Providing Full QoS with 2 VCs in High-speed Switches*

A. Martínez[1], F. J. Alfaro[1], J. L. Sánchez[1], and J. Duato[2]

[1] Departamento de Sistemas Informáticos, Escuela Politécnica Superior
Universidad de Castilla-La Mancha, 02071 - Albacete, Spain
{alejandro,falfaro,jsanchez}@dsi.uclm.es
[2] Dept. de Informática de Sistemas y Computadores, Facultad de Informáica
Universidad Politécnica de Valencia, 46071 - Valencia, Spain
jduato@disca.upv.es

Abstract. Current interconnect standards propose 16 or even more virtual channels (VCs) for provision of quality of service (QoS). However, VCs increase the complexity of the switch and the scheduling delays. In a previous paper, we have shown how to use only two VCs for full QoS support at the switches. In this paper, we explore thoroughly two alternative switch designs that take advantage of this reduction. We analyze their feasibility in a single chip implementation and show that they get a noticeable performance while greatly reducing the cost and power consumption of the network.

1 Introduction

There are many proposals for quality of service (QoS) support in high-speed interconnects. Most of them incorporate 16 or even more virtual channels (VCs), devoting a different VC to each traffic class. In most of the recent switch designs, the buffers are the most silicon area consuming part.

To build the 64-port MIN, we need 48 switches and 192 links when using the 8-port switch designs (*Traditional 8 VCs*, *Traditional 2 VCs*, and *New 2 VCs-B* cases). The final cost of the interconnect greatly depends on the number of switches used, while the power consumption comes mostly from the transceivers needed to drive the links, and thus, depends on the actual number of links [1]. Note that with the *New 2 VCs-P* switch model, it takes just 16 switches (67% less than the traditional case) and 128 links (33% less than the traditional case) to build the 64-port MIN. This greatly reduces the cost and power-consumption of the network.

The buffers at the ports are usually implemented with a memory space organized in logical queues, which consist of linked lists of packets, with pointers to manage them. Therefore, the complexity and cost of the switch heavily depend

* This work was partly supported by the Spanish CICYT under grant TIN2006-15516-C04, by the Junta de Comunidades de Castilla-La Mancha under grant PBC-05-005, and by the Spanish State Secretariat of Education and Universities under FPU grant.

on the number of queues at the ports. For instance, the crossbar scheduler has to consider 8 times the number of queues if 8 VCs are implemented (greatly increasing the area and power consumed by this scheduler). Then, a reduction in the number of VCs (and in the required buffer space) necessary to support QoS can be very helpful in the switch design and implementation.

In [2] we have proposed a strategy to use just two VCs at each switch port for the provision of QoS that obtains similar results as if we were using many more VCs. This is achieved by respecting at the switches some of the scheduling decisions made at network interfaces. Moreover, to provide guarantee even to the low-priority flows and to prevent starvation, a connection admission control (CAC) is used. In this way, at no link the load of regulated traffic will be higher than the available bandwidth.

In this paper, we throughly explore two alternative switch designs that take advantage of this reduction and we evaluate them with different traffic models. Simulation results show that our proposal provides a very similar performance compared with a traditional architecture with many more VCs both for the QoS traffic and for the best-effort traffic. Moreover, comparing our technique with a traditional architecture with 2 VCs, our proposal provides a significant improvement in performance for the QoS traffic, while for the best-effort traffic, the traditional model is unable to provide the slightest differentiation.

There are several differences between this study and the one at [2]. The first is that here we have made a thorough study of silicon area consumption by switch components taking into account the total silicon area available. Moreover, we also propose a chip design that takes advantage of the reduction of VCs. In addition to this, we offer quantitative results concerning the advantages of using our proposal, in terms of component count and power consumption. Finally, in [2] we considered a theoretical model for scheduling times, while here we take an approach based on the actual ASIC design of the switch.

The remainder of this paper is structured as follows. In the following section the related work is presented. In Section 3 we present our strategy to provide QoS support with only two VCs and we study its hardware implications. The details on the experimental platform are presented in Section 4 and the performance evaluation in Section 5. Finally, Section 6 concludes this study.

2 Related work

During the last decade several switch designs with QoS support have been proposed. All of them incorporate VCs in order to provide QoS support. InfiniBand was proposed in 1999 both for communication between processing nodes and I/O devices and for interprocessor communication. InfiniBand Architecture (IBA) [3] proposes three main mechanisms to provide the applications with QoS, including the use of up to 16 VCs.

PCI Express Advanced Switching (AS) architecture is the natural evolution of the traditional PCI bus [4]. It defines a switch fabric architecture that supports high availability, performance, reliability, and QoS. AS ports incorporate up to 20 VCs that are scheduled according to some QoS criteria.

These proposals, therefore, use a significant number of VCs to provide QoS support. However, if a great number of VCs are implemented, it would require a significant fraction of silicon area and would make packet processing slower. Note that this paper deals with single-chip switches, where the buffers, the crossbar, and the scheduler are inside the same chip, in order to be able to provide the low latency necessary in current high-performance networking.

Traditional 2 VC proposals distinguish between just two broad categories (regular and premium) [5]. In contrast, the novelty of our proposal lies in the fact that, although we use only two VCs at the switches, the global behavior of the network is very similar as if the switches were using many more VCs. This is because we are respecting at the switch ports the scheduling decisions performed at the network interfaces, which have as many VCs as traffic classes. In the end, the network provides a differentiated service to all the traffic classes considered.

3 Providing full QoS Support with only two VCs

In [2] we have proposed a new strategy to use only two VCs at each switch port to provide QoS. It achieves similar performance results to those using many more VCs. We review this proposal in this section.

Supporting a large number of queues in a switch is not easy. This affects the scheduling performed to configure the crossbar and the arbitration at the output ports. We are referring to classical unbuffered crossbars. However, a different switch architecture exists, that uses buffered crossbars [6]. In this case, the buffer space is neither at the inputs nor the outputs of the switch, but distributed at the crosspoints of the crossbar. When using this design, supporting many queues is even more difficult, since the space at the crosspoint buffers is very limited and to implement many queues is not possible. Moreover, in both crossbar types, the control data structures needed for managing the queues consume silicon area and switch control unit cycles. Therefore, proposals of 8 or 16 VCs for QoS support are very rarely implemented and, as we mentioned before, the trend is to increase the number of ports per switch, instead of the number of VCs.

Traditionally, network designers have overdimensioned the network in order to provide an acceptable QoS. However, this solution is becoming less and less interesting since the network is becoming the most expensive and power-consuming part of the system [1].

The key idea of our proposal is based in this observation: Assuming that the links are not oversubscribed, all the traffic flows through the switches seamlessly. Therefore, the basic idea of our proposal consists in using only two VCs at the switch ports. One of these VCs is used for QoS packets and the other for best-effort packets. Moreover, we propose to use a connection admission control (CAC) to guarantee that QoS traffic will not oversubscribe the links and we give QoS traffic absolute priority over best-effort traffic, which is not subject to the CAC.

Moreover, the network interfaces are responsible for injecting traffic of the different classes applying any desired algorithm. At the same time, the switches reuse this scheduling. This is the cornerstone of our proposal: To reuse at the switches the scheduling decisions taken at the host interfaces.

We assume that a static priority criterion exists to order packets. In this way, every packet would be stamped with a priority level (typically, 8 or 16 levels). This is necessary because packets arriving at the switches come in the order specified by the interfaces, and the switch has to merge these packet flows at the output ports. The way of performing this is very simple: The scheduler takes into account the service level of the packets (8 or 16 priorities), not just if they are at the QoS or best-effort VC. This is not very complex because very efficient priority encoder circuits have been proposed [7]. On the other hand, from the scope of these priority assignments, the packet ordering established at network interfaces does not need to be changed at any switch in the path because queuing delays for QoS traffic will be short.

Obviously, the network interface can only arbitrate among the packets it holds at a given moment. Therefore, when no more high-priority packets are available, a low-priority QoS packet can be transmitted. If this packet has to wait at a switch input queue, and other packets with higher priority are transmitted from the network interface, they would be stored in the same VC as the low-priority packet, and would be placed after it in the queue. Thus, the arbiter would penalize the high-priority packets, because they would have to wait until the low-priority packet is transmitted. But this situation has a small impact on performance because there is bandwidth reservation for QoS packets. This means that all the QoS packets will flow with short delay.

Remember that, although we assume that QoS traffic does not oversubscribe any link, no assumption is made about best-effort traffic. However, the network interfaces are still able to assign the available bandwidth (the one not consumed by QoS traffic) to the best-effort traffic in the configured proportions. In this way, they can still take into account the QoS requirements of this kind of traffic. Obviously, this is a coarse-grain QoS provision. If stricter guarantees were needed by a particular flow, it should be classified as QoS traffic.

Note that this proposal does not aim at achieving a higher performance but, instead, at drastically reducing buffer requirements while keeping the performance and behavior of systems with many more VCs. In this way, a sophisticated QoS support could be implemented at an affordable cost.

Summing up, our proposal consists in reducing the number of VCs at each switch port needed to provide flows with QoS. Instead of having a VC per traffic class, we propose to use only two VCs at switches: One for QoS packets and another for best-effort packets. Moreover, the scheduling decisions performed at network interfaces are reused at switches. In order for this strategy to work, we guarantee that there is no link oversubscription for QoS traffic by using a CAC strategy.

3.1 Implementation considerations

In order to find out the advantages of using our approach, we have evaluated the cost of implementing the switch both with the 8 VCs (one per traffic class) and with just 2 VCs. In both cases, we have considered a combined input-output queuing architecture, because it is a common design for high-performance switches. Note that all the switch components must be implemented into a single chip, in order to be able to provide the low latency necessary in high-performance networking.

We have considered two switch designs in this section. The first benefits from our proposal and uses the saved silicon area to implement additional ports at the switch. In this case, the switch would have 16 ports. The other design would be a traditional 8 VCs switch, where only 8 ports would be possible.

The components we have considered for this study are the ports (mainly buffer space with some additional logic), the crossbar, and the scheduler. We have taken the design constraints like packet format, routing, and so on, from the PCI AS specification [4]. Table 1 shows area estimates for each module for two different technologies: 0.18 and 0.13 μm.

Table 1. Area consumption by components.

	16 ports - 2 VCs		8 ports - 8 VCs	
Module	Tech. 0.18 μm	Tech. 0.13 μm	Tech. 0.18 μm	Tech. 0.13 μm
Buffers	64 mm^2	32 mm^2	64 mm^2	32 mm^2
Xbar and datapath	10 mm^2	5 mm^2	5 mm^2	3 mm^2
Scheduler	5 mm^2	3 mm^2	10 mm^2	5 mm^2
Total	79 mm^2	40 mm^2	79 mm^2	40 mm^2

The internal clock of the system is 250 MHz and the datapath is 64 bits wide. This provides a speed of 16 Gbits/s, which is twice the speed of the external links. That means there is an internal speed-up of 2.0.

The buffers are 16 Kbytes per port with our proposal (which are shared between the two VCs, 8 Kbytes each) and 32 Kbytes per port in the 8 VCs case (4 Kbytes each VC), both as a compromise between silicon area and performance. The memory area estimates are based on datasheets of typical ASIC (Application-Specific Integrated Circuit) technologies available to European Universities. These numbers include the input and output buffers.

The crossbar and datapath estimates come from the actual numbers of the switch design in [8]. Finally, we base our estimates for the scheduler area on the data provided by McKeown at [9]. The increased area for the scheduler in the 8 VCs design takes into account the queues needed if 8 VCs are implemented (64 queues, 8 VCs multiplied by 8 output ports). The 2 VCs design needs half of the queues (32 queues, 2 VCs multiplied by 16 output ports).

Note that usually there is not a full utilization of the available area in an ASIC design and, therefore, the final chip would be larger. In order to find out more accurate estimates, all the design flow should be performed. However, these area estimates are very helpful to compare the alternative architectures.

Finally, we could also implement an 8-port switch design using our proposed 2 VCs, but with increased buffer space. It would be very similar to the 8 VCs design we have described, but just 16 queues per input port would be necessary (2 per output port). In the performance evaluation section we will evaluate both alternatives for applying our proposal, compared with more traditional solutions.

4 Simulation conditions

In this section, we will give details on the simulated network architecture and the load used for the evaluation.

4.1 Simulated architecture

We have performed the tests considering four cases. First, we have tested the performance of our proposal, which uses 2 VCs at each switch port. It is referred to in the figures as *New 2 VCs*. Note that, with our proposal, the network interfaces still use 8 VCs. We have considered the two variants of this proposal presented in Section 3.1, which are noted *New 2 VCs-P* (16 ports) and *New 2 VCs-B* (8 ports, but larger buffers per VC).

We have also performed tests with switches using 8 VCs. In this case, it is referred to in the figures as *Traditional 8 VCs*. Finally, we have also tested a traditional approach with 2 VCs, noted in the figures as *Traditional 2 VCs*. In this case, the network interfaces also use 2 VCs. Therefore, we have two references to compare the performance of our proposals, one being the lower bound (*Traditional 2 VCs*) and the other the upper bound (*Traditional 8 VCs*). *Traditional 2 VCs* switches have the same number of ports than *Traditional 8 VCs*, but they have 4 times more buffer space per VC, allowing a good performance with bursty traffic.

The network used to test the proposals is a butterfly multi-stage interconnection network (MIN) with 64 end-points. The actual topology is a folded (bidirectional) perfect-shuffle. We have chosen a MIN because it is a usual topology for clusters. However, our proposals are valid for any network topology, including both direct networks and MINs. No packets are dropped because we use credit-based flow control between the switches at the VC level.

The CAC we have implemented is a simple one, based on average bandwidth. Each connection is assigned a path where enough resources are assured. It guarantees that less than 70% bandwith is used by QoS traffic at any link. We also use a load-balancing mechanism, which consists in assigning the least occupied route among those possible. The parameters of the network elements used in this performance study aretaken from PCI AS specification [4].

Table 2. Traffic injected per host.

TC	Name	% BW	Packet size	Notes
7	Network Control	1	[64,512]	self-similar
6	Audio	16.333	128	CBR 64 KB/s conn.
5	Video	16.333	[64,2048]	750 KB/s MPEG-4 trc.
4	Controlled Load	16.333	[64,2048]	CBR 1 MB/s conn.
3	Excellent-effort	12.5	[64,2048]	self-similar
2	Pref. Best-effort	12.5	[64,2048]	self-similar
1	Best-effort	12.5	[64,2048]	self-similar
0	Background	12.5	[64,2048]	self-similar

4.2 Traffic model

In Table 2, the characteristics of the modeled traffic are included. We have considered the traffic classes (TCs) defined by the IEEE standard 802.1D-2004 [10] at the Annex G, which are generally accepted for interconnection networks. However, we have added an eighth TC, *Preferential Best-effort*, with a priority between *Excellent-effort* and *Best-effort*.

The destination pattern of the traffic injected is based on Zipf's law [11], as recommended by the *network processing forum switch fabric benchmark specifications* [12]. In this way, there will be hot spots in the network.

The packets are generated according to different distributions, as can be seen in Table 2. *Audio*, *Video*, and *Controlled Load* traffic are composed of point-to-point connections of the given bandwidth. The self-similar traffic is composed of bursts of 60 packets heading to the same destination, with the packets' sizes governed by a Pareto distribution and the periods between bursts modelled with a Poisson distribution. With this distribution there is a lot of temporal and spatial locality and should show worst-case behavior.

5 Simulation results

In this section, the performance of our proposals is shown. We have considered three traditional QoS indices for this performance evaluation: Throughput, latency, and jitter. Note that packet loss is not considered because no packets are dropped due to the use of credit-based flow control. We also show the cumulative distribution function (CDF) of latency and jitter, which represents the probability of a packet achieving a latency or jitter equal to or lower than a certain value.

Figure 1 shows the performance of QoS traffic. The average latency results are very similar for the four architectures. On the other hand, we can also see the CDF of latency and jitter at a normalized network load of 1.0. The *Traditional 8 VCs* case offers the best results. With our proposal, a small portion of the packets see their latency increased, and thus, the jitter. However, this is a small handicap compared with the benefits, as we will see later.

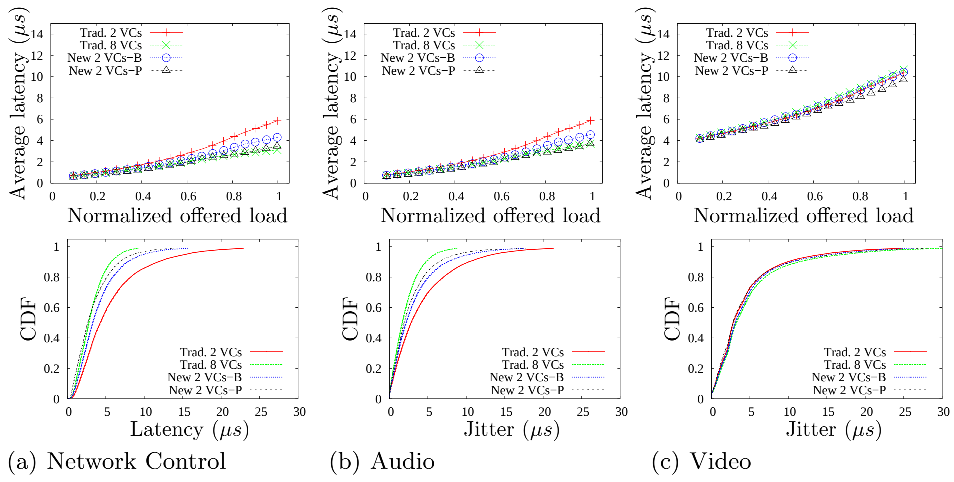

Fig. 1. Results for QoS traffic classes.

In Figure 2 we evaluate the best-effort traffic. In this case, the *Traditional 2 VCs* approach produces the same performance for all the TCs, which is an inadequate behavior, because excellent-effort and preferential best-effort traffic classes should have better performance. The reason for this inadequate behavior is that in the *Traditional 2 VCs* model, all the best-effort classes look the same for the schedulers at both the network interfaces and switches.

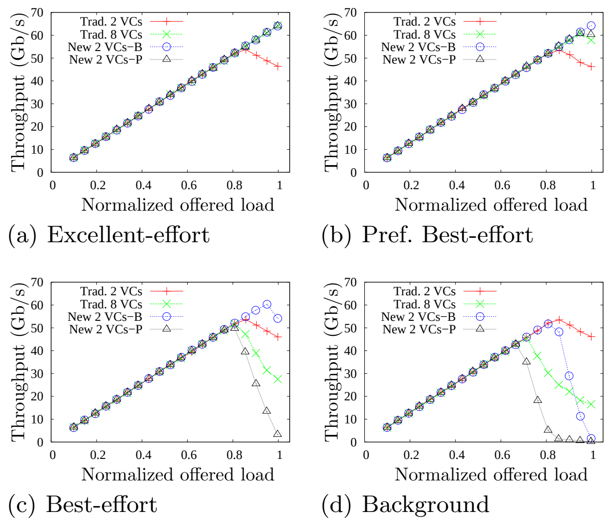

Fig. 2. Throughput of best-effort traffic classes.

On the other hand, the arbiters using our technique take into account the priority of the packets, although they share the same VC. For that reason, our proposals, which devote a single VC at the switches for all the best-effort TCs, can provide a behavior similar to that of the *Traditional 8 VCs* approach, which uses 4 VCs for the best-effort TCs.

The *New 2 VCs-B* offers the best performance because it provides the maximum flexibility in the use of the buffer space, while in the *Traditional 8 VCs* case the same amount of memory is statically partitioned. On the other hand, the *New 2 VCs-P* case offers a worse performance because the buffer space is smaller.

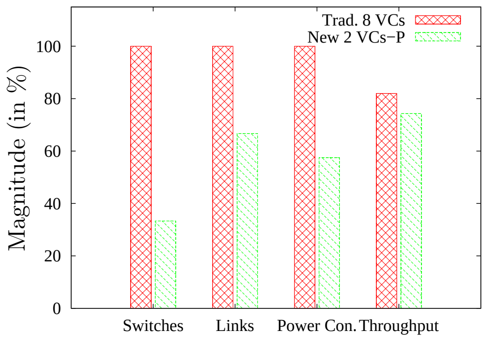

Fig. 3. Summary of the trade-offs of the *New 2 VCs-P* proposal, compared with the *Traditional 8 VCs* case.

Figure 3 summarizes the trade-offs of using our *New 2 VCs-P* proposal. As can be seen, there is a very noticeable reduction in chip[3] and link counts and, therefore, in the associated power consumption of the interconnection network (calculated from estimates of the ASIC chip design). On the other hand, the global throughput achieved with bursty traffic is slightly lower (only a 10% reduction), but the reduction on the component count would lead to a much more cost-effective solution. Also note that the non-uniform traffic pattern for best-effort traffic is very bursty: Throughput under more uniform traffic is higher for both *Traditional 8 VCs* and *New 2 VCs-P* cases.

According to these results, we can conclude that our proposals can provide an adequate QoS performance. The reduction of the number of VCs to just two allows us to offer two alternative switch designs: The first keeps the expenses but produces a higher global throughput by using more flexible buffers, and the second greatly decreases the cost and power-consumption of the interconnection with a small degradation in performance (13% global throughput less than *Traditional 2 VCs* with non-uniform traffic).

[3] Remember that both *New 2 VCs-P* and *Traditional 8 VCs* switches take equivalent sillicon area.

6 Conclusions

In [2] we presented a proposal to use only two VCs at each switch port to provide QoS support. The first VC is used for QoS traffic and the other for best-effort traffic. In this way, we obtained a drastic reduction in the number of VCs required for QoS purposes at each switch port. In that paper, we showed preliminary results using multimedia traffic in a uniform scenario, without a clear study on how to apply this VC reduction.

This paper offers two novel designs for the switch that benefit from a reduction on the number of necessary VCs to provide QoS. The first design increases the global performance of the network against unbalanced traffic by using buffer space in a flexible way. The second design greatly reduces the component count, and thus, the cost and power-consumption of the interconnection, at the cost of a small degradation of performance. However, even under worst-case traffic, this degradation of performance is worthy compared with the savings it brings.

References

1. Shang, L., Peh, L.S., Jha, N.K.: Dynamic voltage scaling with links for power optimization of interconnection networks. In: Proceedings of the 9th Symposium on High Performance Computer Architecture (HPCA). (2003) 91–102
2. Martínez, A., Alfaro, F.J., Sánchez, J.L., Duato, J.: Providing full QoS support in clusters using only two VCs at the switches. In: Proceedings of the 12th International Conference on High Performance Computing (HiPC). (2005) 158–169 Available at `http://www.i3a.uclm.es/documentos/2/congresos/Congreso2_134_HiPC05.pdf`.
3. InfiniBand Trade Association: InfiniBand architecture specification volume 1. Release 1.0. (2000)
4. ASI SIG: Advanced switching core architecture specification. (2005)
5. Dally, W., Carvey, P., Dennison, L.: Architecture of the Avici terabit switch/router. In: Proceedings of the 6th Symposium on Hot Interconnects. (1998) 41–50
6. Duato, J., Yalamanchili, S., Lionel, N.: Interconnection networks. An engineering approach. Morgan Kaufmann Publishers Inc. (2002)
7. Huang, C., Wang, J., Huang, Y.: Design of high-performance CMOS priority encoders and incrementer/decrementers using multilevel lookahead and multilevel folding techniques. IEEE Journal of Solid-State Circuits **1** (2002) 63–76
8. Simos, D.: Design of a 32x32 variable-packet-size buffered crossbar switch chip. Technical Report FORTH-ICS/TR-339, Inst. of Computer Science, FORTH (2004)
9. McKeown, N.W.: The iSLIP scheduling algorithm for input-queued switches. IEEE/ACM Transactions on Networking **7** (1999) 188–201
10. IEEE: 802.1D-2004: Standard for local and metropolitan area networks. `http://grouper.ieee.org/groups/802/1/` (2004)
11. Zipf, G.K.: The Psycho-biology of Languages. Houghton-Miffin, MIT (1965)
12. Elhanany, I., Chiou, D., Tabatabaee, V., Noro, R., Poursepanj, A.: The network processing forum switch fabric benchmark specifications: An overview. IEEE Network (2005) 5–9