Improving the Performance of Interconnection Networks Using DAMQ Buffer Schemes

Mohammad Ali Jabraeil Jamali† and Ahmad Khademzadeh††

† CE Department, Islamic Azad University, Science & Research Branch,Tehran, Iran.
†† Iran Telecommunication Research Center,Tehran, Iran.

Summary
Networks on chip (NoCs) are communication infrastructures that offer parallelism and scalability. In this paper we present high performance dynamically allocated multi-queue (DAMQ) buffer schemes for fault tolerance systems on chip applications that require an interconnection network. Two virtual channels shared the same buffer space. Fault tolerant mechanisms for interconnection networks are becoming a critical design issue for large massively parallel computers. It is also important to high performance SoCs as the system complexity keeps increasing rapidly. On the message switching layer, we make improvement to boost system performance when there are faults involved in the components communication. The proposed scheme is when a node or a physical channel is deemed as faulty, the previous hop node will terminate the buffer occupancy of messages destined to the failed link. The buffer usage decisions are made at switching layer without interactions with higher abstract layer, thus buffer space will be released to messages destined to other healthy nodes quickly. Therefore, the buffer space will be efficiently used in case fault occurs at some nodes.

Key words:
Network on chip, Fault tolerance, DAMQS, Buffer space, Odd-even routing algorithm

1. Introduction

Future system-on-chip (SoC) designs require predictable, scalable and reusable on-chip interconnect architecture to increase reliability and productivity. Current bus-based interconnect architectures are inherently non-scalable, less adaptable for reuse and their reliability decreases with system size. To overcome these problems, it has been proposed to build a message passing network for on-chip communication - network-on-chip.

Due to the constraints of being in a single chip, using an interconnection network on chip needs be restricted in terms of area. Thus, it is extremely important to design the schemes that require less hardware resources and still provide a good performance. Virtual channel multiplexing across a physical channel is extensively used to boost performance and avoid deadlock.

As virtual channels are not equally used in many applications, if they share a common buffer, the whole buffer space will be better utilized. In this paper we present schemes that are based on a DAMQ buffer. These schemes provide similar performance as other statically allocated multiple-queue (SAMQ) buffers using less hardware and, therefore, requiring less hardware.

In order to improve the reliability of SoCs, their interconnect infrastructures must be designed such that fabrication and life-time faults can be tolerated. These irrecoverable faults influence the behavior of NoC fabrics and consequently degrade the system performance. Therefore, achieving on-chip fault tolerant communication is becoming increasingly important in presence of such permanent faults. A fault tolerant algorithm distinguishes from deterministic one according to the fact that it can provide an alternative path so that the message wouldn't be blocked by a faulty component [1], [2]. Consequently we investigate the applicability of partially adaptive algorithms to achieve a certain degree of fault tolerance in NoC communication fabrics.

The paper is organized as follows. In section 2, we review the fault tolerant odd-even algorithm in mesh-based NoC while in section 3 high performance DAMQ buffer schemes are discussed. Experimental results, that show the performance of the proposed approaches, are presented in section 4 followed by conclusions in section 5.

2. Fault Tolerant Odd-even Algorithm in Mesh-Based NoC

Fault tolerance is the ability of the network to function in the presence of component failure. The turn model is a well known partial adaptive routing algorithm, widely investigated for multi-processor environments [3]. The odd-even turn model facilitates deadlock-free routing in mesh network of a NoC. In a two dimensional mesh of size $m \times n$ every node is identified by a two element vector (x, y), $0 \le x \le m-1$, and $0 \le y \le n-1$, where x and y are the coordinates in the two dimensions. The nodes having the same x dimension belong to the same column and those having the same y dimension constitute the same row. A row channel and a column channel refer to channels in the

Manuscript received July 5, 2009
Manuscript revised July 20, 2009

x-dimension and *y*-dimension respectively. A turn is the common point where the tail node of either the row or the column channels meet and the particular node (at which they meet) are referred to as the turning node. Deadlocks in routing usually occur as a result of packets waiting to for each other to form a cycle. Many of the routing algorithms prohibit deadlock by avoiding certain turns, whereas the odd-even routing is based on restricting the locations at which certain turns can be taken so that cyclic dependency never occurs and hence deadlock is avoided. This model allows all types of turns. The odd-even turn model is based on the following routing rules:

Rule 1: East-north and north-west turns are not allowed at any nodes located in the even column and odd column respectively.

Rule 2: East-south and south-west turns are not allowed at any nodes located in the even column and odd column respectively.

The odd and even columns defined above can also be interchanged. Deadlock freedom is achieved as the rightmost column of the waiting path cannot result, if the rules are followed. The detail of odd-even turn model is explained with the help of Fig. 1. In the figure, the odd-even routing algorithm has been illustrated. First, we have considered a case when there is no fault in the network. Then we have considered a network with faults and again show how the data can be routed in the latter case. In Fig. 1 (a), source IPa is trying to send a packet to destination IPb. Two situations are presented in Fig. 1 (a) and (b) respectively. When the network is fault-free, the packet is first routed in x direction before being routed to y direction. When one link L1 is faulty in the path, if normal deterministic x-y routing [4] was adopted, there is no path for the packet to move towards the destination. For odd-even turn algorithm, the packet is routed one hop perpendicular to the top, then toward the destination. As shown in Fig. 1 (b), suppose source IPc is trying to communicate to the destination IPd, and the link L2 is faulty. In this case the packet is routed, one hop toward the down, then toward the destination.

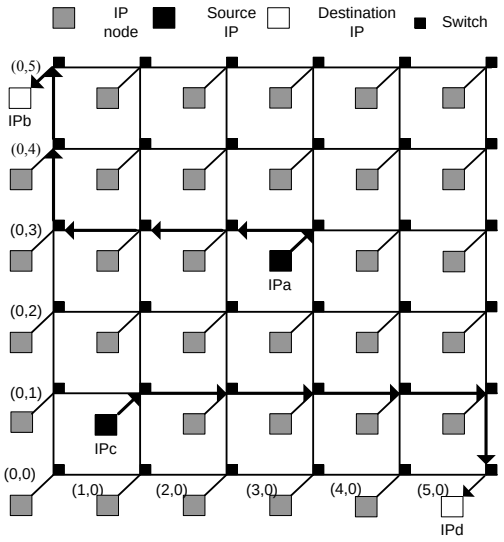
3. High Performance DAMQ Buffer Schemes

In this section DAMQ buffer schemes based on self compacting buffer (SCB) are presented. These schemes are proposed to let traffic flow make an efficient use of input buffer that resides in each communicating node.

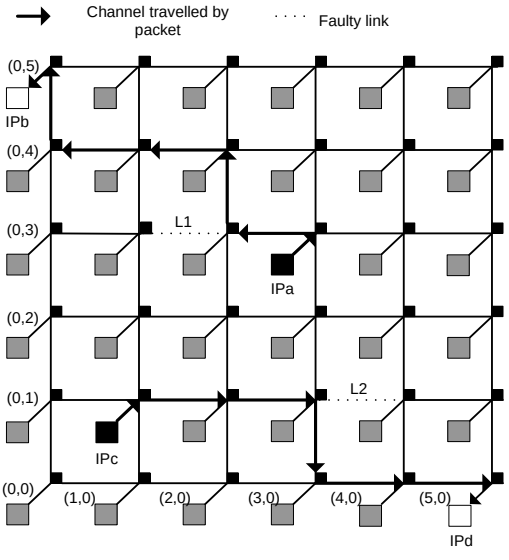
3.1 Linked List Buffer Scheme

In order to let multiple queues of packets share a DAMQ buffer, linked lists can be used to implement the buffer scheme [1] [5] [6].

The basic idea of this approach is to maintain (*k*+1) linked lists in each buffer: one list of packets for each one of the (*k*-1) output ports, one list of packets for the end node interface and one list of free buffer blocks. Corresponding to each linked list there is a head register and a tail register. The head register points to the first block in the queue and the tail register points to the last block. In each output queue, next block information also must be stored in each buffer block to maintain the FIFO ordering.



(a) fault free case



(b) in presents of faults

Fig. 1. A sample for Odd-even turn model algorithm.

3.2 Self-Compacting Buffer Scheme

To reduce the hardware complexity of the linked list scheme, an efficient DAMQ buffer design self-compacting buffer was proposed by [4]. The idea for this buffer scheme is to divide the buffer dynamically into regions with every region containing the data associated with a single output channel. The self compacting buffer scheme has the following properties:

Property 1: If two channels are denoted as i, j with $i < j$, then the dynamically allocated region for channel i and j always resides in a space addressed by addresses Ai and Aj respectively where $Ai < Aj$.

Property 2: There is no reserved space dedicated for any channel. If no data are currently requiring the output channel i, then there is no region reserved for channel i.

Property 3: Within the space for each channel, the data are stored in a FIFO manner.

Property 4: For every output channel i, there is an integer number, δ_i, denoting the number of entries present in the region reserved for output channel i [4].

When an insertion of the packet requires space in the middle of the buffer, the required space will be created by moving down all the data which reside below the insertion address. Similarly, when a reading operation conducted from the top of a region, data removed from the buffer may result in empty space in the middle of the buffer, then the data below the read address is shifted up to fill the empty space.

3.3 DAMQ and DAMQ Shared (DAMQS) Buffer Schemes

DAMQ dynamically allocate buffer blocks according to the packet received. Compared with statically allocated buffer scheme, the advantage of DAMQ is that it uses efficiently the buffer space by applying free space to any incoming packet regardless its destination output port. Since there is no reserved space dedicated for each output channel, the packets destined to one specific output port may occupy the whole buffer space thus the packets destined to other output ports have no chance to get into the buffer. This is the case especially for small and compact routers with limited buffer space where wormhole switching technique and virtual channel mechanism are commonly used. A unidirectional virtual channel is implemented by an independently managed pair of buffers at two adjacent nodes. When several virtual channels multiplex across the physical channel and share a common buffer, the virtual channels which have packets accepted in the buffer prior to other virtual channels may hold the whole buffer space when the output port to next hop node that it destines to is busy. In order to overcome this shortcoming a new buffer scheme, DAMQ with reserved space for all virtual channels (DAMQA) was proposed in [6]. DAMQA is based on self-compacting buffer scheme. The virtual channels belonging to on direction of a bidirectional physical channel shares a buffer.

As shown in Fig. 2, two buffer slots are reserved for each virtual channel before the buffer accepts any incoming flit and during buffers operation. However, in a wormhole-switched network with several virtual channels multiplexing a physical channel, some routing algorithms, for example, the algorithms that pick an available virtual channel sequentially tend to choose one set of virtual channels over others; moreover, even the virtual channels are chosen randomly, the traffic may not evenly distributed into all virtual channels of different physical channel, thus the traffic load usually is not evenly distributed in buffer space of a physical channel and among different physical channels. Therefore, a more efficient approach to use the available buffer space is to let the virtual channels belonging to a physical channel share buffer with virtual channels of another physical channel.

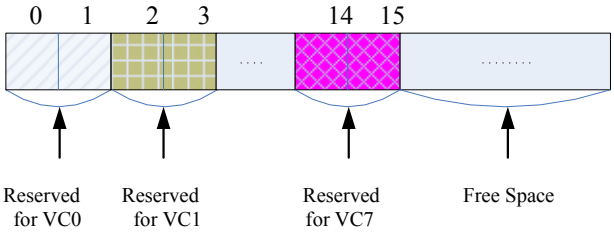


Fig. 2 DAMQA buffer space at the initial state.

As shown in Fig. 3 (a), the simple switch with four input and four output ports adopts DAMQA buffer scheme for the input buffer; there is one dedicated buffer per physical channel, i.e. east X, west X, north Y and south Y. Each physical channel buffer has its own read port and write port, the four virtual channels that are multiplexing a physical channel have their own reserved space (RS) in buffer [7].

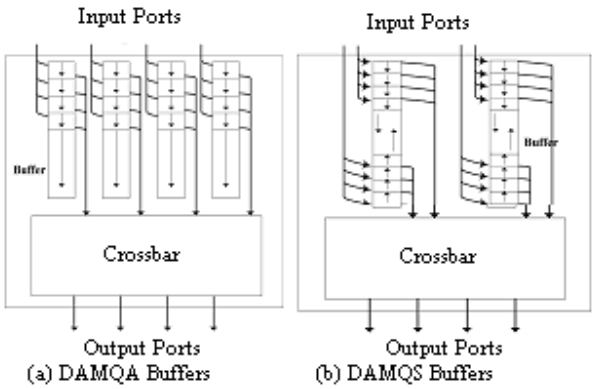


Fig. 3. Switches with DAMQA buffer and DAMQS buffer.

DAMQS buffer combines the buffer for virtual channels from two different physical channels. We combine the

buffer space for east X and south Y virtual channels to build one physical buffer, and west X and north Y forms another buffer group. As shown in Fig. 3 (b), there are two buffers for four physical channels; each buffer is shared by eight virtual channels, and has two read ports and write ports respectively. In this way, DAMQS, the shared space is placed in the middle of the two buffer regions of two virtual link groups then the two buffer regions expand towards center of the free buffer space. This way, there will be less data shifts when a flit is saved into buffer because the movement of one region doesn't depend on shift of another group.

As shown in Fig. 4 (a), two buffer slots are reserved for each virtual channel before any flit comes in the buffer. Virtual channels on X dimension start to occupy the buffer from lower end of the whole buffer space while virtual channels on Y dimension start using buffer on another end. The buffer space of two groups expands and compresses in opposite direction. When a buffer group accepts a flit it expands, when it dispatches a flit it compresses. One register is used to point to the head of each reserved space, i.e. the head of the buffer region for each virtual channel. If two channels are denoted as *Vi*, *Vj* with $i + 1 = j$, then the reserved region for *Vj* will be placed right after the reserved region for *Vi*.

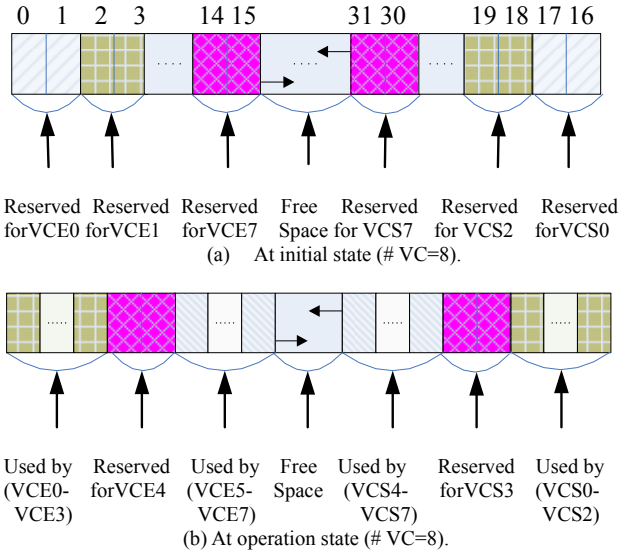


Fig. 4. DAMQS buffer space status. VCE (VC East), VCS (VC South)

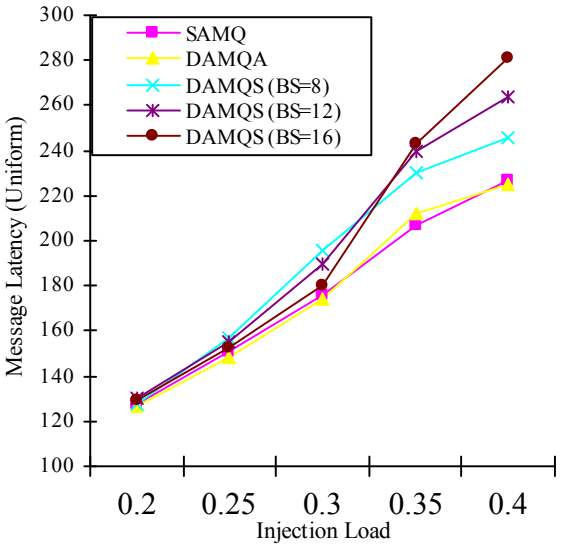
Size of RS for each virtual channel can be adjusted. We choose two flits because two reserved flits scheme yields satisfactory performance while keeping more free space for sharing in our simulation experiments. The RS is always kept if there is no flit or only one flit in the buffer region for a specific virtual channel [5] [6]. As shown in Fig. 4 (b), when the buffer performs shift up or shift down operations, the RSs are also shifted. When a virtual channel accepts a flit, it first uses its RS. If RS is used up, buffer space of the whole group expands toward another group's buffer space to produce a slot. Once the boundaries of the two buffer groups encounter, no more flit will be accepted unless this flit goes to a virtual channel which has RS available. At any time, the number of current flits in buffer plus the number of reserved slots equals to the total amount of buffer slots. Therefore, one or more virtual channels which have the flits come into the buffer at earlier time can never deprive the chance for other virtual channels which get flits later than them to get buffer. Also, in case the earlier coming packets are blocked in the buffer, since there is still reserved space for other virtual channels, the network traffic will keep flowing; therefore the performance of the switch is enhanced. Moreover, as virtual channels from two physical channels are sharing the buffer, the buffer space is more efficiently used by the incoming flits. Hence, to achieve same network performance, by using DAMQS scheme, a switch can use less buffer space than DAMQA and traditional buffer schemes. The results will be shown in next section.

4. Experimental Results

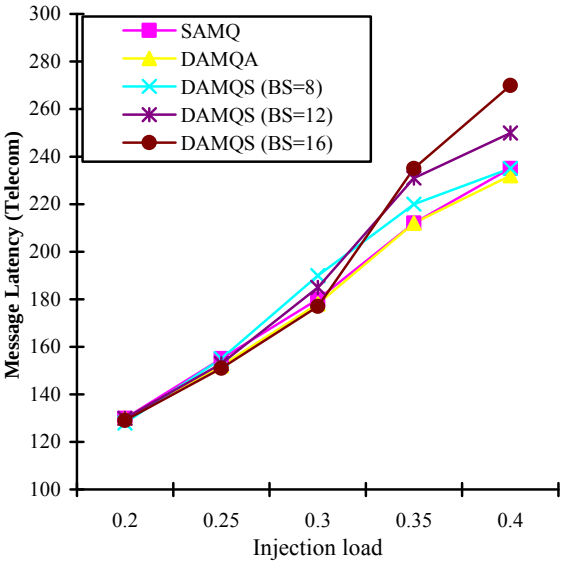
In this section, we consider a system consisting of 64 IP blocks mapped onto mesh-based NoC architecture as shown in Fig. 1. Packets size is set to 32 flits. Virtual channels number for each physical channel is 4. Faults are generated randomly in the network. We set the buffer size (BS) for each virtual channel to 4 flits when DAMQA and SAMQ are used. Since four virtual channels are multiplexing cross one physical channel, the buffer size for each direction of a duplex physical channel is 16 flits when these two buffer schemes are evaluated. To examine the performance of DAMQS with regard to the relationship between buffer size and network performance, we use flits buffer with different sizes. Since the network performance is greatly influenced by the traffic pattern, we applied two different traffic patterns, including synthetic traffic pattern (uniform) and a real-life traffic pattern (telecom) is retrieved from "E3S" benchmark suite [8], which contain 30 tasks.

Fig. 5 (a, b) shows the message latency as function of injection load with 2% fault rate (fr) for two uniform and telecom traffic patterns, respectively. When we further increase the traffic load and the fault rate after the network starts getting saturated, DAMQS shows higher latency than other schemes. The reason is DAMQS holds much more flits in the buffer than other schemes.

Fig. 6 (a, b) shows the throughput characteristics of the NoC for each buffer scheme in presence of permanent faults with different fault rates for two uniform and telecom traffic patterns, respectively.

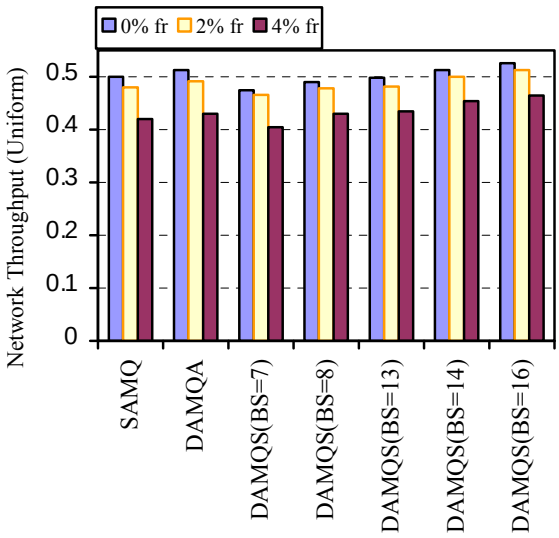


(a) Uniform traffic

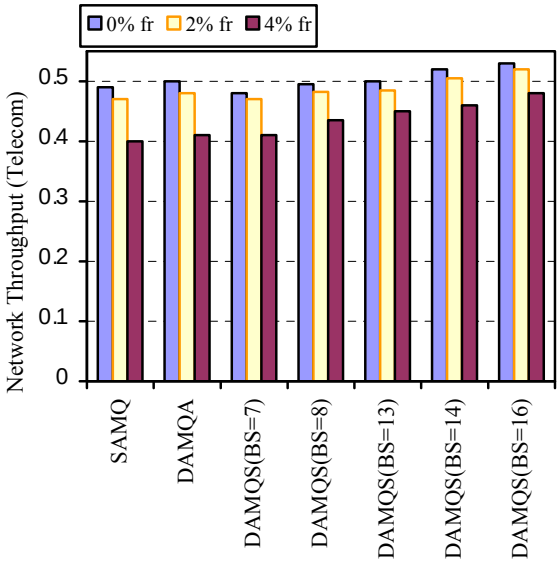


(b) Telecom traffic
Fig. 5. Message latency (2% fr).

As shown in Fig. 6 DAMQS can provide a better performance in presence of faults than other schemes. For example in uniform traffic, 14-flit DAMQS with 0% fr and 8-flit DAMQS with 4% fr achieves approximately the same maximum throughput as a 16-flit DAMQA as shown in Fig. 6 (a). This is to say, to provide a similar network performance on very limited buffer resource, DAMQA with 0% fr achieves similar throughput with 12.5% more buffer space than DAMQS with 0% fr, and DAMQA with 4% fr achieves similar throughput with 50% more buffer space than DAMQS with 4% fr. For telecom traffic, 13-flit DAMQS with 0% fr and 7-flit DAMQS with 4% fr achieves approximately the same maximum throughput as a 16-flit DAMQA as shown in Fig. 6 (b). This result shows that, DAMQA with 0% fr achieves similar throughput with 18.75% more buffer space than DAMQS with 0% fr, and DAMQA with 4% fr achieves similar throughput with 56.25% more buffer space than DAMQS with 4% fr.



(a) Uniform traffic



(b) Telecom traffic
Fig. 6. Network throughput.

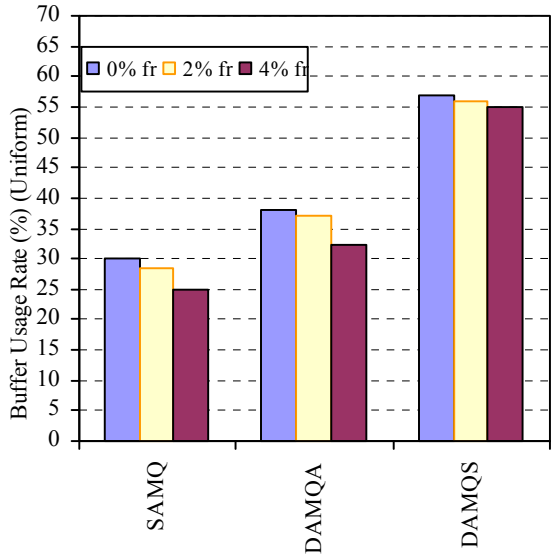
The total buffer space (BUF_{total}) can be obtained by the following formula:

$$BUF_{total} = (N \times PHY - 4\sqrt{N}) \times VC \times VB$$

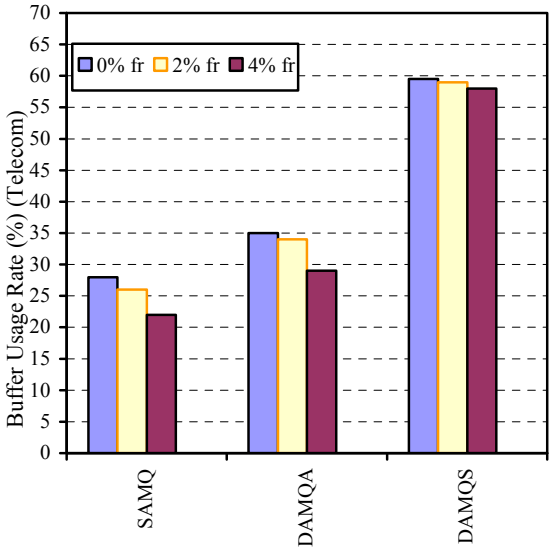
where PHY is the physical channel corresponding to a node port, N is the count of nodes, VC is the count of virtual channel multiplexing a physical channel and VB is

the buffer size of a virtual channel. The total available buffer space is 3584 flits in our simulations. We considered high injection load of 0.35. As shown in Fig. 7 DAMQS can provide a better performance than DAMQA and SAMQ.

Comparing Fig. 5 (a), Fig. 6 (a) and Fig. 7 (a) with Fig. 5 (b), Fig. 6 (b) and Fig. 7 (b)**,** respectively it is clear that the DAMQS is even more beneficial for real benchmarks. Indeed, compared to the uniform traffic pattern, the traffic patterns for real benchmarks are much more unbalanced (for instance, some of the channels have even a load of zero), which makes the idea of share buffer more attractive.



(a) Uniform traffic



(b) Telecom traffic

Fig. 7. Buffer usage rate.

5. Conclusion

For deep sub-micron VLSI processes, the life-time reliability of devices is likely to be compromised by effects such as electromigration and material ageing. Consequently, the performance of NoC interconnect architectures will be severely affected due to presence of permanent faults. Though deterministic routing is very easy to implement, it fails to sustain the desired level of performance in presence of permanent faults. When an adaptive routing protocol such as odd-even turn algorithm is used for the NoC, DAMQS is an excellent scheme to optimize buffer management providing a good throughput when the network has a larger load in presence of faults. It can utilize significantly less buffer space with further increase of the fault rate without sacrificing the network performance.

References

[1] T. Schonwald, J. Zimmermann, O. Bringmann, W. Rosenstiel "Fully Adaptive Fault-tolerant Routing Algorithm for Network-on-Chip Architectures", 10th Euromicro Conference on Digital System Design Architectures, (DSD 2007).

[2] J. Zhou, F. C. M. Lau, "Adaptive Fault-tolerant Wormhole Routing with Two Virtual Channels in 2D Meshes", 7th International Symposium on Parallel Architectures, Algorithms and Networks, 2004. May 2004. pp. 142 - 148.

[3] C.J. Glass and L.M. Ni, "Adaptive Routing in Mesh-connected Networks", Proceedings of the 12th International Conference on Distributed Computing Systems, June 1992. pp: 12-19.

[4] C.J. Glass and L.M. Ni, "The Turn Model for Adaptive Routing", Proceedings of the 19th Annual International Symposium on Computer Architecture, May 1992. pp: 278-287.

[5] J. Liu, J. G. Delgado-Frias, "DAMQ Self-Compacting Buffer Schemes for Systems with Network-On-Chip," Int. Conf. Computer Design, Las Vegas, June 2005, 97-103.

[6] J. Liu, J. G. Delgado-Frias, "A Shared Self-Compacting Buffer for Network-On-Chip Systems," 49th IEEE Int. Midwest Symposium on Circuits and Systems. August 2006.

[7] Y. Tamir and G. L. Frazier, "Dynamically-allocated multiqueue buffers for VLSI communication switches," IEEE Transactions on Computers, vol. 41, no. 2, June 1992, 725–737.

[8] R. Dick. Embedded system synthesis benchmarks suites (E3S), http://www.ece.northwestern.edu/~dickrp/e3s.

Mohammad Ali Jabraeil Jamali received his B.Sc. in Electrical Engineering and M.Sc. in Computer Hardware Engineering from Oromie University in 1994 and Tehran Science and Research Branch of Islamic Azad University in 2003, respectively. Currently, he is a Ph.D. student and Faculty Member at the Tehran Science and Research Branch of Islamic Azad University under super- vision of Dr. Ahmad khademzadeh and Islamic Azad University, Shabestar Branch, respectively. His research interests include VLSI Design, Interconnection Network, Computer Network and Computer Architectures.received the B.E. and M.E. degrees, from Fukuyama Univ. in 1979 and 1981, respectively.

Ahmad khademzadeh is currently associate professor in Iran Telecommunication Research Center. His research interests include VLSI Design, Interconnection Network, Fault Tolerant and Computer Architectures.