Impact of the Head-of-Line Blocking on Parallel Computer Networks: Hardware to Applications[*]

V. Puente[1], J.A. Gregorio[1], C. Izu[2], and R. Beivide[1]

[1] Universidad de Cantabria
39005 Santander, Spain
{vpuente,jagm,mon}@atc.unican.es
[2] University of Adelaide
SA 5005, Australia
cruz@cs.adelaide.edu.au

Abstract. A fully adaptive router with hybrid buffers at the input and output channels was designed, which improves the throughput of its input buffer counterpart by up to 40% and has only 10% higher base latency. An in-depth analysis of different router buffer organization was carried out for a toroidal network, which uses either a deterministic (DOR) or a fully adaptive routing scheme. Each proposal is described in VHDL and evaluated with the *Synopsys* synthesis tool. Technological restrictions obtained were used to evaluate network performance under both synthetic loads and real applications.

1 Introduction

Most routers select a simple buffer organization as a strategy to limit their complexity. Buffer space is attached to each input channel, and a FIFO access policy is applied to route the incoming messages. Thus, it requires a memory with only one reading and one writing port. Notwithstanding, it is well known that this buffer organization can effectively reduce peak throughput due to the so called *head-of-line blocking*. Only the first packet of each FIFO competes for the output resources; thus, when one packet blocks due to network contention, all the remaining packets at its FIFO will also block, even when their possible outputs are available. In particular, a network switch with fixed length packets and a random distribution of their destinations could only achieve about 60% of its link capacity [4].

There are two possible solutions to this problem. The first one consists of applying flexible access policies to the input buffers. From the architectural point of view, this approach presents improved performance when compared with FIFO. However, any architectural gains are eliminated on its physical implementation because of the complexity of the buffer organization. A second solution is to allocate the router buffer space into a central queue or attach the buffers to each

[*] This work is supported in part by TIC98-1162-C02-01

P. Amestoy et al. (Eds.): Euro-Par'99, LNCS 1685, pp. 1222–1230, 1999.

output. Both approaches require multiport structures which increment their silicon area. However, with current technology, the multiport option is not only a possible alternative, but, as this study will show, the small latency penalty of this solution is outweighed by its significant throughput gains.

This paper presents a comparative analysis of the different buffer organizations for either deterministic or adaptive toroidal routers. All logical combinations are explored in order to find their optimal buffer organization. The goal is to quantify the cost, in terms of latency and area, introduced by buffer proposals oriented to increase network throughput. Our evaluation ranges from the hardware design of each proposal up to the analysis of parallel application execution over cc-NUMA architectures with any of the proposed interconnection networks. Moreover, the network is evaluated under various types of synthetic traffic. This down-to-up methodology brings every factor of network design into consideration.

The rest of this paper is organized as follows. Section 2 presents the router architectures that have been considered as part of this analysis. Section 3 introduces the more significant details of their hardware implementation and Section 4 compares their performance at different levels. Finally, Section 5 collates the contributions of this paper.

2 Architectural Proposals for Torus Networks

This section introduces the deterministic and adaptive routing schemes. Then, it discusses how the different buffer organizations are applied to both routers. In all cases, the flow control is virtual cut-through (VCT). All routers use the Bubble mechanism [1] in order to avoid deadlock.

2.1 Routers with Head-of-Line Blocking

As our proposals focus on the reduction of head-of-line blocking (HLB), it is logical to introduce first our baseline routers, whose HLB motivated this study.

As mentioned before, this problem occurs in routers with the most basic buffer organization: input buffers and FIFO management policies. In fact, due to its simplicity, this type of architecture is widely used, including the network routers of commercial machines as described in [10].

The adaptive router for a *k-ary 2-cube* network, using the bubble mechanism to avoid deadlock. There are two virtual channels, attached to each physical input: a deterministic channel and an adaptive channel [3]. Messages progress in order of dimension through deterministic channels, and in any minimal route through adaptive channels. Buffering at both input channels applies a FIFO policy.

Each physical input channel has a synchronization unit. This is because the router design is synchronous but the communications between neighboring nodes are *self-timed*, in order to avoid any problems with the clock skew.

The basic architecture of the deterministic router is quite similar to the adaptive one. The main difference is that it only needs one (deterministic) channel per physical line. For a more exhaustive description of both routers, please refer to [8] and [2].

2.2 Routers without Head-of-Line Blocking

Deterministic Router The strategy followed in this design is simple: move the buffer space to the output ports. The output buffer has only one read port, but this won't cause unnecessary blocking because all packets stored in a given buffer are requesting the same output resource. However, multiple input ports may want to write simultaneously into the same output buffer. Thus, we should provide multiport memories. With DOR, we require 2 writing ports at the X dimension (x input port or injection port) and 4 writing ports at both the Y dimension (the y input port plus any x port or injection port) and consumption.

One of the problems with output buffering is the need to route the packet before applying the flow control to the selected output buffer. By providing a small buffer per input port, we can decouple the two processes, and reduce the internal node delay. This buffer cannot introduce HLB, because it has capacity for a single packet.

Adaptive Router Although we could apply the same approach to the adaptive router, both the addition of adaptive virtual channels and the higher flexibility to route packets from any input port to any output buffer, increase the number of writing ports to 5 for the output buffers in the X dimension and 7 writing ports for the output buffers on the Y dimension. The complexity of a 7-port buffer under current technology is considerable, so this is not a viable option. Besides, the access to the output port should be multiplexed between the deterministic and the adaptive output buffers, which may potentially add one more stage to the internal pipeline.

However, it is a known fact that in this type of adaptive routers, based on an adaptive virtual network plus a deterministic one used as a escape route [3], the deterministic buffer utilization is generally quite low. This fact led us to propose a hybrid buffering space as shown in figure 1. The buffering space for the deterministic channels is allocated at the input ports, and the buffering space for the adaptive channels is allocated at the output ports. Although this does not fully eliminate the HLB, it significantly reduces it because of the low occupancy of the deterministic input buffers. This scheme reduces the number of writing ports at the output buffer to 4 (3 adaptive inputs plus one deterministic) except for the consumption channel that requires 5 writing ports (any of the 4 adaptive input plus one deterministic), making their implementation technologically viable.

Finally, a crossbar allows for packet movement from the adaptive virtual network to the deterministic one and vice versa as shown in figure 1. The crossbar arbitrates between any virtual channel that wants to use a deterministic output channel, and any deterministic channel (or injection port) that wants to use

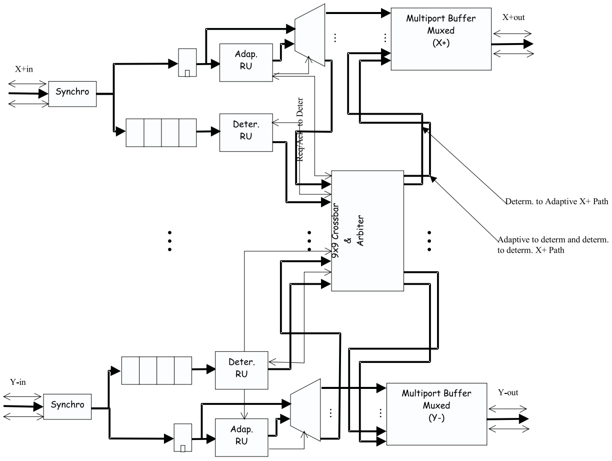

Fig. 1. Architecture of the adaptive Bubble router with hybrid buffering space.

an adaptive output channel. Although the crossbar is not contention free, the potential throughput lost is negligible, because of the low deterministic channel utilization.

3 Hardware Implementation

Once we have proposed the design alternatives to input buffering, we need to evaluate their performance, starting from their implementation cost. The implementation cost is measured by generating a VHDL description of each described router, which is then fed into *Synopsys*, a high-level synthesis tool. This design was mapped into 0.35 μm and 5-layer metal from MIETEC/ALCATEL foundry. The characteristics of each router have been obtained from the synthesis tool. Although we did not descend to the layout level, the results are very close to those of a physical implementation.

This section introduces the most relevant issues regarding the implementation of the routers with output and hybrid buffer space. Therefore, it focuses on the implementation of the multiport memory used for the output buffers. Other blocks present minor modifications from their input buffer router counterparts, whose detailed implementation can be found in [8], [9] and [2].

3.1 Implementation of the Pipeline Multiport FIFO

The design of our output buffers is based on [5] scheme using multiple input reading ports, but just one output port. Thus, messages can be sent out of the output buffer following a FIFO policy. So, the output buffers use a structure which we have called *Pipeline multiport FIFO* or PM-FIFO for short. Figure 2 shows the internal structure of a memory of this type, for a simplified case with 2 writing ports and packet length of 2 phits. The shadowed area (*VC Control*) shows the additional logic for the hybrid buffer scheme (see Fig 1), which multiplexes the adaptive channel (output of the PM-FIFO) with the deterministic channel.

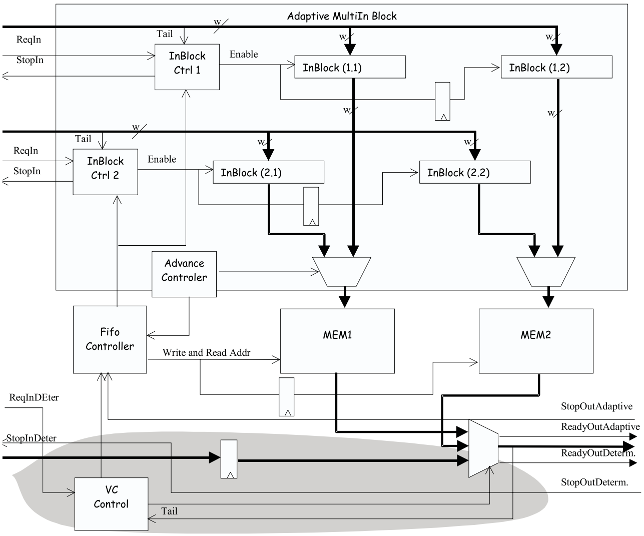

Fig. 2. Output buffer implementation as a Pipeline Multiport FIFO, plus a virtual channel multiplexer (shadowed area).

We can see from figure 2 that the number or stages of the pipeline memory matches the number of phits per packet. However, by setting a word size of 2 phits, we halved the pipeline length. This adds no penalty to the node delay, because the first phit (the header) requires at least one cycle at the RU to be routed to an output buffer.

3.2 Results of the Hardware Implementation

All router implementations share the following conditions: the phit width is 33 bits (4 bytes plus 1 bit tail), and the packet length is set to 10 phits (40 bytes/packet). The total buffer capacity was chosen by examining the gains due to increasing capacity of the adaptive router with hybrid buffers. Experimentally, we observed that little performance is gained by increasing the adaptive channel capacity above 4 packets. Thus, each router requires approx. 1.3 kilobytes. For input buffer routers the length of the pipeline is four cycles. For output buffer routers the length is 5 cycles.

Table 1 shows the characteristics of time and area for each router components for 0.35 μm technology. We should first note that the adaptive router with hybrid buffer improves the cycle time of its input buffer counterpart. The cycle time for the latter router is set by the 9 × 5 crossbar, which is slower than the 9 × 9 crossbar of the hybrid version because it arbitrates two virtual channels per physical output. In fact, the cycle time for the hybrid adaptive router is set by the multiport buffer.

Not only the multiport memory but also the demultiplexers, which provide the direct data paths from each input to any of the output buffers, are responsible for the higher area demands of the routers with output/hybrid buffering.

Module	**Router**							
	InDet.		**InAdapt.**		**OutDet.**		**OutAdapt.**	
	Critical Path (ns.)	Area (mm^2)	Critical Path (ns.)	Area (mm^2)	Critical Path (ns.)	Area (mm^2)	Critical Path (ns.)	Area (mm^2)
Synchr.	2.72	0.020(x5)	2.72	0.020(x5)	2.72	0.020(x5)	2.72	0.020(x5)
Fifo Iny.	3.40	0.462(x1)	3.67	0.444(x1)	3.35	1.405(x1)	3.49	0.504(x1)
Fifo InDet.	**3.40**	**1.69(x4)**	3.67	0.848(x4)	3.35	0.284(x4)	3.49	0.504(x4)
Fifo InAdapt.	–	–	3.67	0.848(x4)	–	–	3.49	0.282(x4)
RU Det.	2.88	0.016(x5)	3.66	0.117(x5)	3.07	0.043(x5)	3.49	0.118(x5)
RU Adapt.		–	3.66	0.117(x4)		–	3.49	0.145(x4)
Crossbar	3.17	0.192(x1)	**3.67**	**0.597(x1)**	–	–	3.49	0.937(x1)
Multiport	–	–	–	–	**3.35**	**1.380(x2)x** **1.760(x2)y**	**3.49**	**1.611(x 4)**
Mulport Cons.	–	–	–	–	**3.35**	**1.574(x1)**	**3.49**	**1.611(x1)**
Total	3.40	7.594	3.67	9.305	3.35	10.710	3.49	13.910

Table 1. Time and area characteristics.

4 Performance Evaluation

The next step is to evaluate the performance of toroidal networks implemented with each of the routers, under the time constraints obtained above. We use a register-level transfer simulator called *SICOSYS* [7], which takes into account the key parameters of the low-level implementation and obtains results which are very close to those of VHDL simulator at a lower computational cost.

Each network has been evaluated under two different scenarios: synthetic loads which represent a variety of patterns present in real applications, and real loads generated by application benchmarks under cc-NUMA architectures.

4.1 Synthetic Loads

We have analyzed the behaviour of each network under random uniform patterns with two length distributions: fixed length (10 phits) and bimodal length(short and long messages). Our bimodal traffic combines 90% of short messages (10 flits) with 10% of long messages (50 phits). We have also considered three non-uniform permutations: *matrix transpose*, *bit-reversal* and *perfect-shuffle*.

Table 2 presents the base latency and peak throughput obtained for an 8×8 toroidal network, for each of the router alternatives. These results show the architectural differences (phits/cycles) of each router as well as their true performance when including their technological cost (phits/nanosecond).

Router	**Random**	**M-Trans**	**Perfec-Shu**	**Bit- Rever**	**Bimodal**
InBuffer Deter.	103.2 11.50 (39.1)	108.3 3.92 (13.3)	103.3 5.27 (17.9)	109.5 3.58 (12.2)	118.9 7.33 (24.9)
InBuffer Adapt.	111.0 10.87 (39.9)	116.8 7.60 (27.9)	111.5 10.14 (37.2)	118.1 8.82 (32.4)	129.2 7.82 (28.7)
OutBuff Deter.	115.7 14.77 (49.5)	124.0 4.08 (14.5)	116.2 6.13 (22.3)	123.0 4.11 (13.7)	130.7 9.11 (30.54)
OutBuff Adap.	119.9 16.30 (56.9)	128.7 9.7 (34.0)	120.1 13.13 (45.8)	129.3 12.35 (43.1)	137.1 11.21 (39.1)

Table 2. Base Latency in nanoseconds (for normalized load of 0.05% with respect to bisection) and maximum throughput accepted in phits/nanosecond (phits/cycle) for a 8×8 Torus.

Regarding the DOR router with output buffers, little gains are observed apart from uniform traffic, which does not justify the additional cost both in silicon area (30% and 15% more than the DOR and adaptive routers with input buffers, respectively) or its higher node delay.

On the other hand, the adaptive router with hybrid buffering achieves significant throughput gains under all traffic patterns in comparison with the other three alternatives. Not only does it outperform the DOR routers for non-uniform traffic, but it also achieves up to 40% higher throughput that its input buffer counterpart. This improvement indicates that the head-of-line blocking (HLB) and, to a lesser extent, the crossbar arbitration for the same output are limiting the performance of the adaptive routing algorithm. On the other hand, it increases base latency by less than 10% compared with the input buffer alternatives. In terms of cost, it requires 50% more area than the adaptive router with input buffers, and 90% (10%) more than the DOR router with input (output) buffering. This cost is easily justified for throughput-sensitive applications.

4.2 Real Loads

Finally, we have evaluated the impact of each network implementation on the performance of parallel applications running on a cc-NUMA architecture. This evaluation is carried out by using the tool ED-SYCOSYS [7] which is based on the RSIM simulator [6] replacing the original RSIM's network by our network simulator (SYCOSYS).

Due to space limitations we will only present the result for the FFT application which belongs to the SPLASH-2 suite [11]. The system is configured with the default values provided by RSIM except for the parameters discussed below. We set the cache line size to 32 bytes, and assume that network commands are 8 bytes long. Hence, data and command packets have a fixed length of 40 bytes (10 phits) and 8 bytes (2 phits) respectively. The network size is set to 64 nodes (8 × 8 torus). Three cases were simulated: two of them with separate data and control networks and processor speeds of 600 MHz and 1GHz respectively, and a third one in which data and control messages share a single network with 1GHz processor nodes. The latter network prevents fetch deadlock by providing sufficiently large consumption buffers at the network interface. The problem size is set to 64K complex doubles.

	600 MHz Processor	**1GHz Processor**	**1GHz Processor**
	Dual Network	**Dual Network**	**Unified Network**
InBuff. Deter.	621656	870482	961864
InBuff. Adapt.	589405	810486	870770
OutBuff. Deter.	629711	886967	956921
OutBuff. Adapt.	586535	799080	842776

Table 3. Execution time, in processor cycles, for FFT with 64K complex doubles over 64 nodes (Torus 8 × 8).

Table 3. shows the execution times for each experiment. The network traffic is only high for short periods of time, hence the small time variations observed in spite of the different network capabilities. As the volume of traffic increases with problem size, so do the time differences between the 4 network alternatives. Due to the complexity of the simulation environment, the study of larger problems was not feasible, but we could extract some conclusions from the above results.

5 Conclusions

An in-depth analysis of the router's buffer organization was carried out for a toroidal network which uses either a deterministic (DOR) or a fully adaptive routing scheme. Two alternatives to the simple input FIFO organization were proposed, one for each routing scheme. After a thorough evaluation, we can draw the following conclusions:

1. In both cases, the elimination of head-of line blocking (HLB) produces an improvement on network performance; but only the adaptive router presents significant gains under non-uniform traffic, which justify its cost in silicon area.
2. Output buffering increases node complexity, so throughput gains must outweigh the penalty on node delay. Such is the case for the adaptive router

which gains 40% throughput with less than 10% increment in base latency. The node delay penalty for the DOR router is similar, but the throughput gains are negligible.

3. The analysis of execution time for parallel applications shows, once more, the required balance between network latency and throughput. In a cc-NUMA environment, applications exhibit execution phases with either low or high network load. The former phases benefit from low latency and the latter ones from high throughput.

Finally, we should note the importance of the evaluation method which takes into account the architectural choices, the technological constraints and the application demands. None of them could be ignored when designing the interconnection network for a parallel system.

References

[1] C. Carrión, R. Beivide, J.A. Gregorio, F. Vallejo, “A flow control mechanism to avoid message deadlock in k-ary n-cube networks,” *Fourth International Conference on High Performance Computing*, pp. 322-329, India, December,1997.

[2] C. Carrion, R. Beivide, J.A. Gregorio, ”Performance Evaluation of Bubble Algorithm: Benefits for k-ary n-cubes. 7th Euromicro on Parallel and Distributed Processing, Madeira, Portugal 1999.

[3] J. Duato,”A necessary and sufficient condition for deadlock-free routing in cut-through and store-and-forward networks”. *IEEE Trans. on Parallel and Distributed Systems*, vol.7, no.8, pp.841-854, August 1996.

[4] M. Karol, M. Hluchyj,S. Morgan “Input Versus Output Queuing on Space Division Packet Switch”, IEEE Trans. On Communications, vol. COM-35, no. 12, Dec. 1987, pp. 1347-1356.

[5] M. Katevenis, P. Vatsolaki, A. Efthymiou “Memory Shared Buffer for VLSI Switches”, ACM SIGCOMM, August 1995.

[6] V. S. Pai, P. Ranganathan, S. Adve ”Rsim: An execution-Driven Simulator for ILP-Based Shared-Memory Multiprocessors and Uniprocessors”, *IEEE TCCA Newsletter*, Oct. 1997.

[7] J.M. Prellezo, V. Puente, J.A. Gregorio, R. Beivide, “SICOSYS: a interconnection network simulator for parallel computers,” available at http://www.atc.unican.es/REPORTS/TR-ATC2-UC98.pdf, June 1998.

[8] V. Puente, J.A. Gregorio,J. M. Prellezo, R. Beivide,J. Duato, C. Izu “Adaptive Bubble Router: a Design to Balance Latency and Throughput in Networks for Parallel Computers”, *ICPP’99*, Sept. 1999.

[9] V. Puente, J.A. Gregorio, C. Izu, R. Beivide, F. Vallejo “Low-level Router Design and its Impact on Supercomputer System Performance”, ICS’99, July 1999.

[10] S. L. Scott, G. Thorson, ”The Cray T3E network: Adaptive routing in a high performance 3-D torus”, *Hot Interconnects Symposium IV*, pp. 147-155, Aug. 1996.

[11] S. C. Woo, M. Ohara, E. Torrie, J.P. Singh, A. Gupta ”The SPLASH-2 Programs: Characterization and Methodological Considerations”. In Proceedings of the *22nd International Symposium on Computer Architecture*, pages 24-36. June 1995.