International Journal of Electrical, Electronic and Communication Sciences
ISSN: 2517-9438
Vol:3, No:6, 2009

DAMQ-Based Approach for Efficiently Using the Buffer Spaces of a NoC Router

Mohammad Ali Jabraeil Jamali, Ahmad khademzadeh

***Abstract*—**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.

***Keywords*—**DAMQ, NoC, fault tolerant, odd-even routing algorithm, buffer space.

I. 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 (NoC).

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. Each virtual channel is realized by a pair of buffers located on adjacent communicating nodes.

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 a scheme that is based on a DAMQ buffer. This scheme provides similar performance as other statically allocated multiple-queue 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.

In the NoC environment initially deterministic routing algorithms were employed due to the ease of implementation. The primary limitation of deterministic routing is that it establishes a fixed path between a pair of source and destination nodes. Consequently it demonstrates poor performance in presence of faults situated on the routing path as it fails to establish alternate routes. Adaptive routing protocols are proposed to make more efficient use of bandwidth and to improve fault tolerance of interconnection network. In order to achieve this, adaptive routing protocols provide alternative paths for communicating nodes. Thus it can overcome the congested areas in the network. Several adaptive routing algorithms have been proposed, showing that message blocking can be considerably reduced, thus strongly improving throughput [1]. The turn model is a well known partial adaptive routing algorithm, widely investigated for multi-processor environments [1] [2]. West-first routing, north-last routing and negative-first routing are three basic types of turn models. Compared to fully adaptive routing algorithms, turn model algorithm is a partially adaptive algorithm because two turns out of eight are forbidden in order to avoid deadlock. The odd-even turn model [3] prohibits some types of turns based on the locations of the nodes in order to make itself deadlock free.

The paper is organized as follows. In section II, we review the related work while in section III, the DAMQS buffer scheme is discussed. Experimental results, that show the performance of the proposed approach, are presented in Section IV followed by conclusions in Section V.

Mohammad Ali Jabraeil Jamali is with Islamic Azad University, Shabestar Branch, Iran (e-mail: m_jamali@itrc.ac.ir).

Ahmad khademzadeh is with Iran Telecommunication Research Center,Tehran, Iran (e-mail: zadeh@itrc.ac.ir).

II. RELATED WORK

The common characteristic of NoC architectures is that the constituent IP cores communicate with each other through switches. Generally wormhole routing is adopted [4]. We analyze the performance of a Mesh-based NoC in presence of permanent faults when different buffer schemes are adopted.

A. Odd-even Turn Model Algorithm

The Odd-even turn model was first proposed by Chiu [5] and is an extension of Glass and Ni's turn model [1]. The odd-even turn model facilitates deadlock-free routing in Mesh network of a NoC [6] [7]. 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 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.

Definition 1: In a 2-Dimensional Mesh network, a particular column is called an even (or odd) column if the x-coordinate of the column is even (or odd).

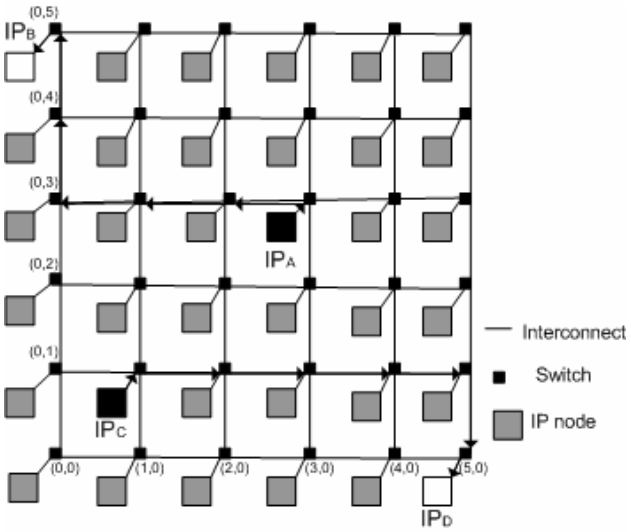
The odd-even turn model is based on the following routing rules:

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

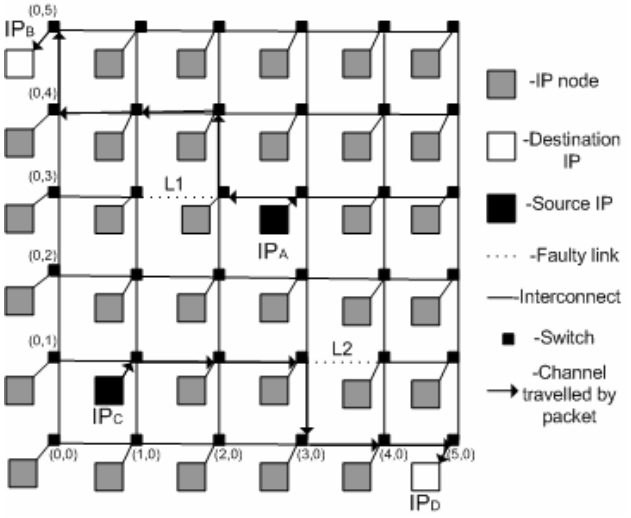
Rule 2: *East-south* and *south-west* turns are not allowed at any nodes located in the even column and odd column respectively.

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 Figure 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 Figure 1. (a), source IP_A is trying to send a packet to destination IP_B. Two situations are presented in Figure 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 [1] 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 Figure 1. (b), suppose source IP_C is trying to communicate to the destination IP_D, and the link L2 is faulty. In this case the packet is routed, one hop toward the down, then toward the destination.



(a) fault free case.



(b) in presents of faults.

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

B. 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.

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 [8] [10]. 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.

2) Self-compacting buffer scheme.

To reduce the hardware complexity of the linked list scheme, an efficient DAMQ buffer design self-compacting buffer (SCB) was proposed by [10]. 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. If two channels are denoted as i, j with $i < j$, then the addresses of buffer regions for the two channels Ai Aj will be $Ai < Aj$. There is no reserved space dedicated for any channel. Data is stored in a FIFO manner within the region for each channel. 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.

III. DAMQ BUFFER SCHEME

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 [11]. DAMQA is based on Self-compacting buffer (SCB) scheme. The virtual channels belonging to on direction of a bidirectional physical channel share a buffer.

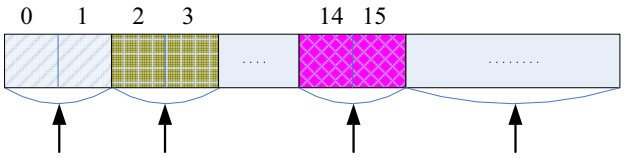


Fig. 2. DAMQA Buffer space at the initial state

As shown in Figure 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.

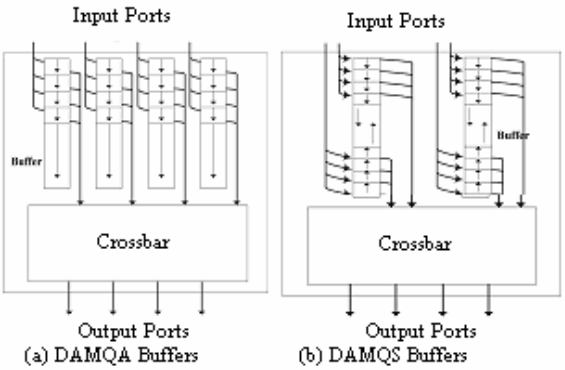
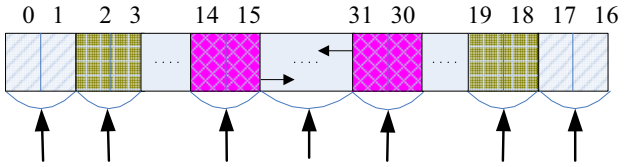


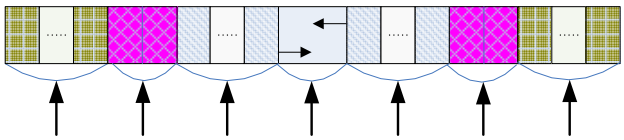
Fig. 3. Switches with DAMQA buffer and DAMQS buffer

As shown in Figure 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. DAMQ Shared (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 Figure 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 Figure 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*. 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 [11].



(a) At initial state (# VC=8).



(b) At operation state (# VC=8).

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

As shown in Figure 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.

IV. EXPERIMENTAL RESULTS

In this section, we evaluate the performance of the buffer schemes discussed above with Odd-even turn model routing algorithm when applied to a Mesh-based NoC. We characterize the performance of the NoC under consideration in terms of network throughput, latency and buffer usage in presence of permanent faults.

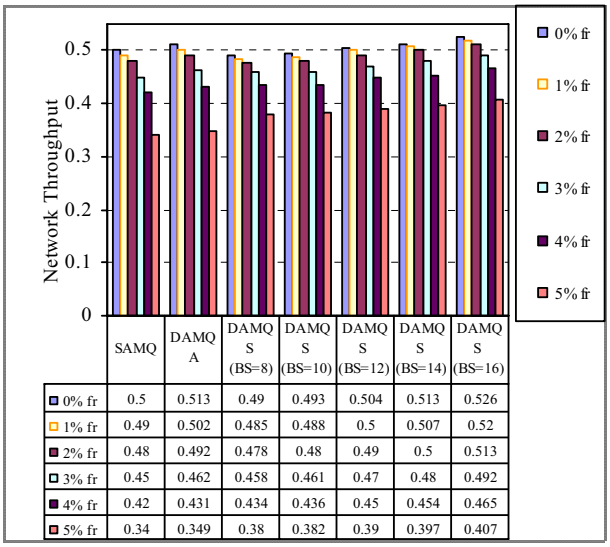
To study the system performance characteristics mentioned above, we consider a system consisting of 64 IP blocks mapped onto Mesh-based NoC architecture as shown in Figure 1. Messages were injected with a uniform traffic pattern (in each cycle, all IP cores can generate messages with the same probability). Faults are generated randomly in the network. Faults are manifested by making a particular inter-switch link or node unavailable permanently for data routing.

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 five different size 8, 10, 12, 14 and16 flits buffer.

We first studied the variation of network throughput and message latency as function of injection load [4] in presence of fault considering the SAMQ, DAMQA and DAMQS buffer schemes. The throughput and message latency are shown in Table I and Figure 5, respectively. It is well known that there is no significant difference for different buffer scheme while the network is not saturated. In our simulation experiments, we keep increasing the traffic load so that we can compare the performance of different buffer schemes when the network is in saturation status. We compare the performance of different buffer schemes when the network starts to saturate on about 0.2 traffic load until it gets saturated after about 0.4 traffic

load is applied.

Figure 5 shows the throughput characteristics of the NoC for each buffer scheme in presence of permanent faults, assuming 0.35 injection load.



	SAMQ	DAMQA	DAMQS (BS=8)	DAMQS (BS=10)	DAMQS (BS=12)	DAMQS (BS=14)	DAMQS (BS=16)
0% fr	0.5	0.513	0.49	0.493	0.504	0.513	0.526
1% fr	0.49	0.502	0.485	0.488	0.5	0.507	0.52
2% fr	0.48	0.492	0.478	0.48	0.49	0.5	0.513
3% fr	0.45	0.462	0.458	0.461	0.47	0.48	0.492
4% fr	0.42	0.431	0.434	0.436	0.45	0.454	0.465
5% fr	0.34	0.349	0.38	0.382	0.39	0.397	0.407

Fig. 5. Comparison on throughput between 8-16 flit buffer DAMQS, 16 flit buffer DAMQA and SAMQ under uniform traffic with different fault rates.

As shown in Figure 5 DAMQS can provide a higher throughput in presence of faults than SAMQ and DAMQA when they all use same size of 16 flit buffers. When the network gets saturated, DAMQS with a 0% fault rate achieves the highest throughput with same size 16-flit buffer is used.

Table II shows the number of flits buffer DAMQS with increasing fault rate as SAMQ and DAMQA using 16 flits buffer, assuming the same maximum throughput. For example, 8-flit DAMQS with a 2% fault rate achieves approximately the same maximum throughput as SAMQ using 16 flits buffer as shown in Figure 5. Also, 10-flit DAMQS with a 2% fault rate achieves approximately the same maximum throughput as SAMQ using 16 flits buffer. In sum, DAMQS achieves best performance among the three buffer schemes we tested, specially in presence of permanent faults.

DAMQS tends to provide a more efficient method for flits to share buffer space than DAMQA which has already shown advantages over traditional SAMQ scheme. DAMQS achieves the best performance among the three buffer schemes we tested. 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. DAMQS in opposition to DAMQA and SAMQ allows use of buffers related to the failed links. Thus buffer space will be released to messages destined to other healthy nodes quickly. Therefore, in DAMQS the buffer space will be efficiently used in case fault occurs at some links and nodes.

TABLE II
THE NUMBER OF FLITS BUFFER DAMQS WITH INCREASING FAULT RATE AS SAMQ AND DAMQA USING 16 FLITS BUFFER, ASSUMING THE SAME MAXIMUM THROUGHPUT

Buffer type	SAMQ(BS per PC=16)						DAMQA(BS per PC=16)					
Fault rate (%)	0	1	2	3	4	5	0	1	2	3	4	5
DAMQS (BS per PC)	12	10	8	7	6	5	14	12	10	9	8	7

As to the message latency, DAMQS managed to hold a similar latency as SAMQ until the network is severely saturated after a load of about 0.35 is applied. When we further increase the traffic load and the fault rate after the network starts getting saturated, DAMQS with 5% fault rate shows higher latency than both DAMQA and SAMQ. The reason is DAMQS with 5% fault rate holds much more flits in the buffer than other schemes. The numbers of average flits that are stored in the buffer are presented in Table III.

The total buffer space (*BUFtotal*) 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, *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 also compared the performance of the DAMQS with DAMQA and SAMQ in terms of the buffer usage rate as a relevant metric as suggested in [11]. We considered high injection load of 0.35. As shown in Figure 6, DAMQS can provide a better performance in presence of faults than DAMQA and SAMQ. It has very high buffer usage rate in presence of faults when we further increase the fault rate.

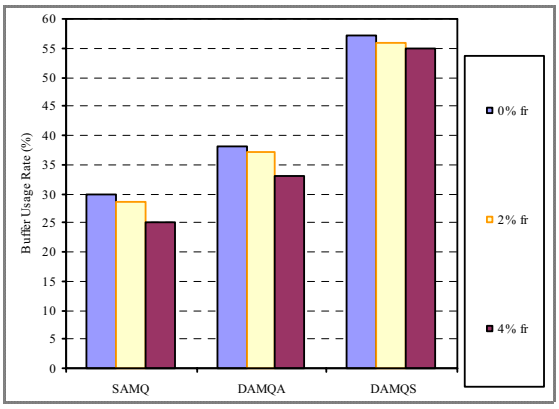


Fig. 6. Comparision on Buffer Usage Rate between 16 flit buffer DAMQS, DAMQA, and SAMQ under uniform traffic in presence of faults.

DAMQS with 0% fault rate utilizes 57% of the whole buffer space while DAMQA and SAMQ with 0% fault rate uses 38% and 30% respectively as shown in Figure 6. Also, a DAMQS with 4% fault rate utilizes 55% of the whole buffer space while DAMQA and SAMQ with 4% fault rate uses 33% and 25% respectively as shown in Figure 6. Because message latency is a time span average on all the flits that are injected into the network, more flits stored in the buffer results in longer time for them in the waiting queues as the network has become drastically saturated.

It has been shown that DAMQS provides a better use of the buffer space. In addition, it should be pointed out that a 12-flit DAMQS buffer with 0% fault rate and 6-flit DAMQS buffer with 4% fault rate achieves approximately the same maximum throughput as a 16-flit SAMQ buffer as shown in Figure 6. Also, a 14-flit DAMQS buffer with 0% fault rate and 8-flit DAMQS buffer with 4% fault rate achieves approximately the same maximum throughput as 16-flit DAMQA buffer. This is to say, to provide a similar network performance on very limited buffer resource, DAMQS with 0% fault rate achieves similar throughput with 25% and 12.5% less buffer space than SAMQ with 0% fault rate and DAMQA with 0% fault rate, respectively and DAMQS with 4% fault rate achieves similar throughput with 62.5% and 50% less buffer space than SAMQ with 4% fault rate and DAMQA with 4% fault rate, respectively.

V. CONCLOSION

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 the fault rate without sacrificing the network performance.

REFERENCES

[1] 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.

[2] 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.

[3] Ge-Ming Chiu, "The odd-even turn model for adaptive routing", IEEE Transactions on Parallel and Distributed Systems, Volume 11, Issue 7, July 2000. pp: 729 – 738.

[4] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, R. Saleh, "Performance Evaluation and Design Trade-offs for Network on Chip Interconnect Architectures", *IEEE Transactions on Computers,* vol. 54, no. 8, August 2005. pp; 1025-1040.

[5] Ge-Ming Chiu, "The Odd-even Turn Model for Adaptive Routing", IEEE Transactions on Parallel and Distributed Systems, Volume 11, Issue 7, July 2000. pp. 729 – 738.

[6] Jie Wu, "A Fault-tolerant and Deadlock-free Routing Protocol in 2D Meshes Based on Odd-even Turn Model", IEEE Transactions on Computers, Volume 52, Issue 9, Sept. 2003. pp. 1154 - 1169.

[7] Jie Wu, Dajin Wang, "Fault-tolerant and Deadlock-free Routing in 2-D Meshes Using Rectilinear-monotone Polygonal Fault Blocks", International Conference on Parallel Processing, 18-21 Aug. 2002. pp. 247 - 254.

[8] 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.

[9] R. Sivaram, C. B. Stunkel, and D. K. Panda, "HIPIQS: A High Performance switch architecture using input queueing," *IPPS/SPDP '98*, Orlando, FL, March 1998, 134–143.

[10] J. Park, B. O'Krafka, S. Vassiliadis, and J. Delgado-Frias, "Design and evaluation of a DAMQ multiprocessor network with self-compacting buffers," *IEEE Supercomputing '94, Conference on High Performance Computing and Communications*, Nov. 14-18, 1994, 713–722.

[11] 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.

TABLE I
MESSAGE LATENCY WHERE 4 VIRTUAL CHANNELS PER PHYSICAL CHANNEL USED WHEN UNIFORM TRAFFIC IS APPLIED

Buffer Scheme	BS per PC	Load Injection Rate														
		0.20			0.25			0.30			0.35			0.40		
		0 % fr	2 % fr	4 % fr	0 % fr	2 % fr	4 % fr	0 % fr	2 % fr	4 % fr	0 % fr	2 % fr	4 % fr	0 % fr	2 % fr	4 % fr
SAMQ	16	123	128	143	145	151	162	169	176	196	199	207	231	218	227	253
DAMQA	16	122	127	142	142	148	160	171	174	194	204	212	236	217	225	252
DAMQS	8	123	128	143	150	157	177	185	196	225	215	230	271	230	246	290
	10	123	128	143	148	154	175	183	193	223	218	234	275	239	256	301
	12	124	130	144	148	155	176	179	190	221	222	240	280	247	264	311
	14	123	129	144	146	153	175	177	189	220	225	242	282	257	275	324
	16	123	129	145	145	153	175	170	180	207	225	243	284	263	281	331

TABLE III
BUFFER USAGE WHERE 4 VIRTUAL CHANNELS PER PHYSICAL CHANNEL USED WHEN UNIFORM TRAFFIC IS APPLIED

Buffer Type	BS per PC	Load Injection Rate														
		0.20			0.25			0.30			0.35			0.40		
		0 % fr	2 % fr	4 % fr	0 % fr	2 % fr	4 % fr	0 % fr	2 % fr	4 % fr	0 % fr	2 % fr	4 % fr	0 % fr	2 % fr	4% fr
SAMQ	16	316	303	266	533	512	448	870	810	732	1074	1031	902	1150	1104	966
DAMQA	16	328	315	262	583	560	491	1020	979	857	1378	1323	1157	1753	1683	1473
DAMQS	8	395	383	348	595	577	524	923	904	849	1050	1040	1008	1070	1027	898
	10	405	393	356	628	609	553	1005	985	925	1242	1230	1192	1170	1158	1123
	12	412	402	371	662	645	596	1070	1054	995	1431	1420	1388	1450	1435	1392
	14	460	448	414	700	683	630	1310	1290	1218	1643	1627	1594	1932	1917	1884
	16	460	451	419	725	710	660	1390	1373	1307	2029	2013	1998	2419	2402	2380