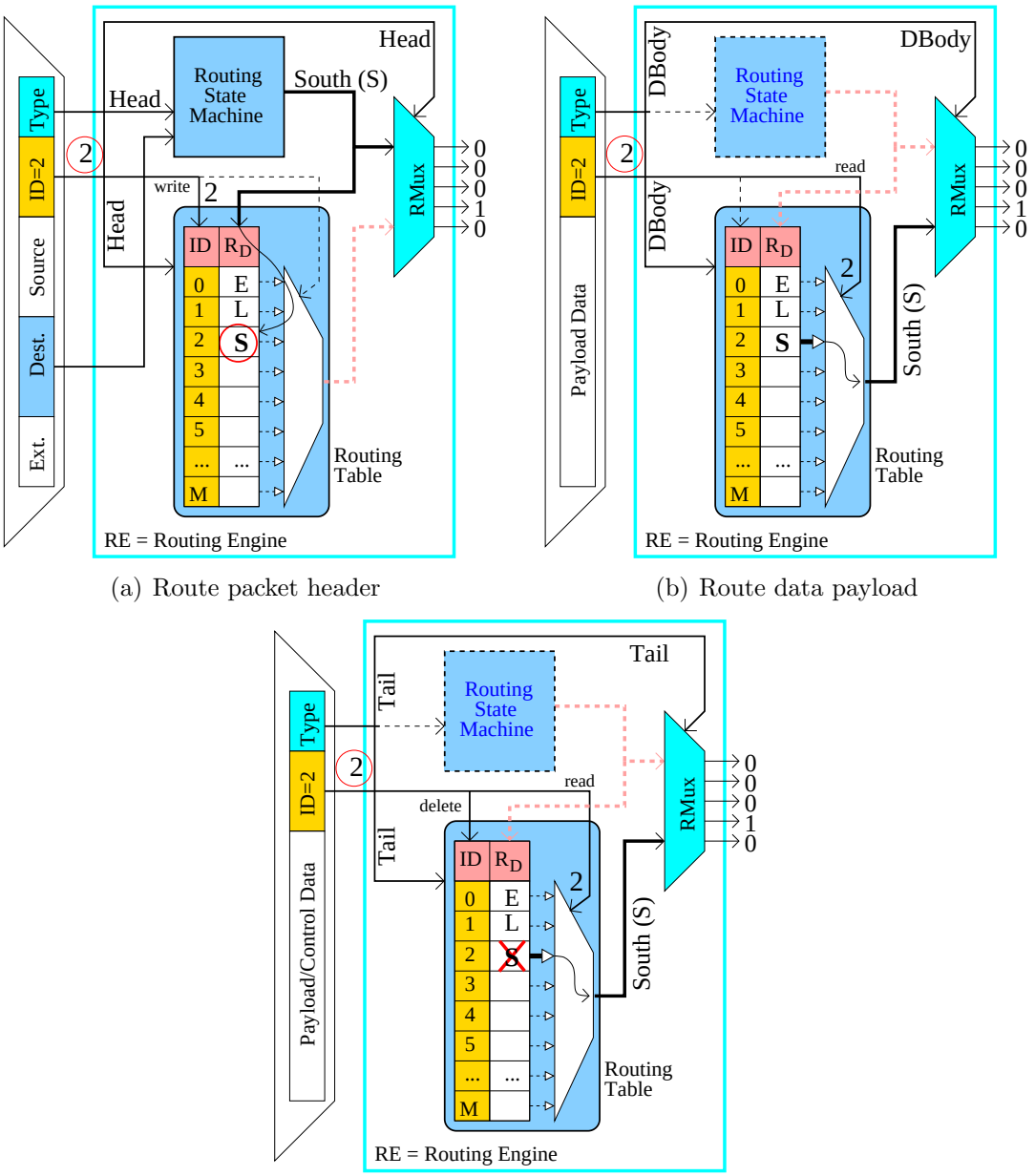
$$ReqBW = \frac{4095}{B_{\max}} B_{inj}(h), \qquad (B_{inj}(h) \leq B_{\max}) \tag{1}$$

A tail flit carries also the requested BW information, which is used to refresh or remove the BW allocation in a bandwidth accumulator unit at an output port. The header and tail flits of BE packets do not contain the expected or requested bandwidth information, since no bandwidth guarantee is given to the BE packets. The response flit is always assigned with ID-tag binary label "1111". It brings also the source and destination address as well as a connection status.

To differentiate routing protocol services for the BE packets and the GT packets/streams, the most left 3-bit field is identified with different binary codes. The binary code "000" is reserved to identify empty flits. The binary codes "001" and "100" are to identify the header flit for BE and GT packets, respectively. The binary codes "010" and "101" are to identify the data body flit for BE and GT packets, respectively. The binary codes "011" and "110" are to identify the tail flit for BE and GT packets, respectively. And, the binary code "111" is to identify the response/status flit for the GT packets, respectively.

3.3. **The routing mechanism.** The routing engine used in the NoC consists of two main parts, i.e., a routing state machine and a runtime programmable routing table. The routing mechanism made by the NoC routing engine is based on the selective use between the programmable routing slot table and the routing state machine. The multiplexing of both routing units is controlled by the flit type of a packet. Figure 3 presents the routing mechanism for three types of the flits, i.e., header, payload or data body and tail flit. The following items describe the mechanism.

- Figure 3(a) presents the routing process for a header flit. When a header flit is identified by the routing engine, the routing engine will compute a routing direction for the header in accordance with target address fields attached on the header flit.



(a) Route packet header

(b) Route data payload

(c) Route packet tail

FIGURE 3. ID-based routing mechanism

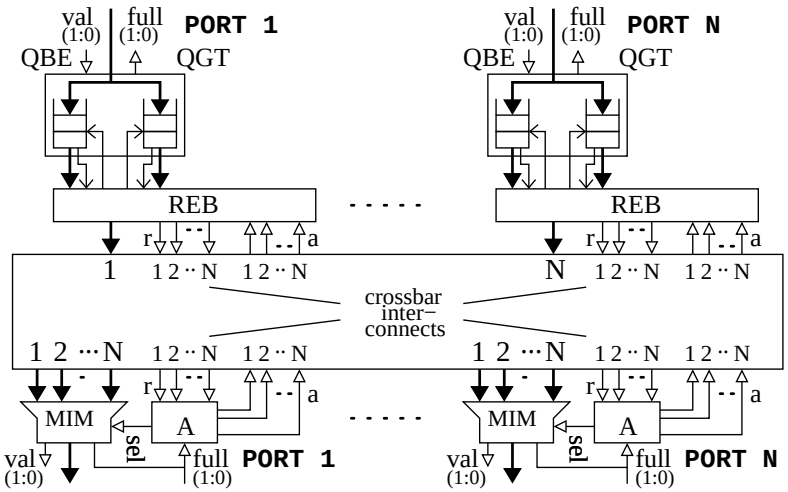
This routing direction is then used to route the header, and is stored in a slot index according to its ID-tag number in the programmable routing slot table. As shown in Figure 3(a), the routing state machine computes the routing direction of the header having ID-tag number 2. The computed routing direction, i.e., South (S) direction, binary encoded as "00010", is then stored in the slot number 2 (according to the header's ID-tag), and the Rmux unit selects the routing direction from the routing state machine.

- Figure 3(b) presents the routing process for a payload or data body flit. When a payload flit is identified by the routing engine, the routing engine will look for the routing direction from the routing slot table, i.e., exactly from the slot number, which is the same as the ID-tag number of the payload flit. As shown in Figure 3(b), the Rmux unit selects the routing direction from the routing slot table, exactly

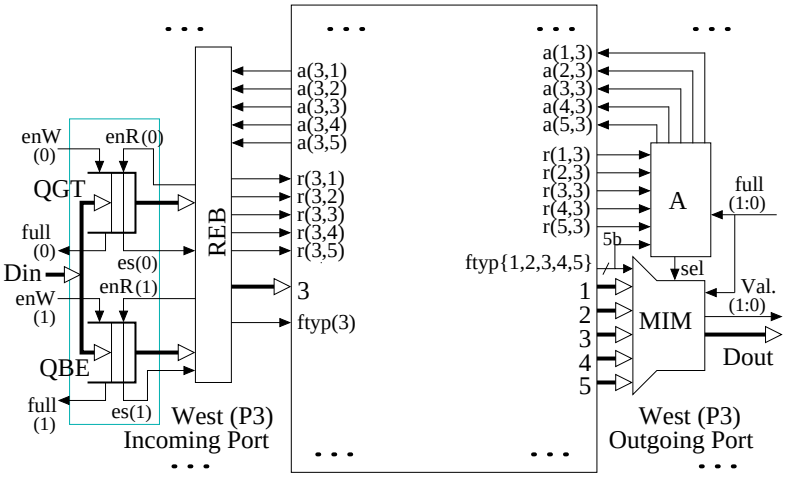
fetched from the slot number 2 according to the payload's ID-tag. Remember that the payload belongs to the same packet with the previously routed header (shown in Figure 3(a)). Therefore, the payload flit has the same ID-tag number as the header's.

- Figure 3(c) presents the routing process for a tail flit. When a tail flit is identified by the routing engine, the routing mechanism is similar to the routing process for the payload flit, but it is followed by an additional process, i.e., the tail flit will remove the routing direction from the routing table slot number in accordance with the tail flit's ID-tag number. Remember again that tail flit belongs to the same packet with the previously router header and payload flit as presented in Figure 3(a) and Figure 3(b), respectively. Hence, they have the same ID-tag number, i.e., tag number 2. As shown in Figure 3(c), the tail flit having ID-tag number 2 removes the routing direction from the slot number 2 of the routing slot table.

4. **The NoC Router Microarchitecture.** The microarchitecture of the router is presented in Figure 4. The router is designed with modular-oriented method, where each modular component is regularly instantiated for each input-output port. The NoC in general, consists of three components in incoming port, i.e., an FIFO buffer for best-effort



(a) Combined GT+BE implementation



(b) Example west I/O port

FIGURE 4. The VLSI architecture for combined BE+GT routing services

(BE) messages (*QBE*), FIFO buffer for guaranteed-throughput (GT) messages (*QGT*) and a *routing engine with multiplexed data buffering* (*REB*). In each outgoing port, there are two components, i.e., a *multiplexor with ID-tag management unit* (*MIM*) and an *arbiter* (*A*) unit. In order to keep the router size small, the depth of each virtual channel is set to 2.

The depth of the FIFO buffer for BE and GB messages can be made equal, namely 2 register slots for example. A soft guarantee is given to the GB-type packets/streams in the virtual buffers placed at the input port. When GT-buffer and BE-buffer are occupied by the GT-type flits or BE-type flits at the same time, respectively, then the routing engine will route firstly the data flit in the GT-buffer until the GT-buffer is empty.

The ID management unit plays an important rule to interleave flits of different message in the same queues and to perform the flexible runtime communication resources reservation. The ID management unit is implemented in the *MIM* component at each output port. The detailed interconnected data and 1-bit control nets in the crossbar switch are presented in Figure 4(b). The arbiter unit selects a data flit from input ports, which will be switched to the output port of the *MIM* module.

Because of using the wormhole cut-through switching with the ID-slot management, a contention between two data flits to acquire a similar outgoing channel can occur. Therefore, our NoC is also equipped with a link-level control to avoid data overflow in the NoC. When a contention happens, FIFO queues occupied by the contenting data flits at incoming ports will be busy or might be full. The congestion (full condition) signals are then traced back to the upstream nodes to avoid other data flits entering the congested FIFO queues. Figure 4(b) presents the full flag (*ff*) signals from FIFO queues in one router to the module *A* (arbiter) and module *MIM* (multiplexor with ID-management unit) in the neighbor router.

5. **Data Communications Service.**

This section explains the communication services used in our NoC router, i.e., the connection-oriented guaranteed-throughput (GT) service and best-effort (BE) communication service.

5.1. **Guaranteed-throughput (GT) communication service.**

The process to establish and to terminate connection for the guaranteed-throughput packet/stream is described into 4 sequential phases as depicted in Figure 5. Core A sends a data stream to core B via the NoC.

1) To initiate a connection with an expected BW, core A injects a request flit to the NoC. The request flit is then acccepted by core B as shown in Figure 5(a).
2) Afterwards, core B analyzes the request flit to find out whether the requested connection with the expected BW is successful or not. Core B will send a response flit as presented in Figure 5(b) to tell core A the connection process.
3) If the connection with guaranteed end-to-end throughput from core A to core B is successfully established as known from the response flit sent back by core B to core A, then, as depicted in Figure 5(c), core A will start sending the data stream to core B with the expected throughput or BW.
4) However, the request flit fails to establish connection or guarantee the expected data communication BW between core A to core B, and then core A would terminate the connection by sending a tail flit to remove the reserved communication resources as presented in Figure 5(d). Afterwards, core A will start again to send a new request to establish a new connection.

The requested connection can fail because there is no more available ID-slot in certain communication resources in intermediate routers or there is no enough reservable BW to

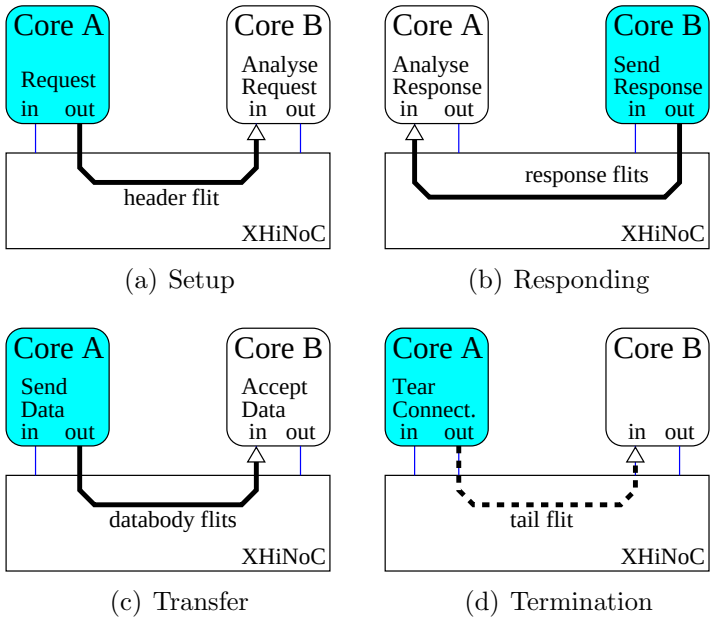


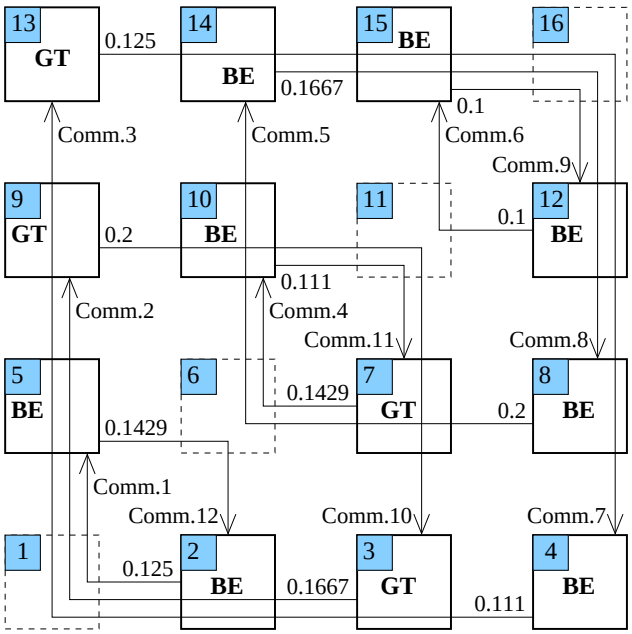
FIGURE 5. Connection setup with progressive approach for connection termination

guarantee end-to-end data throughput. Our current router implementation uses 4 bits for ID-tag field. It means that there are 16 ID slots available on each communication resource. However, we use only 15 ID slot for communication and the remaining one ID slot is reserved to control the flow of header flits which flow in the links that run out of ID slots. In the design, we use ID-tag "1111" as the control ID-tag. For instance, if a header flows through a link that runs out of ID slot, then the header will be assigned with the ID-tag "1111". Once a header is assigned with the ID-tag "1111", then it will be always assigned with the ID-tag "1111" on each communication link until it reaches its destination node.

5.2. **Best-effort (BE) communication service.** Beside the GT packets, the NoC can also route a BE packet. A best-effort data communication is made without making firstly a connection setup (connectionless). A best-effort databody and a last databody (a tail flit) are sent by following its header flit injected in advance without waiting for a response flit from the destination node. Hence, there is a different mechanism to handle a packet that cannot reserve an ID slot in a certain intermediate node.

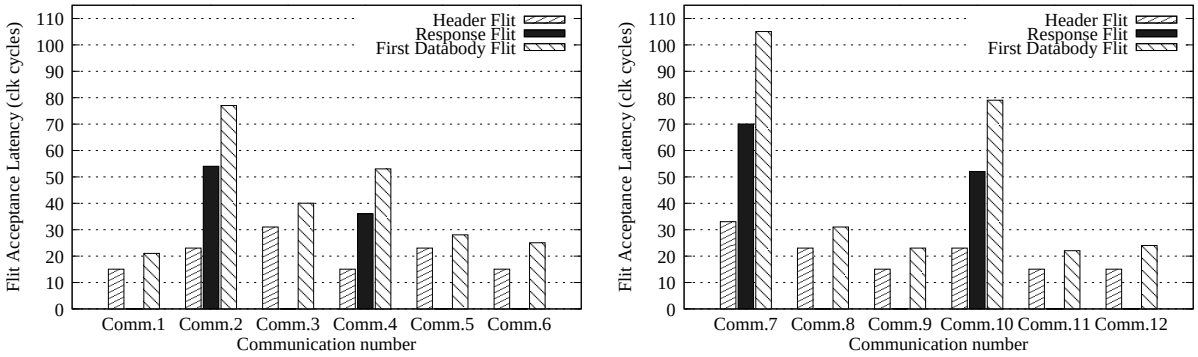
6. **Simulation Result and Analysis.** In this section, an experimental simulation is run in which BE and GT messages are mixed in the matrix transpose traffic scenario (node (i, j) send a message to node (j, i)). As shown in Figure 6, there are 12 communication pairs in the transpose traffic pattern, i.e., from Comm. 1 until Comm. 12. The Comm. 2, Comm. 4, Comm. 7 and Comm. 10 are set as GT-type injector-acceptor communication, while the remaining 8 communication pairs are set as BE-type injector-acceptor communication. A node symbolized with **BE** is a node sending a BE message, while a node symbolized with **GT** is a node sending a GT message.

In the simulation, the workload sizes (the number of injected messages per producer) are set to 500, 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000 and 10000 flits. The decimal values outside the source nodes (beside the paths of each communication pair) presented in Figure 6 are the expected data communication rates measured in an amount of flits per cycle (fpc). On the right side of Figure 6, we can also see a table



Comm.#	Type	Rate in fpc	Rate in MB/s
Comm.1	BE	0.125	400.00
Comm.2	GT	0.1667	533.33
Comm.3	BE	0.111	355.55
Comm.4	GT	0.1429	457.28
Comm.5	BE	0.2	640.00
Comm.6	BE	0.1	320.00
Comm.7	GT	0.125	400.00
Comm.8	BE	0.1667	533.33
Comm.9	BE	0.1	320.00
Comm.10	GT	0.2	640.00
Comm.11	BE	0.111	355.55
Comm.12	BE	0.1429	457.28

FIGURE 6. Mixed GT-BE message data transmissions in the transpose traffic scenario



(a) Comm. 1-6 (b) Comm. 7-12

FIGURE 7. The transfer latency (delay of acceptance) of the header, response and the first databody flits

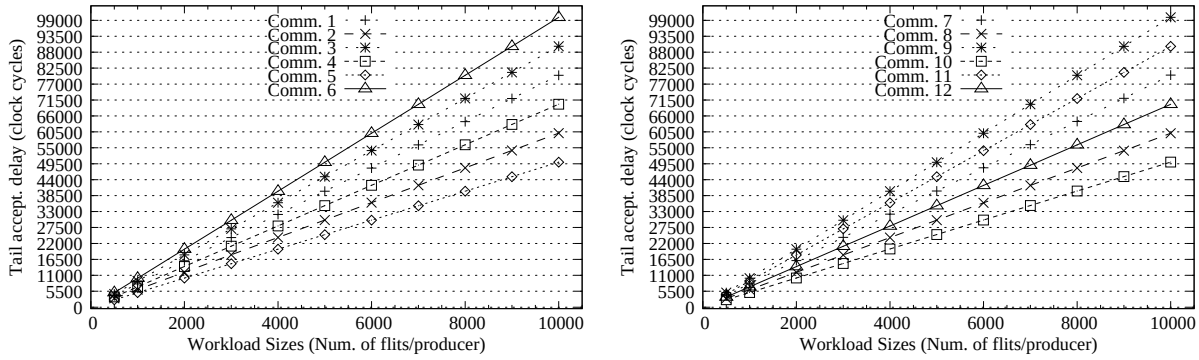
showing the expected bandwidth in flits/cycle (fpc) unit and megabyte/cycle (MB/s) unit for each communication pair. In the simulations, the NoC is clocked in such a way that the maximum bandwidth of each link is 1600 MB/s. The maximum link capacity of the NoC prototype is 0.5 fpc, or with 800 MHz data clock cycle frequency of the NoC router prototype with combined BE-GT services and 32-bit data word, the maximum link capacity is 0.5 fpc $\times$ 4 Byte $\times$ 800 MHz = 1600 MByte/s or 1.6 GByte/s. For example, Comm. 1 with BE communication protocol is expected to be injected from source nodes with 0.125 fpc, which is equivalent to $0.125 \times 4 \times 800 = 400$ MB/s.

Figure 7 presents the measurement of the delay (acceptance latency) of the header, the first databody flit and the response flits in clock cycle period of each communication pair. The response flits will exist only for the GT communication pairs, i.e., Comm. 2, Comm. 4, Comm. 7 and Comm. 10. Figure 7(a) shows the latency measurement for Comm. 1 until Comm. 6, while Figure 7(b) presents the latency measurement for Comm. 7 until Comm. 12. The transfer delay of the header flit is measured from its injection node

until its destination node. While the transfer delay of the response flit is measured from the destination node until the injection node and is accumulated with the transfer delay of the header flit. The transfer delay of the first databody flit is measured from the injection node until the destination node and is accumulated with the previously measured transfer delays of the header and response flits.

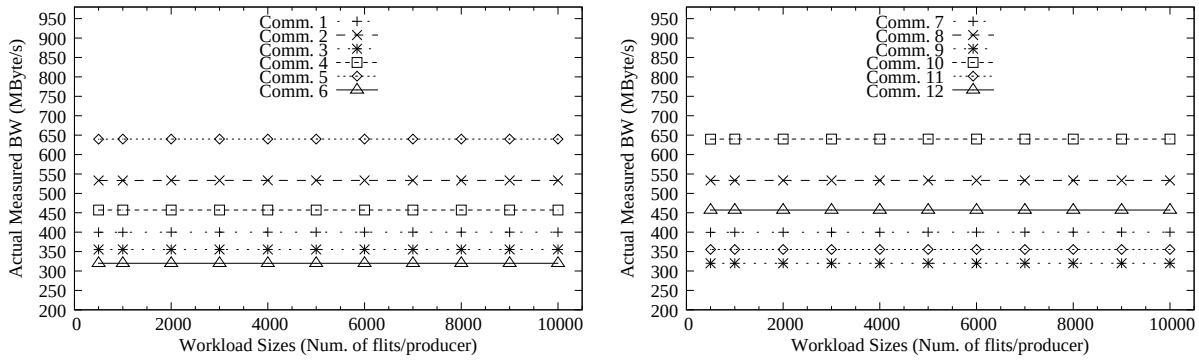
The measurement of the tail acceptance delays with different workload sizes for each communication pair is presented in Figure 8. Figure 8(a) shows the tail acceptance latency measurements for Comm. 1 until Comm. 6, while Figure 8(b) shows the tail acceptance latency measurements for Comm. 7 until Comm. 12. In general, it looks that the tail flit acceptance latency values are increased linearly when the workload (data burst) sizes are incremented.

The measurement of the actual communication bandwidth with different workload sizes for each communication pair is presented in Figure 9. Figure 9(a) shows the actual communication bandwidth measurements for Comm. 1 until Comm. 6, while Figure 9(b) shows the actual communication bandwidth measurements for Comm. 7 until Comm. 12. In general, it looks that the actual communication bandwidths are constant when the workload (data burst) sizes are incremented. The slopes of the tail flit transfer latencies of each communication pair presented in Figure 8(a) and Figure 8(b) have relationship with the communication bandwidth measurements presented in Figure 9(a) and Figure 9(b).



(a) Comm. 1-6 (b) Comm. 7-12

FIGURE 8. The tail acceptance delays with different workload sizes for each communication pair

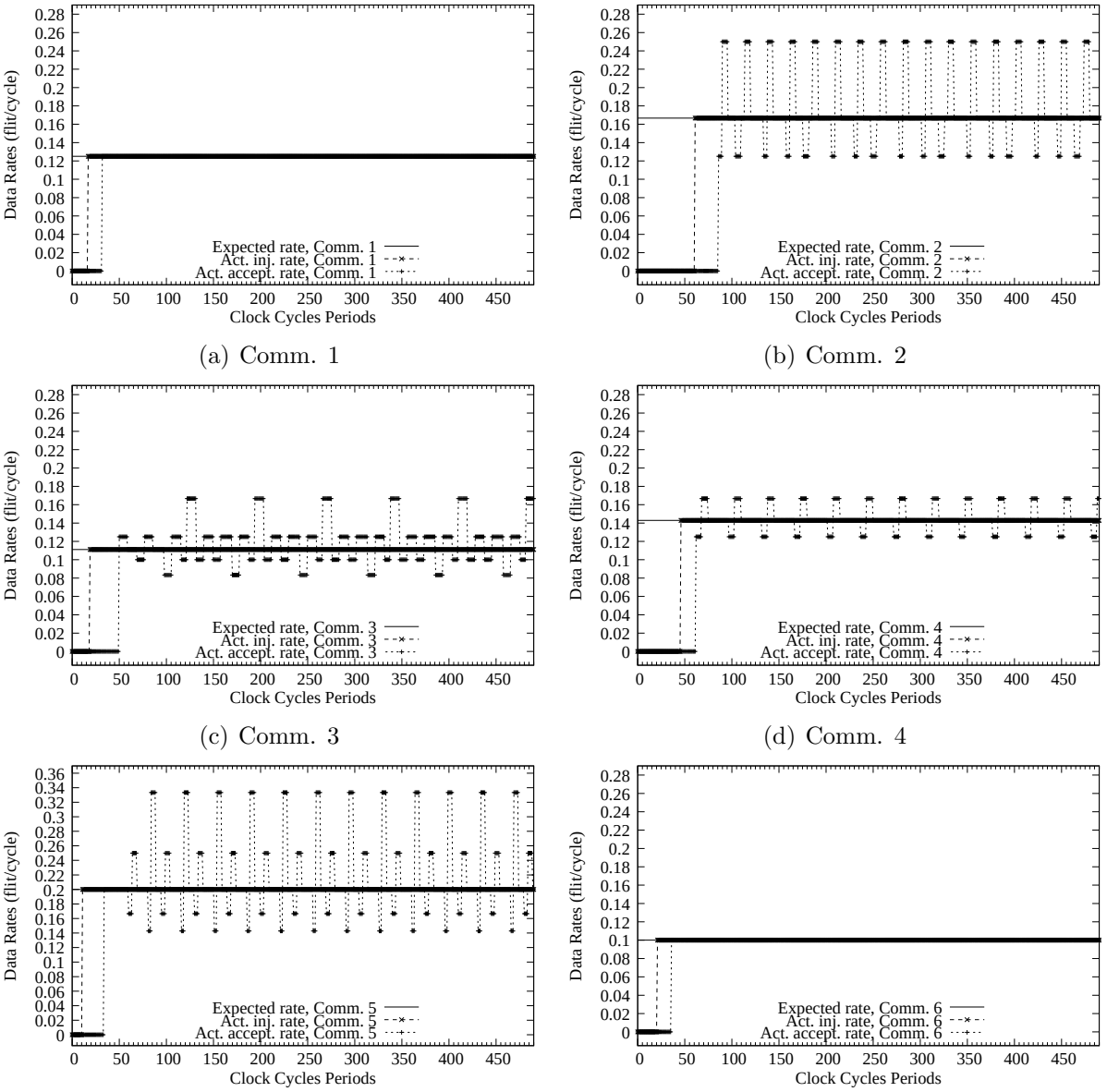


(a) Comm. 1-6 (b) Comm. 7-12

FIGURE 9. The actual communication bandwidth measurement with different workload sizes for each communication pair

The larger the slopes are, the larger the communication bandwidth of the communication pairs is.

Figure 10 presents the transient responses of the actual injection and acceptance rates as well as the expected constant data rate for Comm. 1 until Comm. 6, while Figure 11 shows the transient responses for Comm. 7 until Comm. 12. The simulations in this section have exhibited a very interesting characteristic of the NoC that performs a very flexible runtime communication resource reservation to serve both the BE and GT messages. The BE and GT messages are switched through virtual circuit configurations. The expected data communication rates in the simulation are set in such a way that the NoC is not saturated. Therefore, in line with the general performance characteristic of the NoC, the communication latency is increased linearly with the workload size incrementation,

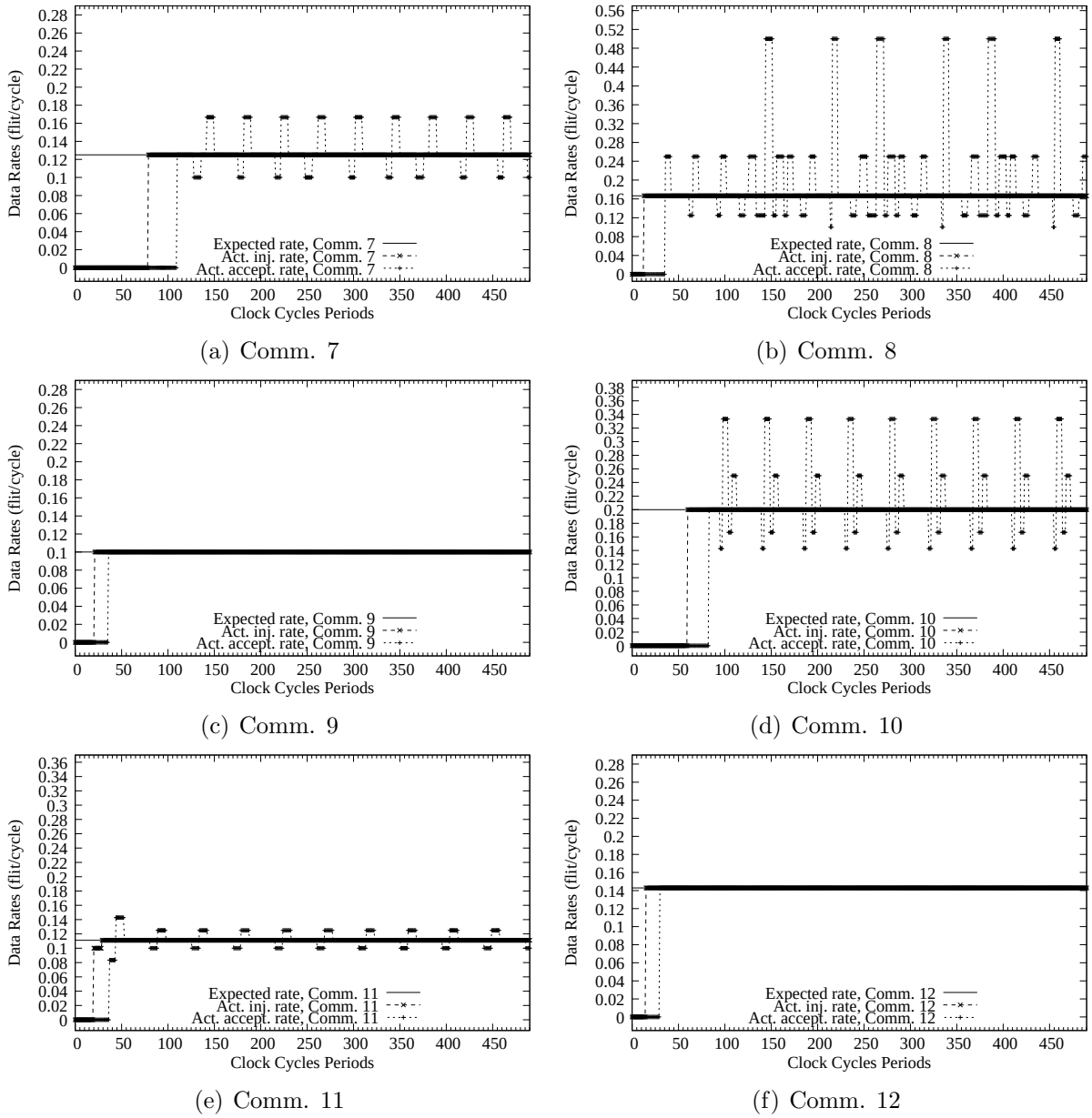


(a) Comm. 1 (b) Comm. 2

(c) Comm. 3 (d) Comm. 4

(e) Comm. 5 (f) Comm. 6

FIGURE 10. Transient responses of the measured data injection and data acceptance rates for communication 1-6



(a) Comm. 7 (b) Comm. 8

(c) Comm. 9 (d) Comm. 10

(e) Comm. 11 (f) Comm. 12

FIGURE 11. Transient responses of the measured data injection and data acceptance rates for communication 7-12

and the communication bandwidth can be kept constant even if the workload sizes are incremented.

Due to the non-saturating condition, the expected bandwidth of every communication pair can be fulfilled. The data acceptance in the experimental results is also lossless, i.e., all injected flits in source nodes are accepted in the target nodes. Although some overshots of the actual measured data acceptance rates appear as shown in Figure 10 and Figure 11, the total average communication bandwidth of every end-to-end communication partner is guaranteed equal to the expected constant data rate. We can see that the acceptance rate of every communication partner fluctuates around the expected constant data rate, but the actual measured injection rate is always equal to the expected constant data rate. The overshots are due to contentions between messages to access the same link in the NoC.

7. **Conclusions.** The NoC routers with guaranteed-throughput (GT) and best-effort (BE) routing services have been presented in this paper. Our proposed concept and microarchitecture to combine both the best-effort (BE) and the guaranteed-throughput (GT) or guaranteed-bandwidth (GB) service can be implemented easily. The interesting feature of the service-combination in our NoC is that the flits of the BE-type and GT-type packets can be mixed and interleaved in the same NoC communication channel. Hence, it enables us to implement a simple data buffering scheme. The need for smaller buffer number and size will potentially make the logic area and static power dissipation of our proposed NoC smaller and lower.

From the simulation results of the selected test traffic scenario, we can see that the BT and GT packets can be interleaved in the NoC. The expected BW for each GT stream can be guaranteed. All flits of both BE and GT streams can be routed and accepted correctly at each destination node without data loses. In general, we can conclude that mixing BE and GT packets in our NoC can be performed simply and effectively.

In our IDMA method, BW characteristic is very special. The total available bandwidth can be set freely in accordance with the available ID slots. In other words, a single ID slot can be allocated with a single variable or different BW space. This unique scheme enables us to apply large varieties of BW management policy on top of application layers in many core processor systems. In addition, by using the IDMA method, BE packets can fully use BW resources in the absence of GT packets/streams.

Acknowledgement. We gratefully acknowledge the Ministry for Research, Technology and Higher Education of the Republic of Indonesia (by Direktorat Riset dan Pengabdian Masyarakat, DRPM) for funding and supporting our research work under the scheme of "The National Strategic Outstanding Research Grant", (*Hibah Penelitian Unggulan Strategis Nasional* or PUSNAS) in the years 2017 and 2018.

REFERENCES

[1] S. D. Ponpandi and A. Tyagi, User satisfaction aware routing and energy modeling of polymorphic network on chip architecture, *Computers & Electrical Engineering*, vol.40, no.8, pp.260-275, 2014.

[2] A. Ganguly, P. P. Pande and B. Belzer, Crosstalk-aware channel coding schemes for energy efficient and reliable NOC interconnects, *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol.17, no.11, pp.1626-1639, 2009.

[3] A. Ejlali, B. M. Al-Hashimi, P. Rosinger, S. G. Miremadi and L. Benini, Performability/energy tradeoff in error-control schemes for on-chip networks, *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol.18, no.1, pp.1-14, 2010.

[4] A. Vitkovski, A. Jantsch, R. Lauter, R. Haukilahti and E. Nilson, Low-power and error protection coding for network-on-chip traffic, *IET Computers & Digital Techniques*, vol.2, no.6, pp.483-492, 2007.

[5] C. Wang and N. Bagherzadeh, Design and evaluation of a high throughput QoS-aware and congestion-aware router architecture for network-on-chip, *Microprocessors and Microsystems – Embedded Hardware Design*, vol.38, no.4, pp.304-315, 2014.

[6] D. Rahmati, S. Murali, L. Benini, F. Angiolini, G. D. Micheli and H. Sarbazi-Azad, Computing accurate performance bounds for best effort networks-on-chip, *IEEE Trans. Computers*, vol.62, no.3, pp.452-467, 2013.

[7] M. Millberg, E. Nilsson, R. Thid and A. Jantsch, Guaranteed-bandwidth using looped containers in temporally disjoint networks within the nostrum network on chip, *Proc. of Design Automation and Test in Europe (DATE'04)*, pp.890-895, 2004.

[8] Z. Lu and A. Jantsch, TDM virtual-circuit configuration for network-on-chip, *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol.16, no.8, pp.1021-1034, 2008.

[9] S. Evain, J.-P. Diguet and D. Houzet, NoC design flow for TDMA and QoS management in a GALS context, *EURASIP Journal on Embedded Systems*, vol.2006, pp.1-12, 2006.

[10] U. M. Mirza, F. Gruian and K. Kuchcinski, Mapping streaming applications on multiprocessors with time-division-multiplexed network-on-chip, *Computers and Electrical Engineering*, vol.40, no.8, pp.276-291, 2014.

[11] A. Leroy, D. Milojevic, D. Verkest, F. Robert and F. Catthoor, Concepts and implementation of spatial division multiplexing for guaranteed throughput in networks-on-chip, *IEEE Trans. Computers*, vol.57, no.9, pp.1182-1195, 2008.

[12] X. Wang, T. Ahonen and J. Nurmi, Applying CDMA technique to network-on-chip, *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol.15, no.10, pp.1091-1100, 2007.

[13] J. Heisswolf, R. Koenig, M. Kupper and J. Becker, Providing multiple hard latency and throughput guarantees for packet switching networks on chip, *Computers and Electrical Engineering*, vol.39, no.8, pp.2603-2622, 2013.

[14] A. Kostrzewa, S. Saidi, L. Ecco and R. Ernst, Ensuring safety and efficiency in networks-on-chip, *Integration*, vol.58, pp.571-582, 2017.

[15] R. Akbar, F. Safaei and S. M. S. Modallalkar, A novel power efficient adaptive RED-based flow control mechanism for networks-on-chip, *Computers and Electrical Engineering*, vol.51, pp.121-138, 2016.

[16] Y. Li, K. Mei, Y. Liu, N. Zheng and Y. Xu, Application-driven dynamic bandwidth allocation for two-layer network-on-chip design, *Computers and Electrical Engineering*, vol.40, no.8, pp.317-332, 2014.