IPSJ Transactions on System LSI Design Methodology Vol.8 85–94 (Aug. 2015)

[DOI: 10.2197/ipsjtsldm.8.85]

Regular Paper

A Performance Enhanced Dual-switch Network-on-chip Architecture

Lian Zeng[1,a)] Xin Jiang[1,b)] Takahiro Watanabe[1,c)]

Received: December 4, 2014, Revised: March 12, 2015,
Accepted: April 29, 2015, Released: August 1, 2015

Abstract: With rapid progress in semiconductor technology, Network-on-Chip (NoC) becomes an attractive solution for future systems on chip (SoC). The network performance depends critically on the performance of packets routing. The delay of router and packets contention can significantly affect network latency and throughput. As the network becomes more congested, packets will be blocked more frequently. It would result in degrading the network performance. In this article, we propose an innovative dual-switch allocation (DSA) design. By introducing DSA design, we can make utmost use of idle output ports to reduce packets contention delay, meanwhile, without increasing router delay. Experimental results show that our design significantly achieves the performance improvement in terms of throughput and latency at the cost of very little power and area overhead.

Keywords: network-on-chip, dual-switch, performance enhanced

1. Introduction

As the semiconductor technology continues to develop, hundreds of cores will be deployed on a signal die in future Chip-Multiprocessors (CMPs) designs. In order to solve the problem of bus based system such as increasing power consumption, the limitation of bandwidth and scalability, the promising packet-switch Network-on-Chip (NoC) has become an attractive solution which can provide low latency, high throughput and low power [1]. Besides different routing algorithms, the router architecture can significantly affect the network performance depends on router delay and packets contention delay. Therefore researchers are continuously confronted by one of major challenges: as the whole network becomes more congested, packets will be blocked more frequently and contention delay increases rapidly. As a result, the network performance is degraded, and it should be considered how to transfer more packets through a router without increasing router delay. On the other hand, power dissipation is also an important factor. Input buffers could consume almost 46% of the power of the whole interconnection network [2]. Therefore, even though simply increasing the size of input buffers will lead to more packets being transmitted and buffered, power increases with the number of buffers. Meanwhile, the area of router will increase rapidly as well.

In this paper, a high performance and power modest dual-switch allocation design is proposed. In order to transmit and buffer more packets, the design becomes a combination of a primary switch allocation (PSA) and a secondary switch allocation (SSA) with no additional input buffers, as the power consumption of buffers dominates the whole power of network and additional input buffers burden router area. In our proposed method, dual-switch allocation is a logical conception, actually, the same switch hardware is reused for both switch allocation to guarantee minimum hardware cost. At low traffic load, almost all packets utilize the PSA to assign their desired output port. Whenever there is a conflict, the packet which fails in the PSA will be assigned to other corresponding idle output port by the SSA. As a result, the dual-switch allocation design enables blocked packets to transmit through router via idle output port as far as possible, thus achieving high throughput and low latency by reducing packets contention delay. On the other hand, power overhead is very little, as there are no additional input buffers and links in our design.

This paper is organized as follows: In Section 2, a brief overview of related work is presented. Proposed dual-switch allocation design is presented in Section 3. Experimental setup and results will be demonstrated in Section 4.

2. Related Work

Various techniques have been proposed to transmit and buffer more packets, considering balance of congestion of interconnection network. J. Suseela and V. Muthukumar proposed a loop-back virtual channel mechanism to improve the performance of a router [3]. This design could minimize latency by additional virtual channel, at the cost of increased power consumption and complexity. Another approach dividing the router into two subnets was Network Processor Array (NePA) [4], which utilized additional input ports including buffers and links in north and south directions. In this case, NePA could separate and transmit the packets which desire north or south direction. However, its power

[1] Graduate School of Information, Production and Systems, Waseda University, Kitakyushu, Fukuoka 808–0135, Japan
[a)] lian_zeng_ips@ruri.waseda.jp
[b)] jiangxin@ruri.waseda.jp
[c)] watt@waseda.jp

© 2015 Information Processing Society of Japan

consumption becomes large due to the increase of input buffers and links. Other techniques in Refs. [5], [6], [7] were also proposed in recent years, however some of them were based on additional input buffers to improve the network performance, so that power consumption remains as a challenging problem.

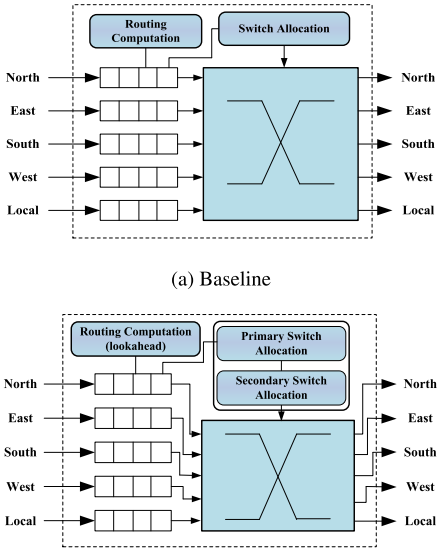
There also exist some methods which improve packets transmitting by avoiding additional input buffers. DXbar (dual-crossbar) combined the advantages of buffered and bufferless networks [8]. This design can improve performance by utilizing two crossbars in which one implements with no input buffers for no conflict switching, and another with buffers for conflicts switching. This method could take the advantage of power-efficient and low-latency bufferless networks for low traffic load. However, it has two important shortcomings. One is the area overhead of utilizing two crossbars. Second, at high traffic load, conflicting packets still will be buffered and have to wait no other packets from bufferless, thus the performance of latency is degraded by packets blocking. In Ref. [9], it decoupled the crossbar into two sub-crossbars for two dimensions. This design utilized smaller crossbars to reduce crossbar power and enable fault tolerance, but the wiring in front of two crossbar is too complex and power consumption will increase. For adaptive router, separable allocators can get good matching and avoid blocking by iteration [1], [21]. However, since alternative paths are available, there exist conflicts in each input port. Thus, more time and hardware overhead are required to deal with these conflicts. In order to resolve these drawbacks, we propose a dual-switch allocation (DSA) design. It can make utmost use of idle output ports to enhance the performance of latency and throughput. At the same time, power and router area overheads are acceptably small.

3. Design of DSA Router

3.1 Architecture

In this Section, we will introduce the architecture of DSA. Our design can be applied to both conventional and virtual channel router. In order to simplify the discussion, the virtual channel is eliminated in the rest of paper. **Figure 1** (a) shows a baseline router which has five input ports and five output ports. When there are incoming flits, they will be buffered firstly, after that the desired output port is determined by routing computation unit. Sequentially, switch allocation will assign flits to the desired output. If the desired direction has been occupied by any other flit, the flit will standby in buffer until the desired direction is available. Eventually, crossbar is controlled by the switch allocation for correctly connecting input ports to output ports.

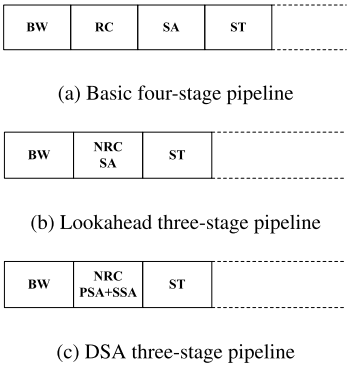
Figure 1 (b) demonstrates the router architecture of the proposed DSA. In logical conception, switch allocation is divided into PSA and SSA. Actually, we reuse the same switch allocation hardware for both in order to save router area overhead. In other words, PSA is firstly executed, after that switching algorithm is re-executed for SSA. In order to guarantee the fairness of assigning, each allocation is based on round-robin method [1]. Look-ahead technique is utilized in our method, which calculates the desired route direction for the downstream router, not for the current router [1], [10]. At first, all flits are buffered in input buffers. After that flits in each buffer will be assigned to their desired output ports by the PSA. If some of them fail in allocation in PSA, they will continuously utilize the SSA to assign its direction according to the lookahead information which is calculated in the current router.



(a) Baseline

(b) DSA

Fig. 1 The architecture of (a) Baseline (b) DSA router.



(a) Basic four-stage pipeline

(b) Lookahead three-stage pipeline

(c) DSA three-stage pipeline

Fig. 2 Comparison of three pipelines architectures. These stages are buffer writing (BW), routing computation (RC), next hop RC (NRC), switch allocation (SA), primary SA (PSA), secondary SA (SSA), and switch traversal (ST).

3.2 Router Implementation

Figure 2 demonstrates pipeline stages in different router designs. Figure 2 (a) shows the router pipeline of baseline design whose stages are buffer writing (BW), routing computation (RC), switch allocation (SA) and switch traversal (ST) [1]. When there are incoming flits, they are stored in input buffers at BW stage, after that routing computation is executed at RC stage according to head flits. Then SA will assign the desired output. Eventually crossbar connects input ports to output ports according to SA result at ST stage. In regard to a lookahead (LA) pipeline shown in Fig. 2 (b), RC is done at the preceding router, and flits can make SA stage once they are buffered in input ports. Thus no RC is independently needed for transmitting the packet to a router neighboring the current router. In this case, router performance can be improved and power consumption can be reduced, as reducing pipeline stages from four to three. Figure 2 (c) shows the pipeline

© 2015 Information Processing Society of Japan

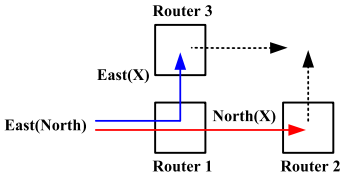


Fig. 3 The operation of DSA. The outside of the bracket is OLD-RI and the inside of the bracket is NEW-RI. The red arrow indicates the PSA and blue arrow indicate the SSA.

of the proposed DSA architecture. By utilizing a lookahead technique, all incoming head flits buffered in input ports will make switch allocation by PSA immediately. If some of these flits fail in PSA, they can complete switch allocation by SSA according to routing information which is calculated by RC at the current router. By utilizing DSA, packets contention delay will be reduced efficiently, at the same time, without increasing the signal router delay (details are described in Section 3.4).

Figure 3 shows the operation of DSA. Some definitions are given as follows:

Definition 1. *old routing information* (*OLD-RI*) *which is calculated in the preceding router and carried by incoming head flits.*

Definition 2. *new routing information* (*NEW-RI*) *is calculated by the current router and decide which output port is desired in the downstream router.*

In Fig. 3, the desired east direction is OLD-RI and north shown in bracket is NEW-RI which is calculated by Router 1. If the desired output port in east direction is occupied by other packet and Router 1 knows this packet should be sent to north direction at the downstream router (Router 2) according to NEW-RI, thus in order to resolve the blocking of this packet, this packet will be firstly transmitted to north direction at Router 1, after that transmitted to east direction at Router 3. If the packet is assigned to its desired direction by PSA, it is transmitted with NEW-RI (north) when leaving the current router. While the packet completes switch assignment by SSA, it is transmitted with OLD-RI (east) when leaving the current router. As a result, the performance of network is improved, because of making utmost of idle output port and making more packets being transmitted and buffered.

Figure 4 shows the flow chart of DSA. At first, the head flits which carry OLD-RI are buffered in each input buffers. After that PSA assigns each flit to its desired output port according to OLD-RI. During the assignment of PSA, routing computation unit calculates the NEW-RI. If switch allocation is completed in PSA, crossbar connects input ports to output ports according to PSA result and then send flits to the downstream router. In this case, when packet leaves the current router, it will carry NEW-RI information together.

At low traffic load, each head flit may enable be assigned to their desired output port, as there is no contention between flits. However, as traffic load increasing, the preceding packet might block the succeeding packet which desires the same output port. Therefore, some of head flits in input buffer will fail assignment in PSA. In this case, router re-execute switching algorithm by utilizing same hardware for SSA. These failed head flits will continuously make switch allocation by SSA according to the value of NEW-RI. If the NEW-RI direction is available for one head

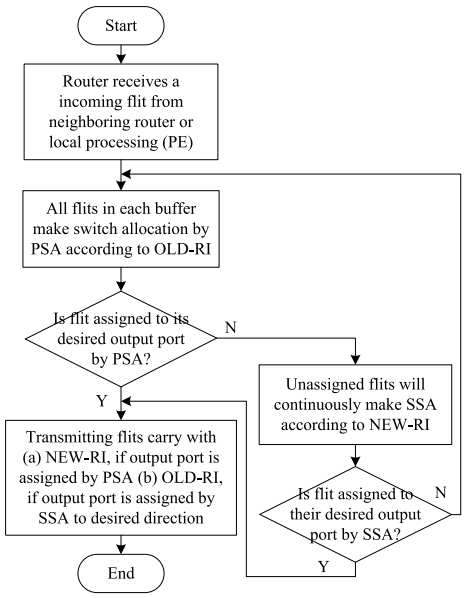


Fig. 4 Flow chart of the proposed DSA.

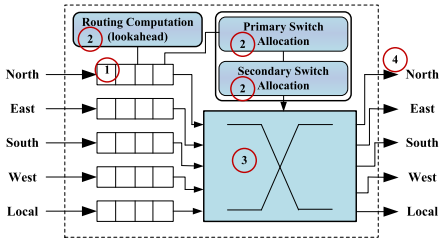


Fig. 5 The timing of a flit.

flit, SSA assigns this packet to this direction. In other words, although the desired direction of the current router is unavailable, the router knows which direction the packet should be transmitted to in the downstream router. Thus the packet can be firstly transmitted to the direction which is desired in the downstream router if it is available, after that sent to the direction which is desired in the current router. Similarly, if switch allocation is completed in SSA, crossbar will connect output port to input port and transmit packet to the downstream router, carrying OLD-RI. Otherwise, if SSA fails, head flits will continue to standby in input buffer for PSA, and repeat this process.

A round-robin technique is utilized in PSA and SSA to guarantee the fairness. The idle output ports can be made utmost use to improve the network performance in DSA, as reducing the contention between flits. Meanwhile, DSA not increase a signal router delay. On the other hand, since SA merely consumes a few power, power overhead of DSA is very small (details are described in Section 4).

The timing of an incoming flit is shown in **Fig. 5**. In cycle 1, the incoming flit arrives at input port and is buffered in input buffer. In cycle 2, PSA make switch allocation according to OLD-RI. Subsequently, if the incoming flit fails in PSA, SSA will make switch allocation according to NEW-RI which is calculated by routing computation unit at the same time. After that crossbar connects input port to output port and transmit flit to the downstream router according to SA results in cycle 3 and 4, respectively. RC and DSA stages (consist of PSA and SSA) are executed in parallel.

© 2015 Information Processing Society of Japan

3.3 Adaptive Router Analysis

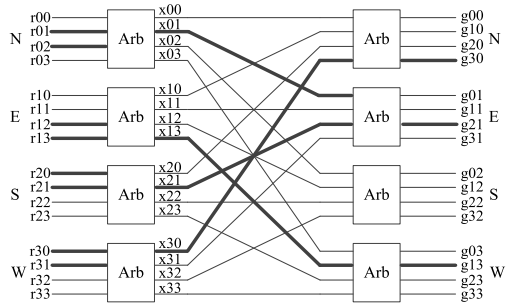
Our proposed DSA can reduce switch allocation time and hardware resource compared with conventional adaptive routers. In this section, we discuss these advantage. Both the proposed DSA and an adaptive router are used to get a good matching between input ports and output ports. For adaptive routers, the simplest and efficient allocator is separable one [1], [12]. There are two types of separable allocators: input-first and output-first. Since the principle of those two types allocators is similar, we explain the difference between the proposed DSA and conventional adaptive routers by using input-first separable allocators.

The advantage of proposed DSA against adaptive routers can be qualitatively explained by using an example as follows. **Figure 6** shows input-first separable allocators for an adaptive router, where 0, 1, 2, 3 mean north, east, south and west, respectively. For example, r01 means a flit in north input port to request east output port.

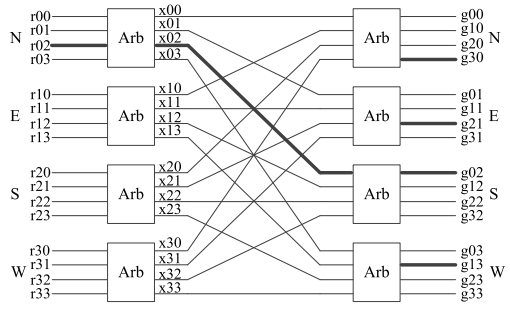
In an adaptive router, since alternative paths can be available between source and destination as shown in Fig. 6 (a), a flit in north input port requests east and south output ports, and a flit in east input port requests south and west output port. Similarly, a flit in south input requests north and east output ports, and a flit in west input requests north and east output ports. We can define a request matrix R_1 to indicate these requests,

$$R_1 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

where the rows and columns indicate input ports and output ports, respectively. For example, the first row indicates that the flit in north input port requests east and south output ports as shown in Fig. 6 (a). On the other hand, the first column indicates which input ports request north output port. An input-first separable allocators takes a request matrix and performs arbitration across the rows first and then down the columns.



(a) First allocation



(b) Iteration for allocation

Fig. 6 Separable allocators for adaptive router.

For an adaptive router, the first stage's arbitration selects the winning request for each input ports in parallel. Thus, an intermediate request matrix X_1 after the input arbiters can be got, where input conflicts are eliminated and there exist at most one non-zero entry in each row. After that the second stage's arbitration will eliminate output conflicts, giving a final grant matrix G_1 with at most one non-zero in each column.

$$X_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, G_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

In G_1 there is no flit assignment to north input port and south output port, meanwhile, north input port requests south output port in R_1. Therefore, this is not a good matching. In order to get a better matching, adaptive router will iterate allocation for losing input port as shown in Fig. 6 (b). Thus, a new request and grant matrixes are got as follows.

$$R_2 = X_2 = G_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Finally, a better matching can be got by cumulating grant matrix G_1 and G_2. Note that the first allocation which gets G_1 and the second allocation which gets G_2 are performed sequentially.

$$G = G_1 + G_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Here, let's assume the time of first and second stage's arbitrations of separable allocators are T_1 and T_2, respectively. Thus, we can get the time consuming of separable allocators.

$$\begin{cases} T_{SA} = T_{R_1 \to X_1} + T_{X_1 \to G_1} = T_1 + T_2 \\ T'_{SA} = T_{R_2 \to X_2} + T_{X_2 \to G_2} = T_1 + T_2 \end{cases} \quad (1)$$

According to Eq. (1), the time of getting a better matching G in adaptive router is $T_G = T_{SA} + T'_{SA} = 2T_1 + 2T_2$.

Compared with separable allocators, let's consider the same example in our proposed DSA as shown in **Fig. 7**. The letter inside a bracket is lookahead routing information. For example, r01 in Fig. 7 is a flit in north input port which requests east output port, and this flit desires south output (S) in a downstream router. Since our proposed DSA uses a lookahead technique, there exist no conflict in each input port and a request matrix R'_1 is got. Thus, only one stage's arbitration is enough in our method, and we can save the first stage's arbitration. Then a grant matrix G'_1 is got by using PSA as shown in Fig. 7 (a). In PSA, all requests are performed in parallel like separable allocators.

© 2015 Information Processing Society of Japan

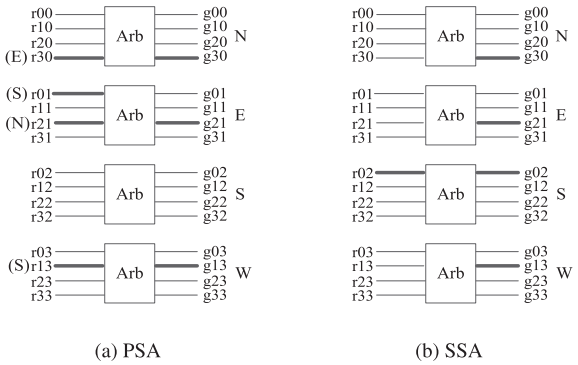


Fig. 7 Allocator of proposed DSA.

$$R'_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, G'_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

As shown in G'_1, north input port is lost in PSA, however, the router knows it's lookahead information that the flit in north input port will be sent to south direction in downstream router. Thus, in order to get better matching, switch allocation will be iterated (in SSA) as shown in Fig. 7 (b), and its request matrix R'_2 and grant matrix G'_2 are as follows.

$$R'_2 = G'_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Finally, we can get a better matching result by cumulating G'_1 and G'_2.

$$G' = G'_1 + G'_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Therefore, the time consuming of proposed DSA is

$$\begin{cases} T_{PSA} = T_{R'_1 \to G'_1} = T_2 \\ T_{SSA} = T_{R'_2 \to G'_2} = T_2 \end{cases} \quad (2)$$

According to Eq. (2), getting the same matching, proposed DSA consumes $T'_G = T_{SSA} + T_{PSA} = 2T_2$ ($< T_G$). Consequently, the proposed DSA can save switch allocation time. Besides, our proposed DSA also can save hardware overhead and energy because of fewer arbitrations.

3.4 Delay Analysis

In this section, we discuss the delay of DSA. Modern router consists of different components, and for different components i, there exist two delay estimates: *latency* (t_i) and *overhead* (h_i). Latency spans from when inputs are presented to the components, to when outputs needed by the next components are stable. On the other hand, the overhead is the time of setup delay required before the next set of inputs can be presented to components. Therefore we can get

Table 1 The delay (in units of τ) of pipeline stage in different designs.

Router[1]	SA		ST	
	t_{SA}	h_{SA}	t_{ST}	h_{ST}
Baseline (four-stage)	39.04	9	100	0
Lookahead (three-stage)	39.04	9	100	0
DSA (three-stage)	78.08[2]	9	100	0

[1] five input/output ports for each design
[2] the t_{SA} of DSA consists of t_{PSA} and t_{SSA}

$$T = \sum_{i=a}^{b} t_i + h_b \quad (3)$$

where T is the stage delay and a, b are the first and last components in the pipeline stage, respectively. The stage with longest critical path delay will set the clock frequency [11].

We model the delay of switch allocation (SA) component of wormhole router by the technology-independent parametric equation [12].

$$\begin{cases} t_{SA}(p) = 21\frac{1}{2}\log_4 p + 14\frac{1}{12} \\ h_{SA}(p) = 9 \end{cases} \quad (4)$$

where p is the number of input/output ports and the result is in unit of τ which is the delay of an inverter with identical input capacitance [13]. Note that we implement dimensional-order-routing which is a very simple deterministic routing algorithm in RC stage of our method. Therefore, the delay of RC stage is less than SA stage. However, for ST stage, since there exists wire delay in crossbar which is not considered in Ref. [12], it significantly affects the component delay. In order to alleviate the impact, we take into account the wire delay, and the delay in the crossbar pipeline stage is assumed 100τ ($= 20\tau_4$ in Ref. [12]). As mentioned in Ref. [12], when crossbar size is small, this assumption is reasonable, including wire delay. Note that the delay of SA and ST stage are calculated in units of τ, that is not τ_4 in Ref. [12].

Table 1 shows the delays derived from Eqs. (3) and (4) for SA and ST pipeline stage achieved by baseline, LA and DSA. The number of input/output ports for each design are set five. From Table 1, obviously, the delay of ST stage is the longest critical path. The switch allocation delay increases in DSA compared with baseline and LA design, as DSA operates twice SA, one for PSA and another for SSA. However, any clock cycle that accommodates the ST stage will also accommodate the increased SA delay in DSA design. As a result, the penalty of additional SA in DSA will hidden in router pipeline.

In regard to network latency, packets contention delay is a significant delay component in network. By utilizing our method, packets contention delay can be reduced efficiently. Let's assume that there are two four-flit packets arriving at different input ports and both desire the same output port. There exist two specific situations. **Figure 8** shows the best situation for packets contention delay saving, that is the head flits of packet A and B are arriving at the current router at the same time. For normal router, if the desired output port is occupied by packet A, packet B have to wait in input buffer until all flits of packet A leaving the desired output, after that packet B can utilize this output port. In Fig. 8, the tail flit of packet A leaves the desired output at cycle 6, thus router

© 2015 Information Processing Society of Japan

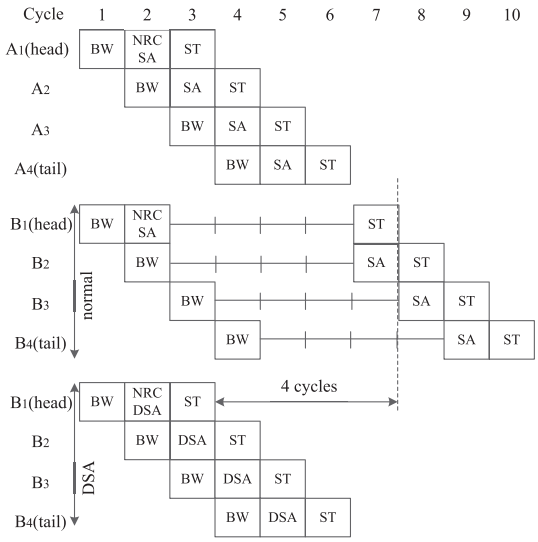


Fig. 8 The best situation that A_1 and B_1 arrive at the same time.

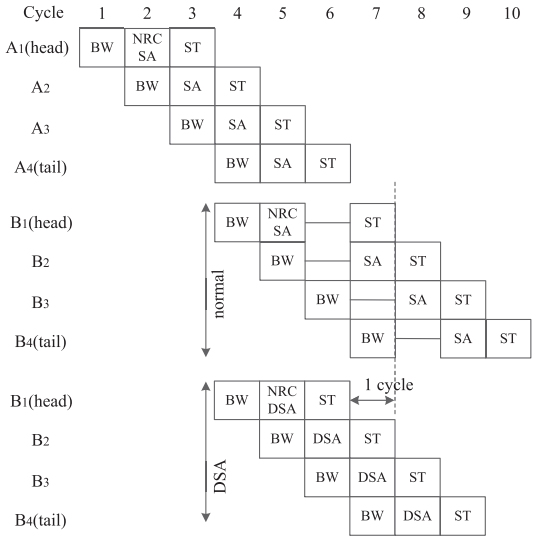


Fig. 9 The worst situation that A_4 and B_1 arrive at the same time.

enable packet B to start transmitting at cycle 7 for normal router. On the other hand, for DSA router, although the desired output port is occupied by packet A, DSA enable a packet to start transmitting according to NEW-RI at cycle 3 without waiting packet A. Thus four cycles can be reduced by DSA in this case.

Similarly, **Fig. 9** shows the worst situation that the tail flit of packet A and the head flit of packet B are arriving at the current router at the same time. In this situation, normal router have to wait one cycle for tail flit of packet A leaving and router releases the desired output port. Nonetheless, if the output port of NEW-RI is available, DSA enable packet to transmit without waiting packet A as above description. Therefore, in this case, one cycle can be reduced for packets contention by our method. Actually, the packets contention delay saving is between the best and worst situations.

The average latency of network can be expressed in the following equation [1]

$$T_{average} = H_{\min} t_{router} + \frac{D_{\min}}{v} + \frac{L}{b} + t_{contention} \quad (5)$$

where $H_{\min}$ is the average hop counts, t_{router} is the delay of a signal router, $\frac{D_{\min}}{v}$ is the time spent on the wires and $\frac{L}{b}$ is serialization latency. The last term is the time of packets contentions. Since our method can reduce the delay of packets contention without increasing a signal router delay, according to Eq. (5), a latency improvement can be achieved by DSA. In our method, we utilize the idle output ports of router to transmit packets as far as possible, thus throughput can also be improved with a few power overhead. Therefore, our proposed method is efficient.

3.5 Examples

There are some different allocation situations in DSA. How flits are assigned for each situation is shown in **Fig. 10**. In DSA, since minimal routing algorithm [1] is deployed, router doesn't consider sending packets to back, in other words, each router will make packets close to their destinations and no detour exist between the current and the destination nodes. If the routing direction of packet is local, this packet is always assigned by PSA. It means even one packet which desires the local output port fails in PSA, SSA will not be performed, and this packet will wait for next PSA. In Fig. 10, the letter out of bracket is OLD-RI which is carried with incoming head flits, while the letter in bracket is NEW-RI which is calculated by the current router. For example, E(S) means that E is OLD-RI and S is NEW-RI.

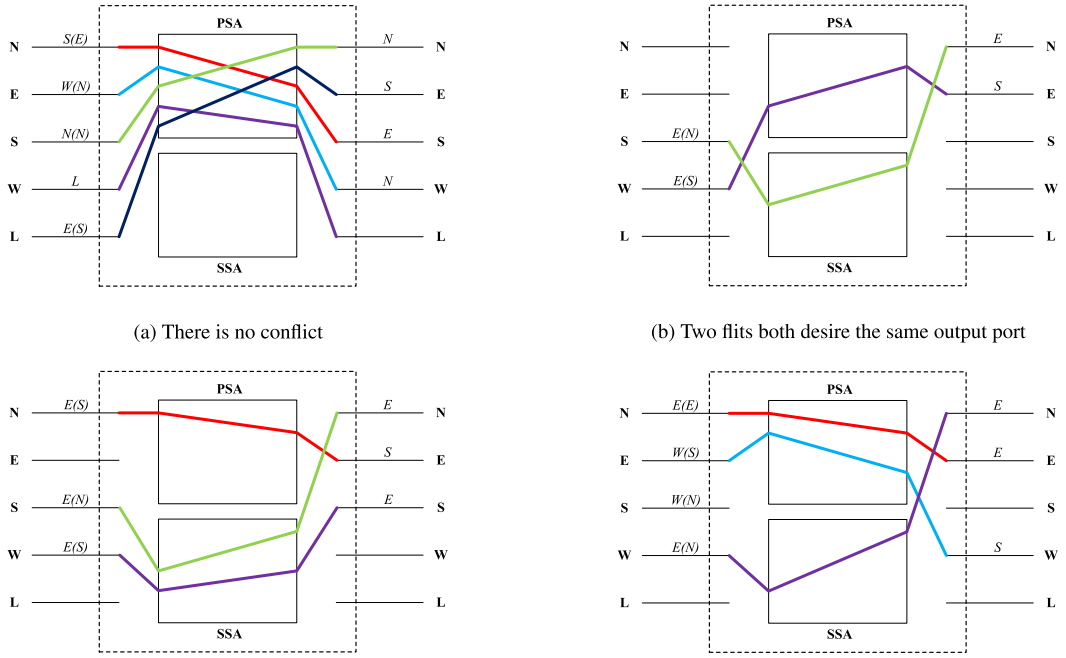
Figure 10 (a) shows the situation with no conflict. All flits can be assigned to their desired output ports by PSA. Thus when flits leave the current router, NEW-RI will be carried with them. For instance, the flit in north input port has been assigned to south output port (S). When it leaves the current router, NEW-RI (E) will be carried with it. For east input port, the flit is assigned to west output port and it will carry with NEW-RI (N) when leaving the current router. Note that if the packet desires local output port, it always completes switch allocation in PSA. In this case, SSA is not utilized.

Figure 10 (b) shows the situation when two flits desire the same direction. The flits in south and west input ports want to be sent to east direction. Firstly, all flits in each buffer will make switch allocation by PSA. Based on round-robin technique, if east output port is assigned to west input port (thus it is unavailable for flit in south input port), the flit in south input port will fail in PSA and continuously utilize SSA according to NEW-RI (N). In this case, the NEW-RI (N) of flit in south input port is available. Thus the router firstly assigns this flit to north output port by SSA and when the flit leaving the current router, it will carry with OLD-RI (E). After that in the downstream router, it will desire east direction.

Figure 10 (c) shows the situation more than two flits desire the same direction. The OLD-RI of flits in north, south and west input ports are east direction and east output port is assigned to the flit in north input port by PSA. These failed flits will utilize SSA to complete assignment. In this case, the NEW-RI directions of flits in south and west input ports (N and S) are available, thus the router will transmit flit in south input port to north direction, and send flit in west input port to south direction. After that in the downstream router, they will desire east output port.

Figure 10 (d) shows the situation that some of flits fail in PSA and SSA. Flits in north and west, east and south input ports desire

© 2015 Information Processing Society of Japan



(a) There is no conflict

(b) Two flits both desire the same output port

(c) More than two flits desire the same output port

(d) Flit both fails in PSA and SSA

Fig. 10 The examples of DSA method.

east and west directions, respectively. North input port is assigned to east output port and east input port is assigned to west output port. Thus flits in south and west input ports fail in PSA. Within SSA, the NEW-RI of these two flits both are north direction. In this case, based on round-robin, if north output port is assigned to west input port, the flit in south input port also fails in SSA and it will wait in input buffer for next PSA.

3.6 Deadlock and Fault Tolerance

In order to avoid deadlock, a deadlock recovery scheme DISHA [14] is utilized in our method. To break deadlock cycle, an extra one flit buffer which is called deadlock flit buffer is equipped at each router to store the head flit of one of deadlock packet. T is set to keep track of the number of clock cycles. The value of T is increased as head flit cannot be sent out. Whenever T is larger than the threshold T_{th}, recovery will be executed. In this case, the deadlock flit will be sent to deadlock flit buffer, and in every downstream routers, these deadlock flits only utilize the deadlock flit buffer until arriving at the destination node. To implement this scheme, a one flit buffer will be appended into each router and crossbar will increase one input port. However, even there exist additional hardware, the hardware overhead is only few (details in Section 4.2).

The DSA design enables hardware-level fault tolerance [15], [16]. If the link between two routers is fault, PSA will fail according to OLD-RI. However, router enables packet to transmit by SSA according to NEW-RI and the packet can forward to its destination avoiding the fault. For instance, **Fig. 11** shows a one-faulty situation in DSA. Node 1 and node 8 are the source and the destination nodes, respectively. In normal case, we assume packets will pass through node 2 and 5 according to OLD-RI. However, if the east link of node 1 is fault, it means packets will fail in PSA. In this case, packets will avoid the faulty link by utilizing SSA according to NEW-RI. Thus packets is able to transmit through node 4 and 5 to approach the destination node.

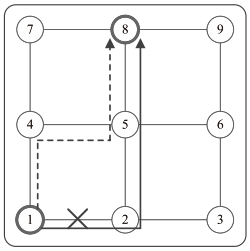


Fig. 11 A simple one-faulty situation in DSA, node 1 and node 8 are the source and the destination. The solid line is normal path using PSA and dash line is fault tolerance path using SSA.

Our proposed method can definitely work on other virtual channel designs [1], [17]. Similarly, if all virtual channels in the desired input port of the downstream router are not available, the blocked packet is able to utilize SSA to forward to the direction which is desired in the next router according to NEW-RI information.

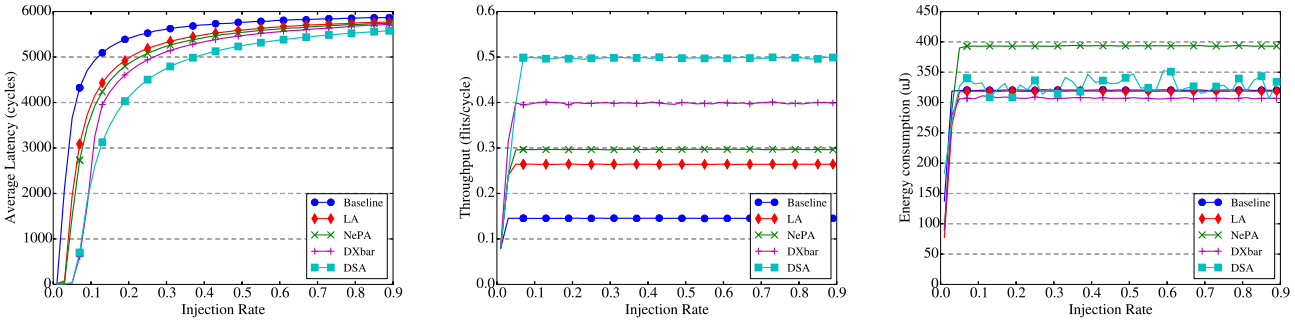
4. Performance Evaluation

In this section, we perform different experiments in order to evaluate the DSA design in terms of area, power consumption and the network performance. We compare our design with baseline router, lookahead router, NePA [4] which utilizes lookahead method and DXbar [8]. Dimensional-order-routing (DOR) [1] is deployed in baseline, lookahead, NePA and DXbar, respectively. At first, we implement all designs in a small 4×4 2D mesh, and then perform the simulation in an 8×8 mesh topology.

4.1 The Network Performance at Synthetic Traffics

We evaluate the performance of network by using an open source simulator Noxim [18], which is developed by Systemc and

© 2015 Information Processing Society of Japan



(a) The latency at random traffic (b) The throughput at random traffic (c) The energy consumption at random traffic

Fig. 14 The performance of different designs at random traffic in 8×8 mesh.

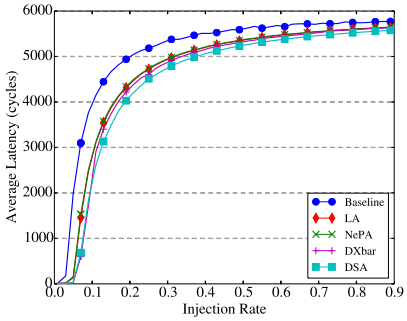


Fig. 12 The average latency at random traffic in 4×4 mesh.

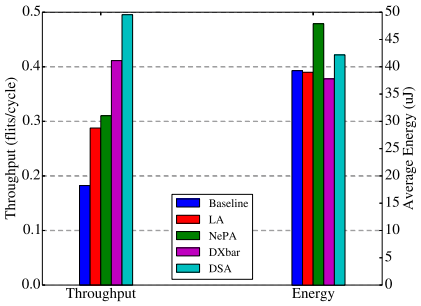


Fig. 13 The throughput and energy consumption at injection rate = 0.5 under random traffic in 4×4 mesh.

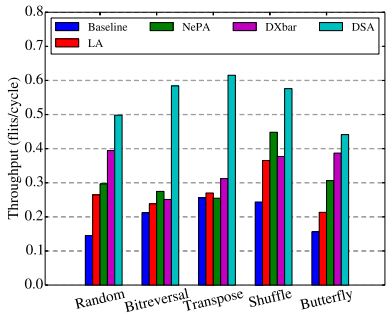
supports cycle-accurate simulation. At first, consecutive simulations are executed in 4×4 and 8×8 2D mesh, where the injection rate is varied from 0.01 to 0.90 using random traffic pattern. To guarantee the fairness of experiments, the total buffer sizes are same for different designs. Each buffer size of NePA and DSA are set three flits and four flits, respectively. Each input buffer size of baseline and LA are set four flits except local input buffer and the local buffer size of them is five flits. One input buffer size of DXbar is set five flits and remaining input buffers are four flits. Therefore total buffer sizes are same for different designs. In this experiment, we compare the performance of DSA with other designs in terms of latency, throughput and energy consumption. In order to achieve stable network performance, 10,000 cycles are taken to record simulation and warmup time is set 2,000 cycles.

Figure 12 shows the compared latency performance in 4×4 mesh. As the injection rate increasing, DSA achieves better performance than other designs. In this case, DSA reduce the average latency 19.9% compared with baseline design and almost 8.4% compared with LA and NePA designs. Compared with DXbar design, DSA achieves 6.7% latency reducing. **Figure 13** demonstrates the throughput and network energy consumption at injection rate 0.5 under random traffic. For throughput, DSA also gains the best performance than other designs. On the other hand, in regard to network energy, DSA consumes more power than baseline, LA and DXbar, but still less than NePA design. From Fig. 13, even there exist more energy consumption in DSA compared with baseline, LA and DXbar, the power overhead is few.

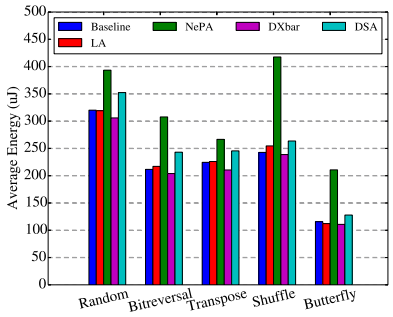
Figure 14 (a) shows the performance of latency between different designs in an 8×8 mesh. Obviously, DSA is also significantly better than others. As traffic load increasing, DSA gets average improvement by 38.8%, 29.6%, 26.5% and 21.3% compared with baseline, LA, NePA and DXbar designs, respectively. From Figs. 12 and 14 (a), at very low injection rate, the performance of DXbar is almost same with DSA design, actually, DXbar has a bit better performance than DSA, as a few packets contention exist at low injection rate and most packets are transmitted by bufferless. However, at high traffic load, packets contention become frequently and conflicting packets still will be buffered, thus packets blocking gravely upset performance. On the other hand, as injection rate increasing, the latency performance improvements of DSA reduce. That is because when injection rate is relative large, the whole network will become very congested, in other words, the number of idle output ports in router is almost none. Thus a few packets can perform SSA to break blocking. At high traffic load, although the latency performance improvements reduce, our proposed method still better than other designs.

Figure 14 (b) shows the throughput of each design in an 8×8 mesh. After saturation point, the throughput of DSA reaches almost 0.5 and outperforms other designs. Figure 14 (c) shows the energy consumption for the whole network. Since NePA design adds two additional input ports including buffers, more buffer operations such as writing and reading will be executed in NePA and then dynamic and leakage power of buffer will increase linearly [19]. For DXbar design, on the one hand, it take the advantage of power-efficient bufferless network, on the other hand, the conflicting packets also will be buffered when packets blocking happen, thus the energy consumption of DXbar is few lesser than baseline and LA designs. From Fig. 14 (c), at low traffic load, DSA consumes almost same energy with baseline and LA designs, as PSA is utilized more frequently. However, as traffic load increasing, more packets blocking happens. In this situation, both PSA and SSA will be utilized, thus the power consumption of DSA becomes a bit greater than baseline and LA designs.

© 2015 Information Processing Society of Japan



(a) Throughput at injection rate = 0.5



(b) Energy consumption at injection rate = 0.5

Fig. 15 The throughput and energy consumption at injection rate = 0.5 of other synthetic traffics in 8×8 mesh.

Although there is more energy consumption in DSA, the average overhead is only 3.0% and 5.9% compared with baseline and DXbar designs, respectively.

Other traffic patterns are also evaluated in each design in the 8×8 mesh, such as bit-reversal, transpose, shuffle and butterfly. **Figure 15** (a) shows the resultant throughput at the injection rate 0.5. Obviously, the throughput of DSA is significantly better than others. Especially, at bit-reversal and transpose traffic, the improvement of DSA is distinct and over two times compared with other designs. Figure 15 (b) shows the energy consumption under different traffics at 0.5 injection rate. From Figs. 13 and 15 (b), there exist a few power overhead in DSA design, as two switch allocation will be operated at high traffic load. However, our method save buffer energy compared with NePA.

4.2 Area and Power Estimation

We use Orion 2.0 simulator [20] to estimate the area and power of router, with the setting of 65 nm technology, 1 GHz router at 1 Vdd, and the flits size is 128 bits. **Table 2** shows the evaluation of area and power for different designs. In regard to area, obviously, the design of DXbar occupies the most area compared with others. This is due to the fact that the crossbar occupies the largest area in a router. It has over two times area than DSA. For the design of NePA, it also has very large area, as it has much more input buffers and a crossbar has more complexity. It almost has two times area compared with DSA. On the other hand, the area of our design has only a bit larger compared with baseline and LA designs, as we only add a one flit buffer and one more input for crossbar and reuse same hardware for two switch allocation operation. In regard to router power, the power of NePA is the largest one among different designs, because the power is rapidly increased with the number of input buffers. The power of buffer dominates the power consumption of whole router. Since the DXbar design takes the advantage of power-efficient bufferless network, it has lesser power than others. For DSA design, there exist two specific cases. One is that the flit passes through a router only assigned by PSA, it means SSA is not utilized. In this case, the power consumption is minimal that is almost same with LA and all switch energy is contributed by PSA. Second is that the flit firstly fails in PSA, and then it is forwarded to the downstream router by SSA. In this case, both PSA and SSA are utilized, thus it gains the maximum power consumption. From Table 2, even both max power and average power of DSA are greater than that of baseline, LA and DXbar designs, it is very little.

Table 2 Area and power estimation for different designs.

Designs	Area (mm^2)	Power (pJ/flit)
Baseline	0.3084	245.18
Lookahead (LA)	0.3084	245.07
NePA	0.6016	460.05
DXbar	0.6483	241.60
DSA	0.3094	245.09[1]/247.54[2]/246.31[3]

[1] the minimum energy, as only PSA is utilized
[2] the maximum energy, as both PSA and SSA are utilized
[3] the average energy consumption

5. Conclusions

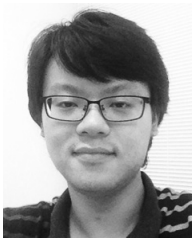
More and more cores will be deployed in NoC. Packets may be blocked more frequently as the traffic load increasing. In order to reduce blocking, additional buffers are used to improve latency and throughput. However, the power consumption will be increased obviously. To improve the network performance under the less overhead power and router area, this paper proposes a dual-switch allocation network (DSA). It allows packets assign their desired output port by the primary switch allocation firstly. If some of packets fail in the primary switch allocation, they can continuously utilize the secondary switch allocation to assign their desired directions. The router can make utmost use of idle output port, as much as possible, to improve the network performance. Experimental results show that our design has significant performance improvement in terms of latency and throughput, at the cost of a small overhead power.

References

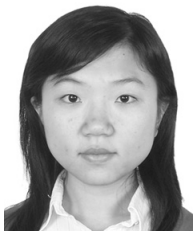
[1] Dally, W.J. and Towles, B.: *Principle and practices of interconnection network*, Morgan Kaufmann Inc., San Francisco, CA (2004).
[2] Kodi, A.K., Sarathy, A. and Louri, A.: IDEAL: Inter-router dual-function energy and area-efficient links for network-on-chip (NoC) architecture, *Proc. 35th International Symposium on Computer Architecture* (*ISCA '08*), pp.241–250 (June 2008).
[3] Suseela, J. and Muthukumar, V.: Loopback virtual channel router architecture for network on chip, *Proc. 9th International Conference on Information Technology: New Generations* (*ITNG 2012*), pp.534–539 (Apr. 2012).
[4] Bahn, J.H.. Lee, S.E., Yang, Y.S., Yang, J. and Bagherzadeh, N.: On design and application mapping of a network-on-chip (NoC) architecture, *Parallel Processing Letters*, Vol.18, pp.239–255 (2008).
[5] Roca, A., Flich, J., Silla, F. and Duato, J.: A latency-efficient router architecture for CMP systems, *Proc. 13th Euromicro Conference on Digital System Design: Architecture, Methods and Tools* (*DSD 2010*), pp.165–172 (Sep. 2010).
[6] Carara, E., Calazans, N. and Moraes, F.: A new router architecture for high-performance intrachip network, *Journal of Integrated Circuits and Systems*, Vol.3, pp.23–31 (2008).
[7] Mullins, R., West, A. and Moore, S.: The design and implementa-

© 2015 Information Processing Society of Japan

tion of a low-latency on-chip network, *Proc. 11th Asia and South Pacific Conference on Design Automation* (*ASP-DAC 2006*), pp.164–169 (Jan. 2006).
[8] Zhang, Y., Morris Jr., R. and Kodi, A.K.: Design of a performance enhanced and power reduced dual-crossbar network-on-chip (NoC) architecture, *Microprocessors and Microsystems*, Vol.35, pp.110–118 (2011).
[9] Kim, J., Nicopoulos, C., Park, D., Narayanan V., Yousif, M.S., and Das C.R.: A gracefully degrading and energy-efficient modular router architecture for on-chip networks, *Proc. 33rd International Symposium on Computer Architecture* (*ISCA 2006*), pp.4–15 (2006).
[10] Ahmed, A.B. and Abdallah, A.B.: LA-XYZ: Low latency, high throughput look-ahead routing algorithm for 3D network-on-chip (3D-NoC) architecture, *Proc. IEEE 6th International Symposium on Embedded Multicore SoCs* (*MCSoC 2012*), pp.167–174 (Sep. 2012).
[11] Jerger, N.E. and Peh, L.-S.: *On-chip networks*, Morgan & Claypool (2009).
[12] Peh, L.-S. and Dally, W.J.: A delay model and speculative architecture for pipelined routers, *Proc. 7th International Symposium on High-Performance Computer Architecture* (*HPCA*), pp.255–266 (Jan. 2001).
[13] Dally, W.J. and Poulton, J.W.: *Digital systems engineering*, Cambridge University Press (1998).
[14] Anjan, K.V. and Pinkston, T.: An efficient fully adaptive deadlock recovery scheme: DISHA, *Proc. 22nd Annual International Symposium on Computer Architecture* (*ISCA*), pp.201–210 (June 1995).
[15] Fangfa, F., Jianxin, M., Zixu, W. and Jinxiang, W.: Low-cost router microarchitecture based on dimension-switch for fault-tolerance in 2D-mesh NoC, *Proc. 2011 Academic International Symposium on Optoelectronics and Microelectronics Technology* (*AISOMT 2011*), pp.313–319 (Oct. 2011).
[16] Lehtonen, T., Liljeberg, P. and Plosila, J.: Fault tolerance analysis of NoC architecture, *Proc. 2007 IEEE International Symposium on Circuits and Systems* (*ISCAS 2007*), pp.361–364 (May 2007).
[17] Ning, H., Duoli, Z., Gaoming, D. and Yukun, S.: Design and performance evaluation of virtual-channel based NoC, *Proc. 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication* (*ASID 2009*), pp.294–298 (Aug. 2009).
[18] Noxim, available from ⟨http://www.noxim.org⟩.
[19] Kahng, A.B., Bin, L. Peh. L.-S. and Samadi, K.: ORION 2.0: A power-area simulator for interconnection networks, *Proc. Very Large Scale Integration* (*VLSI*) *Systems*, Vol.20, pp.191–196 (2012).
[20] Kahng, A.B., Bin, L., Peh. L.-S. and K. Samadi: ORION 2.0: A fast and accurate NoC power and area model for early-stage design space exploration, *Proc. 2009 Design, Automation & Test in Europe Conference & Exhibition* (*DATE '09*), pp.423–428 (Apr. 2009).
[21] McKeown, N.: The iSLIP scheduling algorithm for input-queued switches, *Proc. IEEE/ACM Trans. Networking*, Vol.7, No.2, pp.188–201 (Apr. 1999).



Lian Zeng was born in MianYang, China on July, 1990. He received his B.E. degree in Software Engineering in 2011, from Sichuan University. In 2013, he received his M.S. degree and be currently working toward a Ph.D. degree in Graduate School of Information, Productions and Systems, from Waseda University. His research interests include NoC routing and router optimization. He is a member of IEICE.



Xin Jiang was born in LiaoNing, China 1981. In 2006, she received her M.S. degree in Electronic Engineering in Shanghai Marine University. And she received her Ph.D. degree in Graduate School of IPS, from Waseda University. Her current research interests include optimization algorithms in Electronics DA designs. She is a member of IEICE.

Takahiro Watanabe was born in Ube, Japan in October, 1950. He received his B.E. and M.E. in Electrical Engineering, Yamaguchi University, and Dr. Eng. from Tohoku University. In 1979, he joined Research and Development Center of TOSHIBA Corporation, where he worked in the field of LSI design automation. In August 1990, he joined Yamaguchi University, the Department of Computer Science and Systems Engineering, and in April 2003, he moved to Waseda University, Graduate School of Information, Production and Systems. His current research interests are EDA algorithm, Microprocessor and MPSoC, NoC, FPGA and their applications. He is a member of IEICE, IPSJ and IEEE.

(Recommended by Associate Editor: *Tetsuo Hironaka*)

© 2015 Information Processing Society of Japan