IPSJ Transactions on Advanced Computing Systems Vol. 4 No. 2 84–93 (Mar. 2011)

Regular Paper

A High-throughput Router Architecture with On-the-fly Virtual Channel Allocation for On-chip Networks

Son Truong Nguyen[†1] and Shigeru Oyanagi[†1]

Designing high throughput and low latency on-chip networks with reasonable area overhead is becoming a major technical challenge. This paper proposes an architecture of router with on-the-fly virtual channel (VC) allocation for high performance on-chip networks. By performing the VC allocation based on the result of switch allocation, the dependency between the VC allocation and switch traversal is removed and these stages can be concurrently performed in a non-speculative fashion. In this manner, the pipeline of a packet transfer can be shortened without the penalty of area. The proposed architecture has been implemented on FPGA and evaluated in terms of network latency, throughput and area overhead. The experimental results show that, the proposed router with on-the-fly VC allocation can reduce the network latency by 40.9%, and improve throughput by 47.6% as compared to the conventional VC router. In comparison with the look-ahead speculative router, it improves the throughput by 8.8% with 16.7% reduction of area for control logic.

1. Introduction

System-on-Chip (SoC) design methodologies provide a powerful, capable and flexible solution to integrate complex systems on a single chip with the development of high-density VLSI technology. The requirement to design high-performance, energy-efficient and scalable interconnections in such the SoCs is becoming a major technical challenge[1),2)]. A high-performance scalable communication infrastructure called a Network-on-Chip (NoC), that provides better support for communication on SoCs compared to the traditional bus-based architecture is becoming more and more popular[3),4)]. Connecting components through an on-chip network has various advantages such as the low-power communication, good structure and modularity.

To meet the need for high performance interconnections for the large-scale SoCs as well as CMPs (chip multiprocessors) expected to dominate computing in the near future, reducing the communication latency with a high network throughput while maintaining a reasonable area budget becomes one of the most critical design challenges for on-chip routers[2)]. Basically, an on-chip router forwards incoming packets through a critical path of four pipeline stages including the routing computation (RC), virtual channel allocation (VA), switch allocation (SA) and switch traversal (ST). The delay of the critical path at each node is one of the causes leading to the network latency. Recently, numerous on-chip router architectures focusing on the reduction of the network latency have been proposed. The representatives include *speculation, look-ahead, bypassing, prediction* and *slicing*[5)–16)].

In the speculation model, the router speculatively performs multiple pipeline stages of a packet transfer in parallel to cut down the critical path[5)–8)]. Look-ahead routing is a mechanism that removes the serialization delay due to the routing computation by determining the route of a packet one hop in advance[6)–11)]. As a bypassing technique, express virtual channels (EVCs) are used to reduce the communication latency by bypassing some of pipeline stages at non-adjacent bypass hops[12)]. Several other router architectures allow bypassing one or more pipeline stages for the specific routes[13),14)]. On the other hand, the prediction router predicts an output channel being used by the next packet transfer and speculatively completes the switch arbitration[15)]. Another architecture is the dimension-sliced router, that partitions the crossbar into two separate crossbars, and prioritizes the packets that are in flight (continuing to travel in the same dimension) to remove the switch arbitration from the critical path[16)].

In this paper, we propose a novel high performance router architecture in which the virtual channel (VC) allocation is performed in parallel with the switch traversal in a non-speculative fashion named *on-the-fly VC allocation*. Once a packet has determined its output channel with one or more free VCs being allocated, the packet will directly proceed to the switch allocation for allocating the time slot on the crossbar switch and output channel without performing any speculation. The VC allocation is performed after the switch arbitration for winning packets. In this manner, the dependency between the VC allocation and switch traversal is removed and these stages can be non-speculatively performed in parallel. Due

†1 Department of Computer Science, Ritsumeikan University

to the fact that only packets in active are allocated VCs, this fashion of VC allocation results in a high channel utilization. As a result, there is no impact of speculation failures, and the router can attain a higher throughput than the state-of-the-art schemes which employ the speculation mechanism.

The proposed router architecture is implemented on FPGA for exploring microarchitectural characteristics and evaluating network performance. Experiments are conducted on the two-dimensional mesh network developed in Verilog. In addition, for relative comparison the routers with the conventional and look-ahead speculative architectures are also implemented and discussed in this work.

The remainder of this paper is organized as follows. Section 2 provides background on a conventional on-chip router and motivation for a high throughput router with the simplicity of hardware design. The proposed on-the-fly VC allocation architecture is described in Section 3. Implementation and experimental results are presented and discussed in Section 4, and Section 5 concludes the paper.

2. Conventional On-chip Router Architecture and Motivation

In this section, we examine the background of the architecture of a VC router that is known as a conventional on-chip router[6], and discuss the motivation for designing a high throughput architecture with reasonable area overhead and design simplicity.

2.1 Basic Design

In this work, we assume a dimension-ordered wormhole router used for the two-dimensional mesh topology as a baseline, though it can be extended for various other topologies and routing algorithms. The VC router is typically composed of the input buffers, crossbar switch and control logic as shown in **Fig. 1**, where five bi-directional ports named as North (N), South (S), East (E), West (W), and Local (L) are necessary for communication in the network. The major control logic components include a routing computation logic, a VC allocator and a switch allocator.

In this model, to begin advancing a packet the RC stage must first be performed to determine the output channel to which the packet can be forwarded. Next, in the VA stage, the packet requests an output VC from the VC allocator. Once a route and an output VC have been determined, the packet proceeds to the SA stage where it arbitrates for the switch input and output ports. On winning the output port the packet then proceeds to the ST stage where it traverses the crossbar. This four-stage pipeline of a packet transfer at each router causes progressive latency on the interconnection networks.

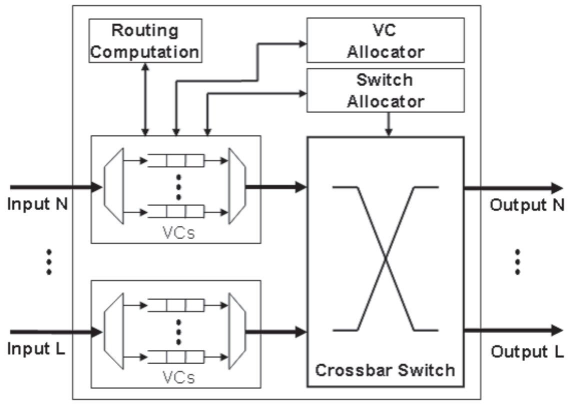

Fig. 1 Architecture of a conventional VC router.

2.2 Motivation

As mentioned above, in the conventional architecture before advancing to the SA stage and then ST stage, a packet is required to complete the RC and VA operations. To shorten this process, many designs utilize the speculation mechanism[5] in which multiple pipeline stages can be concurrently performed in a speculative fashion[7)–11)]. The speculation works well under a light network load, however it is very inefficient when the network becomes heavily loaded. Furthermore, to avoid the impact of speculation on throughput, there is a need for two separable switch allocators for non-speculative and speculative requests, with a higher priority assigned to non-speculative ones. This causes the design complexity and remains the limited throughput issue under heavy network load conditions.

Another technique is look-ahead routing, that reduces the number of pipeline stages by performing the routing computation ahead of time to remove it from the critical path[6]. Employing this mechanism allows shortening the four-cycle pipeline to a three-cycle. Combining the speculation and look-ahead routing schemes to attain a single-cycle pipeline seems to be a good way. However, speculatively performing numerous pipeline stages in parallel may lead to the degradation of frequency and the need for complex additional controls[6)–11)].

In addition, while concentrating on the reduction of the network latency, various router architectures enable the bypassing of some of the pipeline stages using express virtual channels for express channels[12], path frequency analyzers for determining fast paths[13], preferred path detectors for defining pre-configured preferred paths[14] or predictors for bypassing channels[15]. However, these techniques usually require an additional hardware cost that may cause the complexity of hardware design.

The dimension-sliced technique is a low-cost alternative that partitions the router into two separate smaller routers[16]. These small routers are created as bufferless routers, and intermediate buffers between them are used to decouple the flow control and routing for reducing the cost of on-chip networks. With providing the priority for the packets already in the network that continue to advance to the same dimension, a single-cycle packet transfer can be achieved. However, this scheme can cause starvation, with the result that fairness may become an issue. In addition, this router architecture is restricted to the dimension-ordered routing algorithm used in the two-dimensional mesh topology, and may be hard to employ for various other network topologies and routing algorithms.

In this work, we present a router architecture that allows minimizing the critical path, while maintaining the simplicity of hardware design without the restriction on specific routing algorithms and topologies. Instead of speculatively performing the VA in parallel with the SA, our proposed router only performs the VA for the packets which have won the switch arbitration. A VC is allocated for a packet during the time the packet is traversing the crossbar switch. To avoid the unexpected deadlocks between packets in the VA and SA stages, and ensure that the performance is not degraded in the cases where a packet wins the switch arbitration but no VC is available for being allocated, the switch requests from incoming packets not yet allocated a VC, are generated to the switch allocator only if both the following conditions are satisfied:

(1) an output channel has been determined for the packets, and
(2) there is at least one free VC at the required output channel.

With this constraint, the dependency between the VA and ST is removed, and these stages can be concurrently performed in a non-speculative fashion. This manner of the VC allocation allows a packet to directly enter the SA without performing any speculation. As a result, there is no impact of speculation on throughput even in the heavy network loads, and the efficiency of channel utilization is significantly enhanced.

3. Proposed On-the-fly VC Allocation Architecture

In this section, we describe the architecture of proposed on-the-fly VC allocation router on a two-dimensional mesh network for exploiting its design regularity. The *XY routing* algorithm is adopted for guaranteeing deadlock-free routing in the two-dimensional mesh network, and *iSLIP scheduling* algorithm is employed for the switch allocation because of its high performance, flexibility and no starvation[17]. In addition, in this work the look-ahead routing technique is adopted as well, to remove the routing computation from the critical path for further reduced communication latency. The packets are divided into multiple flits (flow control digits) where the head flit contains all the necessary routing information (including a route for the next hop), and the others contain payload data.

3.1 On-the-fly VC Allocation

The block diagram of our proposed router architecture is shown in **Fig. 2**. In this architecture, the VC allocator is replaced by an output VC access control module. The output VC access control module consists of an *access matrix* for each output VC. The access matrix is a control logic that operates as a bit mask for controlling the access of incoming packets to the output VC. At the initial time, the access matrix is set to be accessible for all incoming packets of input ports. On running, it is updated to an appropriate value at each cycle, depending on the result of switch arbitration. The operation of an access matrix is explained later in Section 3.2.

Unlike the conventional VC model and other existing on-chip router architectures mentioned above, an incoming packet, once it has determined its output channel with one or more free VCs for being allocated, directly enters the SA

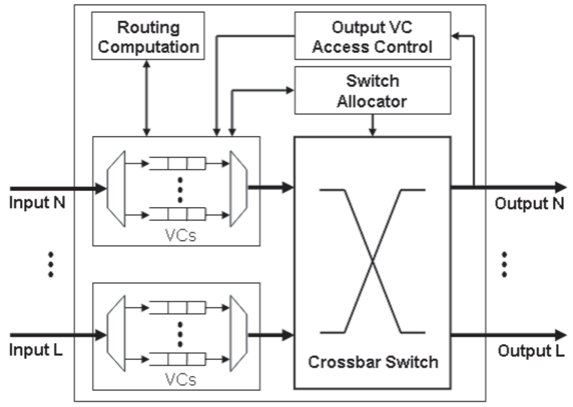

Fig. 2 Architecture of proposed on-the-fly VC allocation router.

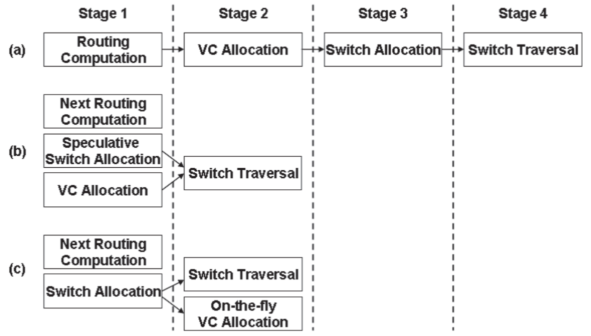

Fig. 3 Pipeline-stage dependencies of (a) a conventional VC router, (b) a look-ahead speculative VC router, (c) a look-ahead on-the-fly VC allocation router.

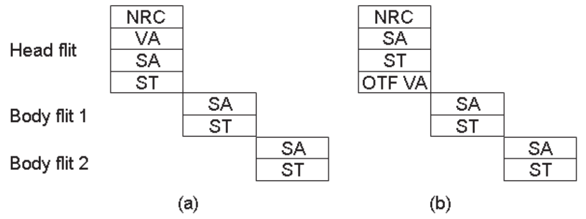

Fig. 4 One-cycle pipelines of (a) a look-ahead speculative router, (b) a look-ahead on-the-fly VC allocation router.

stage without performing any speculation. If there is no free VC at the output channel, the incoming packet is deactivated (i.e., no request is sent to the switch allocator) until the output channel becomes available. On winning the switch arbitration (i.e., a path to the output channel is granted), the packet traverses the crossbar to a free VC of the output channel during the ST stage. At this point, this output VC is allocated for the packet and becomes unavailable for all the other incoming packets. This state of the output VC is kept until the entire packet is transferred to its desired output channel. When the last flit of the packet is encountered, the output VC is released and becomes available for all packets that are destined for the output channel. This way of allocating an output VC for a packet is called *on-the-fly* VC allocation.

With on-the-fly VC allocation, a VC router removes the dependency between the VA and ST stages, and cuts down on its critical path delay. The comparison of pipeline-stage dependencies among our proposed router, conventional VC router, and look-ahead speculative VC router is shown in **Fig. 3**. Although the number of pipeline stages is similar, our proposed solution differs from the speculative solution in that the VC allocation is only performed for winning packets in the switch arbitration. Consequently, the VA and ST can be concurrently performed in a non-speculative fashion, and a higher throughput can be achieved.

In addition, the pipeline of a packet transfer can be minimized to a one-cycle process if the ST and SA stages are speculatively performed in parallel as shown in **Fig. 4** (b), where the next routing computation is abbreviated as NRC and the on-the-fly VC allocation is denoted as OTF VA. In this manner, both the ST and VA are performed simultaneously with the success of SA speculation. It is different from the speculative model in which the ST is performed only if both the SA and VA speculations succeed (Fig. 4 (a)). Due to the depth of a single speculation, the successful probability in our proposed architecture is higher than the speculative architecture.

In this work, with the aim of exploiting the design regularity of the on-the-fly VC allocation, the deadlock avoidance of dimension-ordered routing is used for the simplification of implementation. In cases of adaptive routings, the other deadlock-free approaches such as the *structured buffer pool*[18], *minimal or non-*

minimal adaptive routing, or *turn model*[19] can be employed for our proposed router architecture. However, the look-ahead routing technique may be hard to adopt, and an additional control may be required for an aggressive reduction of the critical path.

3.2 Operation of Packet Transfer

At the input channel, when the head flit of a new incoming packet is detected, an appropriate VC is prepared based on its VC identification. Since the route has been computed at the previous hop, the output channel of the head flit is simply determined by the current NRC. A new NRC for the next hop is now calculated based on the current routing information. Before the head flit is stored to its VC, the new routing information is also updated. All the other flits of the packet follow the head flit to the same VC in a pipeline fashion. When a flit reaches the head of a VC and its required output channel is available, a request is sent to the switch allocator to get a grant for passage of the flit to the desired output channel. In our design, the VCs of input ports are implemented as FIFO (First In First Out) queues, with the use of a simple on/off flow control mechanism for simplifying the control logic needed to maintain the buffer state.

Incoming packets from input VCs that are destined for the same output channel compete for one or more of the remaining free output VCs. Once the head flit of a packet wins the arbitration on the crossbar, then it is allocated to one of the free output VCs. This output VC is reserved for all the other flits of the same packet, and becomes unavailable for any other packet. The flits traverse the output channel and leave for the next hop with their VC identification updated to the allocated output VC. To ensure the sequential transfer of a packet, each output channel maintains an access matrix for controlling the access of incoming packets to its output VCs, as illustrated in **Fig. 5**. Each access matrix maintains a separate access vector (AV) for each output VC (oVC). The access vector is a bit mask (an 8-bit mask in this figure, assuming that each input port is composed of two VCs) used for controlling the access of incoming packets from input VCs (iVCs) to the output VC. Logic "1" indicates the accessible state of the output VC, and in contrast logic "0" indicates the inaccessibility.

When the head flit of a packet advances along an output VC, the output VC is required to be held for all remaining flits of the packet. During the packet transfer, all the other packets of input VCs cannot access this output VC. The control is performed by maintaining an appropriate value for the corresponding access vector of the output VC. When the last flit of the packet leaves the router, the output VC is deallocated and its state is set to be available for all incoming packets. For example, when the incoming packet from the VC_0 of input port S takes a path to the output port N, it is allocated to the free output VC_0. At the instant that a head flit is identified on the output VC_0, the AV_0 of the access matrix N is immediately updated to "10000000" as available for only the VC_0 of input port S. In the next cycle, the packet from the VC_0 of input port L wins the switch arbitration and occupies the remaining free output VC_1. At this point, the AV_1 is set to "00000010" for busy. While these packets advance along their output VCs, packets from the other input VCs cannot be forwarded to the output port N. When a tail flit is presented on the output VC_0 (or VC_1), the AV_0 (or AV_1) of the access matrix N is returned to "11111111" as available for any packet of input VCs.

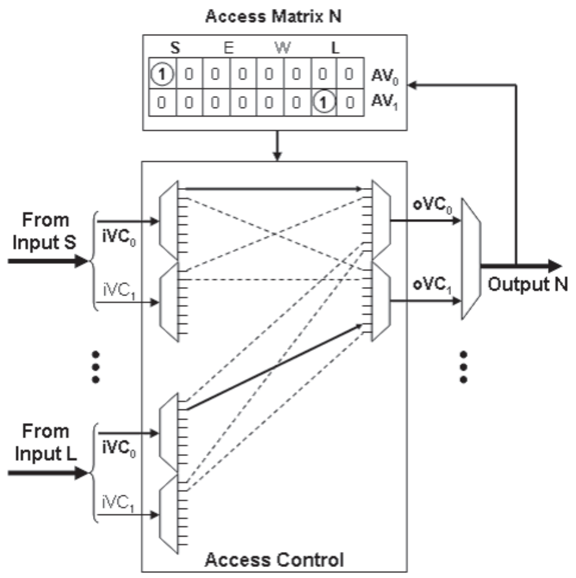

Fig. 5 Output VC access control logic.

Table 1 Implementation results of routers.

	Baseline (4-cycle)	LA+Spec (2-cycle)	LA+OTF (2-cycle)	LA+Spec (1-cycle)	LA+OTF (1-cycle)
Number of ports	5	5	5	5	5
Number of VCs	2	2	2	2	2
Data width	16-bit	16-bit	16-bit	16-bit	16-bit
Buffer's size	4-flit	4-flit	4-flit	4-flit	4-flit
Slices	398	472	389	461	384
LUTs	1,477	1,591	1,480	1,683	1,449
Flip-Flops	240	410	425	355	360
BlockRAM	5 blocks	5 blocks	5 blocks	5 blocks	5 blocks
Frequency	166 MHz	165 MHz	166 MHz	104 MHz	110 MHz

Table 2 Breakdown of area.

	Baseline (4-cycle)			LA+Spec (2-cycle)			LA+OTF (2-cycle)			LA+Spec (1-cycle)			LA+OTF (1-cycle)		
	Slices	LUTs	FFs	Slices	LUTs	FFs	Slices	LUTs	FFs	Slices	LUTs	FFs	Slices	LUTs	FFs
IU	210	876	130	309	1,060	300	241	1,051	300	298	1,152	245	235	1,010	235
RC	25	70	-	-	-	-	-	-	-	-	-	-	-	-	-
VA	83	296	100	83	296	100	-	-	-	83	296	100	-	-	-
SA	9	25	10	9	25	10	9	25	10	9	25	10	9	25	10
CB	71	210	-	71	210	-	71	210	-	71	210	-	70	220	-
AC	-	-	-	-	-	-	68	194	115	-	-	-	68	194	115

In our design, the switch allocation must be performed after the access matrices have completed their updates. To ensure the router frequency does not suffer by this dependency in the two-cycle pipeline design, the switch grant-enable signal of an output arbiter is kept unchanged in one cycle after a head flit transfer, and deactivated in one cycle after a tail flit transfer. In this manner, the router can operate at a high speed with a trivial impact on the network performance.

4. Evaluation

In this work, we have implemented and evaluated five architectures of router - the conventional four-cycle pipeline VC router (baseline), look-ahead speculative (LA+Spec) and look-ahead on-the-fly VC allocation (LA+OTF) routers with the two-cycle pipeline and one-cycle pipeline, respectively. All routers have the same parameters as five bi-directional ports, two VCs per port, 16-bit data width and 4-flit buffer size.

The router core is developed on FPGA with the use of Xilinx Virtex-5 XC5VFX70T that is composed of 11,200 slices and 148 BlockRAM blocks for a target device, Verilog-HDL for the circuit design, and ModelSim XE III 6.3c for the functional and structural simulation. The Xilinx integrated tool environment ISE 10.1i is used for the automated logic synthesis, mapping, placing and routing of circuits.

4.1 Implementation Results

The implementation results of routers summarized in most of the primary characteristics of NoC are shown in **Table 1**. The result shows that, our proposed LA+OTF router with the two-cycle pipeline can operate at a maximum frequency of 166 MHz, that is similar to the baseline router and LA+Spec router with the pipeline of two-cycle. When implemented with the pipeline of one-cycle, its frequency is decreased by 33.7% as compared to that of the two-cycle pipeline architecture, because many pipeline stages are concurrently performed in one clock cycle. However, in comparison with the same one-cycle pipeline architecture of the LA+Spec it can operate with a higher frequency.

In terms of area overhead (not including BlockRAMs used for buffers), the proposed two-cycle pipeline LA+OTF router consumes only 389 slices that is slightly smaller than the baseline router, and reduced by 17.6% as compared to the two-cycle pipeline LA+Spec design. This reduction of area is due to no control logic being needed for the speculative switch allocation. Similarly, in the one-cycle pipeline design, the LA+OTF router reduces the area by 16.7% as compared to the LA+Spec design, and still maintains smaller than that of the baseline design. For more detailed evaluation, the breakdown of area in IU (input unit), RC (routing computation), VA (VC allocation), SA (switch allocation), CB (crossbar), and AC (access control) is also listed in **Table 2**. As a result, the AC consumes a smaller number of slices than the VA, and the IU of our proposed design requires less hardware resource than that of the speculative solution, while the others are similar.

Obviously, the architecture based on our proposed on-the-fly VC allocation not only reduces the number of pipeline stages as compared to the baseline design, but also brings the simplicity of hardware design with a significant reduction of area in comparison with the speculative solution.

4.2 Latency and Throughput

Experiments are conducted to evaluate the performance of the proposed on-the-fly VC allocation architecture against the conventional VC and look-ahead speculative designs, using a 4×4 two-dimensional mesh network developed in Verilog. Packets are generated at a constant rate and queued until they are able to enter the network. Latency of a packet is measured from the time when the first flit is injected into the packet source queue, to the time when its last flit is ejected from the network, assuming immediate ejection. The simulation is performed in two phases of warm-up and measurement with an assumption of 5-flit length for all injected packets. A simulation is terminated if all packets are received at the destination nodes under normal conditions. Otherwise, the simulation terminates if the average latency of the packets exceeds a threshold of 100 cycles. All nodes inject their flits at the same time for the worst case of a high network load. All simulations are performed under the uniform random traffic. The network latency is shown as a function of the injection rate.

Figure 6 shows the network latencies of the two-cycle pipeline LA+OTF and LA+Spec routers, and the four-cycle pipeline baseline router with the buffer size of 4-flit. The figure shows that the zero-load latency of the baseline router is 22 cycles, while it is 16 cycles in our proposed router that is similar to the LA+Spec router. This 27.3% improvement of latency in comparison with the baseline router is due to the reduction of pipeline stages from four to two stages. The throughput of our proposed router saturates at about 51% of capacity, that is improved by 21.4% and 6.2% as compared to the throughput of the baseline and LA+Spec routers, respectively. This considerable increase of throughput is reasonable, because in the LA+OTF router there is no impact caused by the speculation failures between the VA and SA stages.

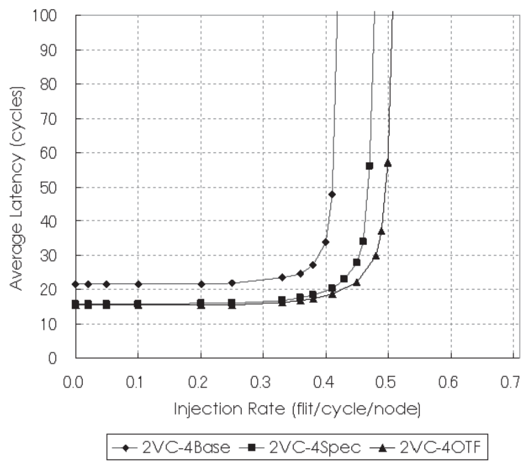

Fig. 6 Network latencies of 2-cycle pipeline routers (except baseline) in cycle normalization (2VC-4 = two 4-flit VCs/port; Base = Baseline, Spec = Look-ahead speculative, OTF = Look-ahead on-the-fly VC allocation).

The zero-load latency of our proposed one-cycle pipeline architecture is further reduced to 13 cycles as shown in **Fig. 7**. It is 40.9% lower than that of the baseline router. Besides, the throughput is also extended to 62% of capacity, that improves by 47.6% and 8.8% as compared to the baseline and one-cycle pipeline LA+Spec routers, respectively. **Figures 8** and **9** show the comparisons of network latency in time normalization. In any figure, our proposed on-the-fly VC allocation router is pointed out as the best result in terms of communication

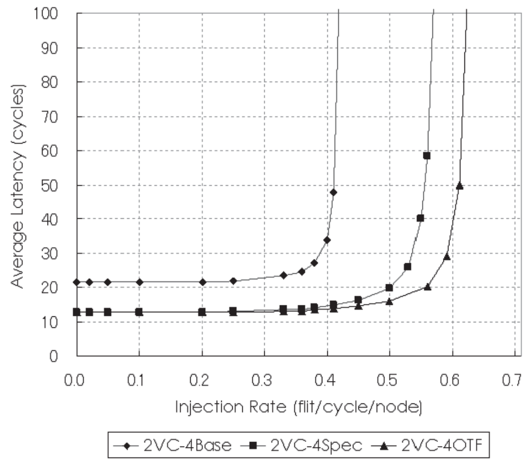

Fig. 7 Network latencies of 1-cycle pipeline routers (except baseline) in cycle normalization (2VC-4 = two 4-flit VCs/port; Base = Baseline, Spec = Look-ahead speculative, OTF = Look-ahead on-the-fly VC allocation).

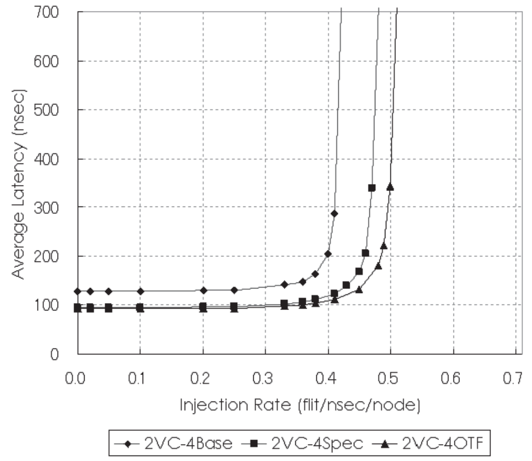

Fig. 8 Network latencies of 2-cycle pipeline routers (except baseline) in time normalization (2VC-4 = two 4-flit VCs/port; Base = Baseline, Spec = Look-ahead speculative, OTF = Look-ahead on-the-fly VC allocation).

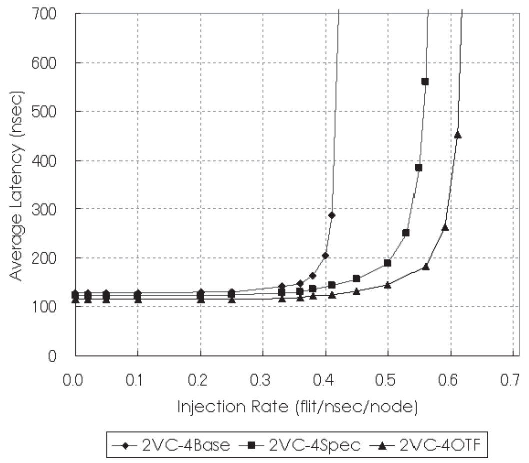

Fig. 9 Network latencies of 1-cycle pipeline routers (except baseline) in time normalization (2VC-4 = two 4-flit VCs/port; Base = Baseline, Spec = Look-ahead speculative, OTF = Look-ahead on-the-fly VC allocation).

latency and throughput.

These experimental results show that, the router design based on the on-the-fly VC allocation outperforms the conventional VC router in terms of communication latency and throughput without the penalty of area. It also outperforms the look-ahead speculative architecture in terms of throughput and hardware amount.

5. Conclusion

In this paper, we have presented an architecture of low-latency high-throughput router based on the on-the-fly VC allocation that removes the dependency between the VC allocation and switch traversal. In the manner of performing the VC allocation based on the result of switch allocation, the pipeline of a packet transfer can be shortened in a non-speculative fashion and a higher throughput can be achieved. The experimental result showed that the design not only maintains the simplicity, but also significantly improves the overall performance of the network. The proposed router with on-the-fly VC allocation can reduce the network latency by 40.9%, and improve throughput by 47.6% as compared to the conventional VC router. In comparison with the look-ahead speculative router, it improves the throughput by 8.8% with 16.7% reduction of area for control logic. This fact indicates the ability of our proposed architecture to provide low cost high performance on-chip networks for practical SoCs.

References

1) Benini, L. and De Micheli, G.: Networks on Chips: A New SoC Paradigm, *IEEE Computer*, Vol.35, No.1, pp.70–78 (2002).
2) Owens, J.D., Dally, W.J., Ho, R., Jayasimha, D.N., Keckler, S.W. and Peh, Li-S.: Research Challenges for On-Chip Interconnection Networks, *IEEE Micro*, Vol.27, No.5, pp.96–108 (2007).
3) Dally, W.J. and Towles, B.: Route Packets, Not Wires: On-Chip Interconnection Networks, *Proc. 38th ACM Design Automation Conference*, pp.684–689, Las Vegas, Nevada, USA (2001).
4) Bjerregaard, T. and Mahadevan, S.: A Survey of Research and Practices of Network-on-Chip, *ACM Computing surveys*, Vol.38, No.1, pp.1–51 (2006).
5) Peh, Li-S. and Dally, W.J.: A Delay Model and Speculative Architecture for Pipelined Routers, *Proc. 7th International Symposium on High-Performance Computer Architecture*, pp.255–266 (2001).
6) Dally, W.J. and Towles, B.: *Principles and Practices of Interconnection Networks*, Morgan Kaufmann (2004).
7) Kim, J., Park, D., Theocharides, T., Vijaykrishnan, N. and Das C.R.: A Low Latency Router Supporting Adaptivity for On-chip Interconnects, *Proc.the 42nd Design Automation Conference* (*DAC'05*), pp.559–564 (2005).
8) 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. International Symposium on Computer Architecture* (*ISCA'06*), pp.4–15 (2006).
9) Mullins, R., West, A. and Moore, S.: Low-Latency Virtual-Channel Routers for On-Chip Networks, *Proc. 31st International Symposium on Computer Architecture*, pp.188–197 (2004).
10) Mullins, R., West, A. and Moore, S.: The Design and Implementation of a Low Latency On-Chip Network, *Proc. Asia and South Pacific Design Automation Conference* (*ASP-DAC'06*), pp.164–169 (2006).
11) Bertozzi, D. and Benini, L.: Xpipes: A Network-on-Chip architecture for gigascale Systems-on-Chip, *IEEE Circuits and Systems Magazine*, Vol.4, No.2, pp.18–31 (2004).
12) Kumar, A., Peh, Li-S., Kundu, P. and Jha, N.K.: Express Virtual Channels: Towards the Ideal Interconnection Fabric, *Proc. International Symposium on Computer Architecture* (*ISCA'07*), pp.150–161 (2007).
13) Park, D., Das, R., Nicopoulos, C., Kim, J., Vijaykrishnan, N., Iyer, R. and Das, C.R.: Design of a Dynamic Priority-Based Fast Path Architecture for On-Chip Interconnects, *Proc. IEEE Symposium on High-Performance Interconnects* (*HOTI'07*), pp.15–20 (2007).
14) Michelogiannakis, G., Pnevmatikatos, D.N. and Katevenis, M.: Approaching Ideal NoC Latency with Pre-Configured Routes, *Proc. International Symposium on Networks-on-Chip* (*NOCS'07*), pp.153–162 (2007).
15) Matsutani, H., Koibuchi, M., Amano, H. and Yoshinaga, T.: Prediction Router: Yet Another Low Latency On-Chip Router Architecture, *Proc. IEEE International Symposium on High-Performance Computer Architecture* (*HPCA'09*), pp.367–378 (2009).
16) Kim, J.: Low-cost Router Microarchitecture for On-chip Networks, *Proc. 42nd IEEE/ACM International Symposium on Microarchitecture*, pp.255–266 (2009).
17) McKeown, N., Anantharam, V. and Walrand, J.: Achieving 100% Throughput in an Input-Queued Switch, *IEEE Trans. Commun.*, Vol.47, No.8, pp.1260–1267 (1999).
18) Raubold, E. and Hänle, J.: Method of Deadlockfree Resource Allocation and Flow Control in Packet Networks, *Proc. 3rd International Conference on Computer Communication*, pp.483–487 (1976).
19) Ni, L.M. and McKinley, P.K.: A Survey of Wormhole Routing Techniques in Direct Networks, *IEEE Computer*, Vol.26, No.2, pp.62–76 (1993).

(Received July 26, 2010)
(Accepted November 2, 2010)

Son Truong Nguyen received his B.Sc. degree in Electrical Engineering from Le Quy Don Technical University, Vietnam in 1996 and his M.E. degree in Electrical and Computer Engineering from the National Defense Academy, Japan in 2004. He is currently a doctoral student at the Graduate School of Science and Engineering, Ritsumeikan University, Japan. His research interests focus on networks-on-chip, interconnection networks and computer architectures. He is a member of IPSJ.

Shigeru Oyanagi is a professor at the Department of Computer Science, College of Information Science and Engineering, Ritsumeikan University, Japan. He received his M.E. and Ph.D. degrees from Kyoto University in 1974 and 1979, respectively. His research interests include parallel processing, computer architecture, database and data mining. He is a member of IEICE, IPSJ, ACM and IEEE.