A Virtual Channel Allocation Algorithm for NoC

Dongxing Bao[1], Xiaoming Li[2(✉)], Yizong Xin[3], Jiuru Yang[1], Xiangshi Ren[4], Fangfa Fu[2], and Cheng Liu[2]

[1] School of Electronic Engineering, Heilongjiang University, Harbin 150080, China
[2] Department of Microelectronics Science and Technology, Harbin Institute of Technology, Harbin 150001, China
lixiaoming@hit.edu.cn
[3] School of Information Engineering, Shenyang University of Technology, Shenyang 110870, China
[4] School of Information, Kochi University of Technology, Kochi 780-8520, Japan
eastarbox@163.com

Abstract. Virtual channel (VC) flow control proves to be an alternative way to promote network performance, but uniform VC allocation in the network may be at the cost of chip area and power consumption. We propose a novel VC number allocation algorithm customizing the VCs in network based on the characteristic of the target application. Given the characteristic of target application and total VC number budget, the block probability for each port of nodes in the network can be obtained with an analytical model. Then VCs are added to the port with the highest block probability one by one. The simulation results indicate that the proposed algorithm reduces buffer consumption by 14.58%–51.04% under diverse traffic patterns and VC depth, while keeping similar network performance.

Keywords: VC allocation · Block probability · Network-on-chip

1 Introduction

SoC designs are confronted with various challenges caused by the increasing complexity of the designs [1]. The on-chip communication bandwidth requirement is growing rapidly, and simultaneously the interconnect delay exceeds the average on-chip clock period [2]. Network-on-chip (NoC) which replaces bus with network to implement the communication among processing elements (PE) has been proposed [3, 4], and becomes one of the most promising on-chip interconnection architectures [5]. NoC is composed of Network Interface (NI), Router, and Link basically. Compared with traditional on-chip bus, NoC has many advantages—reusability, scalability, parallelism, etc., which satisfies future SoC interconnection requirements [3–5].

The benefits of NoC are attractive, but attaining their full potential will present lots of challenges among which power consumption stands out as one of the most critical

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2018
X. Gu et al. (Eds.): MLICOM 2017, Part II, LNICST 227, pp. 333–342, 2018.
https://doi.org/10.1007/978-3-319-73447-7_37

challenges [5]. Since router is one of the kernel components of NoC and it has significant influence on both the performance and power consumption of NoC [6], we mainly focus on it in this paper. A typical virtual channel router structure is shown in Fig. 1. It is composed of Routing Computation (RC), VC Allocator (VA), Switch Allocator (SA), VC, crossbar, and Mux as well as Demux. Among the five parts of router, VC has prominent effect on router. However, VC takes up most of the power and area consumption of router [7], especially more than sixty percent of the static power consumption [8, 9]. On the other hand, VC number and capacity determines the router performance with specific router architecture. With more VCs and larger capacity, data in the network will be able to forward to the destination node more fluently, while under the opposite situation, the network are prone to get blocked and saturated. To get out of such a dilemma, the researchers proposed many strategies from diverse points of view. The authors of [10, 11] introduce power gating to shut down idle VCs. Although power gating can reduce the system static power consumption, it needs additional technology support. What's worse, the static power saving will decrease greatly under heavier workload. The authors of [12, 13] propose dynamic buffer allocation strategy adjusting the VC number or depth based on network status. The strategy can be effective in diverse traffic; however, the designs are usually complex and result in additional hardware consumption. Therefore little power saving can be achieved when the data width is not very large. It is tough to find out a general method to make compromise on performance and power consumption.

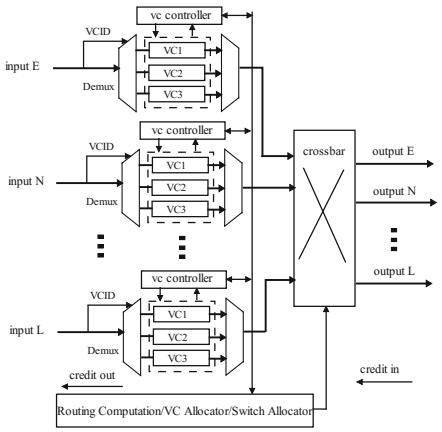


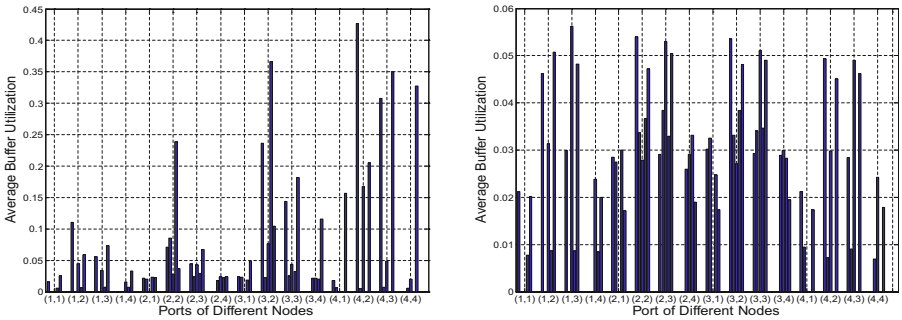
Fig. 1. A general virtual channel router architecture.

NoC design typically aims at certain specific application, thus the NoC architecture can be customized to specific application to obtain the best design trade-offs [14]. Taking this matter into consideration, Hu and Marculescu work out an analytical performance model for NoC, and then proposed a buffer capacity allocation algorithm based on the performance model [7]. According to the algorithm, the buffer with the highest full probability is assigned larger buffer capacity. However, such an allocation algorithm is limited to router with single buffer channel. Consequently, when it comes

to higher throughput requirements application, router based on VC is necessary and the algorithm will be not available. To remedy this situation, Huang et al. [15] develop a VC planning algorithm. The algorithm only adds VCs to channels which present the highest bandwidth usage. Although the algorithm presents prominent power savings when the VC is deep, it doesn't work well when the VC is shallow. In fact, the probability due to feedback of VCs in the next router can't be ignored, especially when the VCs are shallow. In this paper, a novel VC number allocation algorithm is proposed. Taking both arbitration contention and VC feedback probability into account, we add VCs to port with the highest block probability.

2 Buffer Utilization Characteristic for NoC

To get a better view of VC utilization characteristic in NoC, we make a stat. of VC utilization under hotspot traffic with a cycle accurate SystemC simulator. X-Y routing algorithm is adopted in a 4 × 4 2D Mesh NoC. And there are three four-flit-VCs in each input port of the router. Average injection rate is 0.3 flit/node/cycle and the hotspot locates in (2, 2). The simulation begins with a warm-up period of 100000 cycles. Then performance data is collected 100000 cycles later. Figure 2-(a) shows the buffer utilization in different ports of different nodes. It is obviously that the buffer utilization of different nodes differs notably. Even buffer utilization in the same node varies significantly. The lowest buffer utilization in the network is 0.0053, while the highest is 0.427 and it is 80 times higher than the lowest! Figure 2-(b) indicates the buffer utilization under uniform traffic. It is amazing that buffer utilization differs greatly even under uniform traffic. The highest buffer utilization is 0.0562, which is about 8 times higher than the lowest.



(a) Traffic with Hotspot. (b) Uniform.

Fig. 2. VC utilization of different ports in each node.

The simulation results illustrate that unbalance of buffer utilization exits in diverse traffic patterns including uniform traffic. The main reason is 2D Mesh topology as well as X-Y deterministic routing. In fact, the application characteristic, routing algorithm, topology etc. have notable effect on buffer utilization in network. Therefore,

unbalanced buffer utilization in NoC is usual. As buffer utilization varies across the network, with uniform VC allocation, VCs in some ports might be idle and wasted, while VCs in some other ports might be insufficient and a mass of data might be blocked. Hence customizing the VCs configuration in NoC based on the application characteristic will decrease buffer consumption through reducing superfluous VCs and alleviate block by adding VCs. At the same time, the network performance can be maintained or even improved.

3 Static Virtual Channel Number Allocation

3.1 Basic Idea

To fully utilize the limited VCs in the network and satisfy the performance requirements with the least VCs, VCs are allocated across the network based on needs of the ports. But the problem is how to define the needs of ports in the network. To make the problem clear, we explain how the data is transmitted in a router first. Router receives and stores data injected from the input ports. When the data in the VCs gets through the Mux, and there are available VCs in the next router, it sends request to VA. With the grant of VA, it still needs to make sure that the assigned VC is not full before the data sends request to SA for the output port. With the grant of SA, the data will be sent to crossbar and then leaves from the output port. All the data that fails to request or be granted by VA or SA is stored in VCs of current router. In other words, all the data that is blocked will be stored. The higher the block probability is, the more data needs to be stored and the worse the performance is. Therefore alleviating the block probability will be one of the possible ways to improve the network performance.

The root of block is limited bandwidth of the link and limited buffer capacity of the router. Generally speaking, the bandwidth of the link is fixed. Under this situation, especially when the link bandwidth of the link is sufficient, we have to turn to the buffer capacity. Buffer is used to smooth the injection rate and the ejection rate of the router. Increasing buffer capacity of the corresponding port in the next router will alleviate the block probability and increase the ejection rate. As buffer capacity is limited, we only increase VCs where the block probability is high. As a result, there are fewer VCs where the block probability is low. However, when the link becomes saturated, adding VC will bring few benefits. On the opposite, the hardware consumption increases. So VC number in each port will be limited. Therefore we develop a VC allocation algorithm based on block probability. Compared with uniform VC configuration, the buffer consumption is reduced significantly, while achieving similar performance. The analytical model of the blocking probability and VC number allocation algorithm will be described in detail below.

3.2 Problem Formulation

For convenience, the notations used in analysis are listed in Table 1. And the problem of VC number allocation can be formulated as follow.

Table 1. Parameter notion.

Parameter	Description
S	Total VC number budget
$\lambda_{x,y}$	PE injection rate of node (x, y)
$P_{x,y}^{x',y'}$	The probability that node (x, y) sends data to node (x', y')
R	Routing function
D	VC depth
U	Port number of the router
$v_{x,y,j}$	VC number of input port j in node (x, y) , $j \in \{E, N, W, S, L\}$
$\lambda_{x,y,j}$	Injection rate of input port j in node (x, y) , $j \in \{E, N, W, S, L\}$
$\lambda_{x,y,j,k}$	Flit transmission rate of node (x, y) that injects from input port j in and ejects from output port k, $j \in \{E, N, W, S, L\}$, $k \in \{E, N, W, S, L\}$
$\mu_{x,y,j}$	Service rate of input port j in node (x, y), $j \in \{E, N, W, S, L\}$
$\rho_{x,y,j}$	Traffic intensity of input port j in node (x, y), $j \in \{E, N, W, S, L\}$
$P_{input\_block\_x,y}$	Block probability of input port j in node (x, y) , $j \in \{E, N, W, S, L\}$
$P_{block\_x,y,k}$	Block probability of corresponding output port in previous node of input port k in node (x, y) , $k \in \{E, N, W, S, L\}$
$P_{ARB\_con\_x,y,k}$	Arbitration contention probability of output port k in node (x, y) , $k \in \{E, N, W, S, L\}$
$P_{ARB\_con\_x,y,k}$	Arbitration contention probability of n flits requesting for output port k in node (x, y), $k \in \{E, N, W, S, L\}$, $n \in \{0, 1, 2, 3, 4, 5\}$
$P_{VC\_full\_con\_x}$	VC full probability of input port k in node (x, y) , $k \in \{E, N, W, S, L\}$

Assume:

PE injects packet with a Poisson distribution.

Given:

Total VC number budget, S
Application communication characteristic, $\lambda_{x,y}$ and $P_{x,y}^{x',y'}$
Routing algorithm, R
Virtual channel depth, D

Determine:

VC configuration $v_{x,y,j}$
Which minimizes average network latency Lat

$$\text{Min}(Lat) \qquad \text{s.t.} \qquad \sum_{\forall x} \sum_{\forall y} \sum_{\forall j} v_{x,y,j} \leq S \tag{1}$$

3.3 Block Probability Analysis

For convenience, single VC is configured in each input port. $\forall x, \forall y, j \in \{N, E, S, W, L\}$, $v_{x,y,j} = 1$. With previous analysis, it is natural that the block probability is consist of two aspects—arbitration probability including VA and SA and VC feedback probability. Therefore $P_{block\_x,y,k}$ can be derived:

$$P_{block\_x,y,k} = 1 - \left(1 - P_{ARB\_con\_x',y',k'}\right) \times \left(1 - P_{VC\_full\_con\_x,y,k}\right) \tag{2}$$

Node (x', y') indicates the node that connects with input port k in node (x, y) ,and k' is corresponding output port in node (x', y').

When more than two flits request for the same output port k, there is an arbitration contention. Then the arbitration contention probability is:

$$P_{ARB\_con\_x,y,k} = \sum_{n=2}^{U} P_{ARB\_con\_x,y,k,n} \tag{3}$$

$$\begin{aligned} P_{ARB\_con\_x,y,k,n} = {} & \left(1 - \lambda_{x,y,E,k}\right) \times \cdots \times \left(1 - \lambda_{x,y,j,k}\right) \\ & \times \underbrace{\lambda_{x,y,j+1,k} \times \cdots \times \lambda_{x,y,j+n,k}}_{n} \\ & \times \left(1 - \lambda_{x,y,j+n+1,k}\right) \times \cdots \times \left(1 - \lambda_{x,y,L,k}\right) \end{aligned} \tag{4}$$

$$\lambda_{x,y,j} = \sum_{k \in \{N,E,S,W,L\}} \lambda_{x,y,j,k} \tag{5}$$

$$\lambda_{x,y,j,k} = \sum_{\forall x_s, y_s} \sum_{\forall x_d, y_d} R(x_s, y_s, x_d, y_d, x, y, j, k) \times \lambda_{x_s,y_s} \tag{6}$$

$R(x_s, y_s, x_d, y_d, x, y, j, k)$ indicates the routing algorithm. When there is data transmitted from source node (x_s, y_s) to destination node (x_d, y_d), and the data is injected from input port j of node (x, y) and leaves from output k of node (x, y), R returns 1, or else it returns 0.

The full probability of VC can be calculated with M/M/1/K queuing model.

$$P_{VC\_full\_con\_x,y,k} = \frac{1 - \rho_{x',y',k'}^{D}}{\left(1 - \rho_{x',y',k'}\right)^{D+1}} \tag{7}$$

Node (x', y') is the node that corresponds to output port k of node (x, y), and k' is the corresponding input port of node (x', y').

Without block, the service rate is 1. Thus the service rate of the queue can be approximated:

$$\mu_{x,y,j} = 1 - P_{input\_block\_x,y,j} \tag{8}$$

$$P_{input\_block\_x,y,j} = \sum_{k=1}^{N} P_{x,y,j,k} \times P_{output\_block\_x,y,j,k} \tag{9}$$

$$P_{output\_block\_x,y,j,k} = \sum_{i\in\{N,E,S,W,L\},i\neq j} \lambda_{x,y,i,k} \tag{10}$$

$$P_{x,y,j,k} = \frac{\lambda_{x,y,j,k}}{\sum_{i\in\{N,E,S,W,L\}} \lambda_{x,y,i,k}} \tag{11}$$

Then the traffic intensity of the queue is:

$$\rho_{x,y,j} = \lambda_{x,y,j} / \mu_{x,y,j} \tag{12}$$

When there are multiple VCs in ports of the router, as an output port won't be blocked before all the VCs are blocked. Therefore, $P_{block\_x,y,k}$ can be approximated:

$$P_{block\_x,y,k} = \left(1 - \left(1 - P_{VA\_con\_x',y',k'}\right) \times \left(1 - P_{VC\_full\_con\_x,y,k}\right)\right)^{v_{x,y,k}} \tag{13}$$

Node (x', y') indicates the node that connects with input port k in node (x, y) ,and k' is corresponding output port in node (x', y').

3.4 VC Number Allocation Algorithm

As shown in Fig. 3, an effective greedy algorithm based on the aforementioned block probability model is proposed.

Given the design parameters (topology, routing algorithm and so on) and Communication Task Graph (CTG), the network scale and traffic characteristic can be acquired with mapping. At the same time, VC budget can be derived from power budget of the target application with NoC power model. Then the VC number allocation begins. First of all, the number of VCs in each port is initialized to one. Then the block probability of each output port can be calculated to find out the port with the highest block probability. If the VC number in the corresponding input port is equal to the upper VC limit *max_vc*, set the block probability zero and search for the next candidate. Or else, add a VC to the input port, and decrease by one from the total VC budget. Iteration restarts as long as the VC budget is still available. When the process stops, the VC configuration is derived.

max_vc is greatly influenced by the topology, packet length, traffic characteristic, VC number and depth, routing algorithm and son on, so it is almost impossible to determine with an analytical model. With specific simulation environment, we add VC uniformly. When the performance won't be improved notably, the VC number in each port is *max_vc*. *max_vc* used in this paper is four.

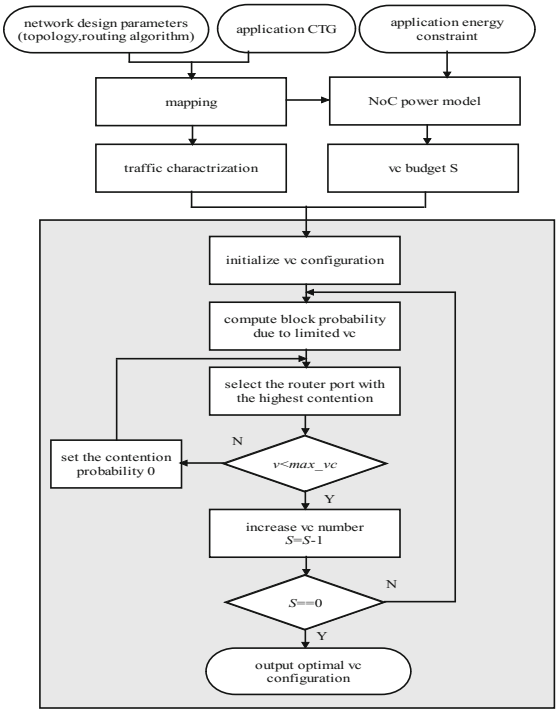


Fig. 3. VC allocation algorithm flowchart.

4 Simulation and Analysis

The simulation environment is based on a cycle-accurate, flit-level SystemC simulator.

During simulation, the data packets are yielded by a Poisson process at a definite traffic rate from the source PEs. 10000 packets are injected into each node and all the performance is abstracted after 100000 warm-up cycles. Detailed simulation condition is listed in Table 2. Besides, V0-NoC, V1-NoC and V2-NoC respectively indicate NoC

Table 2. Experiment condition.

Topology	2D mesh
Network scale	4 × 4
Temporal distribution	Poisson
Packet length	8
VC depth	4, 8, 16
Flit width	64 bit
Routing algorithm	X-Y deterministic routing
Arbitration algorithm	Round Robin
Flow control	Worm-hole

with uniform VC allocation, NoC with VC allocation algorithm proposed in [15] and NoC with VC allocation algorithm proposed in this paper.

Tables 3, 4 and 5 exhibit the average network latency when the VC depth is 4, 8, 16 respectively under diver traffic. It indicates that the system benefits from customizing VC configuration with both allocation algorithms under almost all the traffic patterns. Buffer consumption can be reduced by 14.58%–51.04% under diverse traffic patterns and VC depth, while keeping similar network performance. However, Compared with the algorithm proposed in [15], the algorithm developed in this paper achieves 4.17%–42.71% more buffer savings when the VC depth is 4. And it still performs better basically when the VC depth is 8. While when the VC depth is 16, the two algorithms acquire similar buffer savings. The reason for this is that the algorithm proposed in [15] doesn't take VC depth into consideration. In fact, when the VC is shallow, the block probability due to feedback of VC has notable influence on network performance. When the VC gets larger, the feedback probability can be ignored, therefore the two algorithms perform similar.

Table 3. NoC performance when VC depth is 4.

Evaluation metrics	Uniform distribution			Hotspot in the center			Hotspot in the edge			Hotspot in the corner		
	V0-NoC	*V1-NoC*	*V2-NoC*	*V0-NoC*	*V1-NoC*	*V2-NoC*	*V0-NoC*	*V1-NoC*	*V2-NoC*	*V0-NoC*	*V1-NoC*	*V2-NoC*
Average latency	247.5	239.3	220.1	289.6	294.9	278.3	332.5	334.2	328.7	176.5	169.8	179.6
VC number	192	192	110	192	128	100	192	112	104	192	144	120

Table 4. NoC performance when VC depth is 8.

Evaluation metrics	Uniform distribution			Hotspot in the center			Hotspot in the edge			Hotspot in the corner		
	V0-NoC	*V1-NoC*	*V2-NoC*	*V0-NoC*	*V1-NoC*	*V2-NoC*	*V0-NoC*	*V1-NoC*	*V2-NoC*	*V0-NoC*	*V1-NoC*	*V2-NoC*
Average latency	152.7	149.1	168.9	213.5	231.1	221.2	249.4	214.8	224.1	240.4	236.3	223.9
VC number	192	188	164	192	114	94	192	112	108	192	112	112

Table 5. NoC performance when VC depth is 16.

Evaluation metrics	Uniform distribution			Hotspot in the center			Hotspot in the edge			Hotspot in the corner		
	V0-NoC	*V1-NoC*	*V2-NoC*	*V0-NoC*	*V1-NoC*	*V2-NoC*	*V0-NoC*	*V1-NoC*	*V2-NoC*	*V0-NoC*	*V1-NoC*	*V2-NoC*
Average latency	224.7	229.5	252.1	239.4	233.7	257.9	343.8	310.3	334.7	277.6	252.9	274.6
VC number	192	128	132	192	96	96	192	96	96	192	100	108

Acknowledgment. This work has been supported by the Research Funds of Education Department of Heilongjiang Province, Grant No. 12531518.

References

1. Ho, R., Mai, K., Horowitz, M.: The future of wires. Proc. IEEE **89**(4), 490–504 (2001)
2. Dally, W.J., Towles, B.: Route packets, not wires: on-chip interconnection networks. In: The 38th Design Automation Conference, pp. 684–689 (2001)
3. Benini, L., De Micheli, G.: Networks on chips: a new SoC paradigm. IEEE Trans. Comput. **35**(1), 70–78 (2002)
4. Guerrier, P., Greiner, A.: A generic architecture for on-chip packet-switched interconnections. In: Design Automation and Test in Europe (DATE 2000), pp. 250–256 (2000)
5. Bjerregaard, T., Mahadevan, S.: A survey of research and practices of network-on-chip. ACM Comput. Surv. **38**(3), 1–51 (2006)
6. Kim, J., Nicopoulos, C., Park, D., et al.: A gracefully degrading and energy-efficient modular router architecture for on-chip networks. In: The 33rd International Symposium on Computer Architecture (ISCA 2006), pp. 4–15 (2006)
7. Hu, J., Marculescu, R.: Application-specific buffer space allocation for networks-on-chip router design. In: The IEEE/ACM International Conference on Computer Aided Design (ICCAD), pp. 354–361 (2004)
8. Nicopoulos, C.A., Park, D., Kim, J., et al.: VichaR: a dynamic virtual channel regulator for network-on-chip routers. In: The 39th Annual IEEE/ACM International Symposium on Microarchitecture, pp. 333–344 (2006)
9. Chen, X., Peh, L.-S.: Leakage power modeling and optimization in interconnection networks. In: The International Symposium on Low Power Electronics and Design, pp. 90–95 (2003)
10. Matsutani, H., Koibuchi, M., Wang, D., Amano, H.: Run-time power gating of on-chip routers using look-ahead routing. In: Design Automation Conference (ASPDAC), pp. 55–60 (2008)
11. Matsutani, H., Koibuchi, M., Wang, D., Amano, H.: Adding slow-slient virtual channels for low-power on-chip networks. In: The 2nd IEEE International Symposium on Networks-On-Chip, pp. 23–32 (2008)
12. Ding, J., Bhuyan, L.N.: Evaluation of multi-queue buffered multistage interconnection networks under uniform and non-uniform traffic patterns. Int. J. Syst. Sci. **28**(11), 1115–1128 (1997)
13. Ni, N., Pirvu, M., Bhuyan, L.: Circular buffered switch design with wormhole routing and virtual channels. In: Computer Design: VLSI in Computers and Processors, pp. 466–473 (1998)
14. Bolotin, E., Cidon, I., Ginosar, R., Kolodny, A.: QNoC: QoS architecture and design process for network on chip. Spec. Issue Netw. Chip J. Syst. Architect. **50**(2–3), 105–128 (2004)
15. Huang, T., Ogras, U.Y., Marculescu, R.: Virtual channels planning for networks-on-chip. In: Proceedings of the 8th International on Quality Electronic Design (ISQED), pp. 879–884 (2007)