Scalable Low-cost QoS Support for Single-chip Switches*

A. Martínez, F. J. Alfaro, J. L. Sánchez
Dept. de Sistemas Informáticos
Universidad de Castilla-La Mancha
02071 - Albacete, Spain
{alejandro,falfaro,jsanchez}@info-ab.uclm.es

J. Duato
Dept. de Informática de Sistemas y Computadores
U. Politécnica de Valencia
46071 - Valencia, Spain
jduato@disca.upv.es

Abstract

Virtual channels (VCs) are a popular solution for the provision of quality of service (QoS). Current interconnect standards propose 16 or even more VCs for this purpose. However, most commercial implementations do not offer so many VCs because it is too expensive in terms of silicon area. Therefore, a reduction of the number of VCs necessary to support QoS can be very helpful in the switch design and implementation.

We have shown that this number of VCs can be reduced if the system is considered as a whole rather than each element being taken separately. Some of the scheduling decisions made at network interfaces can be easily reused at switches without significantly altering the global behavior. In this paper, our aim is to explore the scalability of the technique, considering the restrictions of the final chip implementation.

Keywords: *Quality of service, switch design, interconnection networks, SANs, clusters, performance evaluation*

1. Introduction

The last decade has witnessed a vast increase in the amount of information and services available through the Internet. These services rely on applications executed in many servers all around the world. Clusters of PCs have emerged as a cost-effective platform to implement these services and run the required Internet applications. These clusters provide service to thousands or tens of thousands of concurrent users. Many of these applications are multimedia applications, which usually present bandwidth and/or latency requirements [13]. These are known as quality of service (QoS) requirements.

*This work was partly supported by the Spanish CICYT under grant TIC2003-08154-C06, by the Junta de Comunidades de Castilla-La Mancha under grant PBC-05-005-1, and by the Spanish State Secretariat of Education and Universities under FPU grant.

In the next section, we will be looking at some proposals to provide QoS in clusters. Most of them incorporate 16 or even more VCs, devoting a different VC to each traffic class. This increases the switch complexity and required silicon area and also prevents the use of these VCs for other purposes. Moreover, it seems that, when the technology enables it, the trend is to increase the number of ports instead of increasing the number of VCs per port [12].

In most of the recent switch designs, the buffers are the most silicon area consuming part (see [15] for a detailed design). 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 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 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 [11], we propose a strategy to use just two VCs at each switch port for the provision of QoS that allows very similar results as if we were using many more VCs. One of these two VCs is used for QoS traffic and the other one for best-effort traffic. In this way, we obtain a noticeable reduction of buffer space. This is achieved by reusing in 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 no link will receive more bandwidth than it is able to handle. This proposal has been tested using multimedia traffic. Simulation results have shown that applications achieve a similar QoS performance, but using fewer VCs.

In this paper, we explore the scalability of the technique. We will examine different network sizes to confirm that the results are acceptable with the bigger ones. We also consider the restrictions of the final chip implementation, ex-

amining two different alternatives for the saved VCs. Finally, this performance evaluation considers bursty traffic in all the traffic classes, offering a worst-case scenario.

The remainder of this paper is structured as follows. In the next section, the related work is presented. In Section 3, we review our strategy to provide QoS support with only two VCs. The details on the experimental platform are presented in Section 4 and the performance evaluation in Section 5. Finally, Section 6 summarizes 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. In these proposals, different scheduling algorithms are used to arbitrate between the different existing traffic flows, providing each one with QoS according to its requirements.

The Multimedia Router (MMR) [5] is a hybrid router. It uses pipelined circuit switching for multimedia traffic and virtual cut-through for best-effort traffic. Pipelined circuit switching is connection-oriented and needs one VC per connection. This is the main drawback of the proposal because the number of VCs per physical link is limited by the available buffer size and there may not be enough VCs for all the possible existing connections (in the order of hundreds). Therefore, the number of multimedia flows allowed is limited by the number of VCs. Moreover, the scheduling among hundreds of VCs is a complex task.

InfiniBand was proposed in 1999 by the most important IT companies to provide present and future server systems with the required levels of reliability, availability, performance, scalability and QoS [9]. Specifically, the InfiniBand Architecture (IBA) proposes three main mechanisms to provide the applications with QoS. These are traffic segregation with service levels, the use of VCs (IBA ports can have up to 16 VCs) and the arbitration at output ports according to an arbitration table. Although IBA does not specify how these mechanisms should be used, some proposals have been made to provide applications with QoS in InfiniBand networks [2].

Finally, PCI Express Advanced Switching (AS) architecture is the natural evolution of the traditional PCI bus [1]. 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. In the AS specifications, two arbiters are proposed, one of them being table-based and the other based on bandwidth allocation.

These proposals, therefore, use a significant number of VCs to provide QoS support. Moreover, 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.

Traditional two VC proposals distinguish between just two broad categories (regular and premium) [4]. 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 reusing at the switch ports the scheduling decisions performed at the network interfaces, which have as many VCs as traffic classes (8 VCs in our performance evaluation).

To the best of our knowledge, only Katevenis and his group [3] have proposed something similar before. However, their proposal is aimed at a single-stage router based on a single buffered crossbar. In this crossbar there are small buffers at the crosspoints that the authors split into two VCs. In contrast, our proposal is a simpler and more general technique, as we will see in the next section.

3. Providing full QoS support with two VCs

In [11] 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 [14].

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 [8]. 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.

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

The reduction of VCs can be reflected in two different ways in the final design. The first is to reduce the amount of memory at each port. By doing so, it is possible to increase the number of ports on the chip. This change is not direct, i.e. if we reduce VCs in a factor of 4, it is not directly translated into 4 times more ports because the complexity of the crossbar and the scheduler also depend on the number of ports. However, taking into account the numbers at [15] and memory area datasheets, we can make a conservative estimate: Going from 8 to 2 VCs roughly translates into 2 times more ports and also allows an increase of 50% the memory per VC. More details of this can be found in Section 4.1.

The advantage of the aforementioned approach is that the resulting network is cheaper (less switches) and less power-consuming (increasing the number of ports reduces the number of links to build up the network). On the other hand, the total amount of buffer space per port is reduced, which may lead to poor performance under unbalanced and bursty traffic. For that reason, the other alternative is to just keep the same amount of memory per port when going from more VCs to only 2, but gaining the advantage that this memory can be shared by more flows. Under these conditions, we expect a better behavior with unbalanced traffic. This option does not decrease the cost or the power consumption of the network, but leads to a better performance.

4. Simulation conditions

In this section, we will explain the simulated network architecture and how the previous proposals are translated to the network design. We will also give details on the parameters of the network and the load used for the evaluation. The main differences with the study at [11] are:

- We study the influence of the network size.
- We give two alternatives for the chip design that take into account in different ways the reduction of VCs.
- We study the performance of the technique in two different scenarios, uniform traffic and bursty traffic, serving the latter as a worst-case scenario.

- We offer quantitative results of the advantages of using our proposal, in terms of component count.
- In [11] we considered a theoretical model for scheduling times, while here we take an approach based on the actual design of the switch.

4.1. Simulated architecture

Our objective is to evaluate the performance of our technique under fair conditions. In order to achieve this, we will define a complete network architecture, including all the elements necessary for it to work. However, many of them are not directly related with our work. We aim at defining a fair scenario in which to compare several switch architectures. For that reason, we have not used state-of-the-art routing techniques, congestion control mechanisms, etc., but the most popular and well-known solutions.

First, we have tested the performance of our proposal, which uses 2 VCs at each switch port (the network interfaces still use 8 VCs). The two variants of this proposal are noted *New 2 VCs-P* (more ports) and *New 2 VCs-B* (larger buffers for the only two VCs).

We have also performed tests with switches using 8 VCs (as many VCs as traffic classes). 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 at switch ports and network interfaces, noted in the figures as *Traditional 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 the former have 4 times more buffer space per port, allowing a better performance with bursty traffic.

The network used to test the proposals is a butterfly multi-stage interconnection network (MIN) varying from 64 to 512 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. The switches use a combined input and output buffer architecture, with an unbuffered crossbar to connect the buffers. No packets are dropped because we use credit-based flow control between the switches.

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, leaving the rest of the bandwidth for best-effort traffic. We also use a load-balancing mechanism, which consists in assigning the least occupied route among the possible paths. The main parameters of the network elements of this performance study are given in Table 1.

Table 1. Main simulation parameters

Packet size	[64,2048] bytes
Header size	8 bytes
Credits msg. size	6 bytes
Channel BW	8 Gb/s
Crossbar BW	16 Gb/s

In the four switch models, virtual output queuing (VOQ) is implemented to solve the head-of-line blocking problem at the switch level. The implementation of VOQ is achieved with dynamic lists of blocks of 64 bytes at the buffers. The block size matches the transfer unit of the crossbar.

There is a speed-up of 2.0 in the crossbar of the four switch models. Note that this is internal speed-up (the external links keep their throughput) and is achieved by doubling the word size in the access to the memories and at the crossbar. Note that some of the internal speed-up is consumed in padding for packets whose length is not a multiple of the block size due to the synchronous configuration of the crossbar. Note also that the block size and the crossbar throughput allow a scheduling delay of 32 ns (64 bytes at 16 Gb/s), enough for several iterations of the scheduler.

The *Traditional 8 VCs* single-chip switch provides 2 packets buffering for each VC (a total of 4 Kbytes per VC, 32 Kbytes per port). To build the 64 ports MIN, we need 48 switches and 192 links. 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 [14]. This design would take 100-150 mm^2 using 180 nanometers technology, according to our calculations based on [15].

The *Traditional 2 VCs* and *New 2 VCs-B* designs simply take as a starting point the *Traditional 8 VCs* and unify the 8 VCs into just two buffers. This would simplify the scheduler design (less requests to attend) but we have supposed equal area and delay than *Traditional 8 VCs* for the sake of simplicity. Therefore, the *New 2 VCs-B* proposal does not reduce the cost nor the power-consumption, but makes a more effective use of buffer space. For clarity reasons, we neglect in this study the small savings in silicon area and scheduler speed due to this improvement.

Finally, the *New 2 VCs-P* design would implement more ports in the chip by reducing the amount of buffer space per port. As we discussed in Section 3.1, reducing the memory per port in a factor of 4 would not translate directly into an increase of ports also in a factor of 4. The reason for this is that, although memory is the most silicon area consuming component, the crossbar and the scheduler would be more complex with more ports. Taking a conservative approach,

we can assume that we can implement twice the ports and 3 packets buffering per VC (6 Kbytes per VC, 12 Kbytes per port). The rest of the saved silicon area due to unimplemented memory would be consumed by the new (and bigger) crossbar and scheduler.

Note that the *New 2 VCs-P* switch model greatly reduces the cost and power-consumption of the network. For instance, it takes just 16 switches (67% less than the other designs) and 128 links (33% less than the other designs) to build a 64 ports MIN.

4.2. Traffic model

We will inject traffic from 8 service levels (SLs). In this way, the workload is composed of 8 different SLs: Four QoS SLs (SLs from 0 to 3) and four best-effort SLs (SLs from 4 to 7). Each SL has increasing priority, such that SL 0 has the highest priority and SL 7 has the lowest. Each interface will inject the same amount of traffic from each SL, using 90% of the network capacity.

The destination pattern of the traffic injected is based on Zipf's law [16], as recommended in [7]. In this way, the traffic is not uniformly distributed, but, instead, for each SL and input port it is established a ranking among all the possible destinations. The probability that an arriving packet is heading toward a destination with rank i is given by Equation 1, where i is the rank of packet destination, k is the Zipf order and N is the number of addresses.

$$Zipf(i) = \frac{i^{-k}}{\sum_{j=1}^{N} j^{-k}} \quad (1)$$

Therefore, there will be destinations with a higher chance of being elected by a group of flows, where this probability is obtained with the aforementioned Zipf's law. The global effect is a potential full utilization of the network, but with a reduced performance compared with a uniform distribution. Note that a value of $k = 0$ would produce an uniform destination pattern.

We have used self-similar traffic to evaluate our proposal, defining a worst-case scenario, with $k = 1$. This traffic is composed of bursts of packets heading to the same destination. The packets' sizes are governed by a Pareto distribution, as recommended in [10]. In this way, many small size packets are generated, with an occasional large size packet. The periods between bursts are modelled with a Poisson distribution. With this distribution, if the burst size is long (60 packets, approximately 20 Kbytes), there is a lot of temporal and spatial locality and should show worst-case behavior because at a given moment, many packets are grouped going to the same destination. The length of the bursts will be noted as the B parameter in the figures.

5. Simulation results

In this section, we show the performance of our proposals. We have considered three traditional QoS indices for this performance evaluation: Throughput, latency, and jitter. Note that no packets are dropped due to the use of credit-based flow control. Maximum jitter determines the receiver's user space for multimedia traffic. Inappropriate results of latency or jitter may lead to dropped packets at the application level. For this reason, we also show maximum values of these indices.

In the following, we will see two experiments. The first one uses uniform traffic and no bursts. This is controlled by the $k = 0$ and $B = 1$ parameters. We also test a worst-case scenario with $k = 1$ and $B = 60$. This will produce unbalanced and bursty traffic, with a lot of spatial and temporal locality. In both cases we test various network sizes, with an input load of 90% of network capacity.

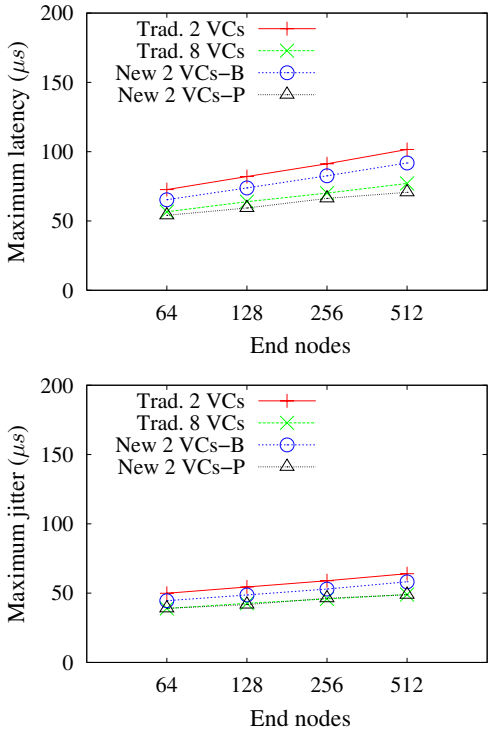

Figure 1. Performance of SL 0 traffic, uniform scenario ($k = 0$, $B = 1$).

Figure 1 shows the performance of QoS traffic under uniform traffic. The average latency results are very similar for the four architectures. In this case, the best performance corresponds to our *New 2 VCs-P* proposal, mostly because the switches in this interconnection network have more ports, which in turn leads to MINs with less stages

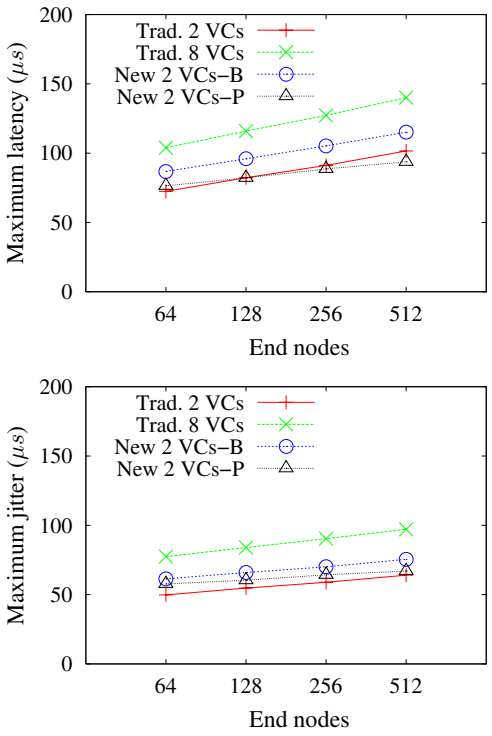

Figure 2. Performance of SL 3 traffic, uniform scenario ($k = 0$, $B = 1$).

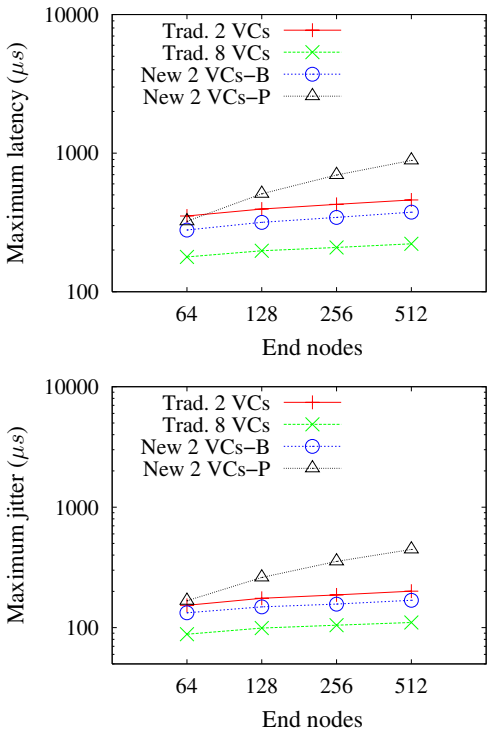

Figure 3. Performance of QoS traffic, worst-case scenario ($k = 1$, $B = 60$).

than the other alternatives.

On the other hand, we can also see the maximum values of latency and jitter. Our proposal *New 2 VCs-P* offers the best results again. We can see, thus, that with light load and/or traffic without bursts, our proposal outperforms the *Traditional 8 VCs* case due to the reduction in component count (more on this later).

When we use the worst-case scenario, we can see in Figure 3 that the results are quite different. In this case, our proposal *New 2 VCs-P* offers the worst results in terms of latency and jitter. The reason for this is the reduced size of buffers with this design. However, note that the latency results are still acceptable, the throughput is guaranteed by the CAC and, finally, this is the worst performance that you can get because the traffic we have used is very bursty. As shown in [11], when we use realistic multimedia traffic to test our proposal, the performance of this proposal is very similar to the *Traditional 8 VCs* case.

Nevertheless, these results confirm that our proposals are able to produce a good performace for QoS traffic. The *New 2 VCs-B* case does offer similar performace to the *Traditional 8 VCs* case, while the *New 2 VCs-P* proposal offers very acceptable performance and would reduce significantly the cost and power consumption of the interconnection network.

In Figure 5, we evaluate the best-effort traffic for the aforementioned network architectures, using uniform traffic. We can see that the four architectures offer good results, although there is a degradation in the performance of the traffic class with the lowest priority when using the *Traditional 8 VCs* and *New 2 VCs-P* cases. This is due to the buffers design in these cases.

If we repeat the experiment using worst-case traffic (Figure 6), the results are quite different. In this case, the *Traditional 2 VCs* approach shows the same performance for all the traffic classes, which is an inadequate behavior, because the SLs with more priority should have better performance. The reason for this behavior is that all the best-effort classes look the same for the schedulers at both the interfaces and switches in the *Traditional 2 VCs* case.

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

The *New 2 VCs-B* offers the best performance because

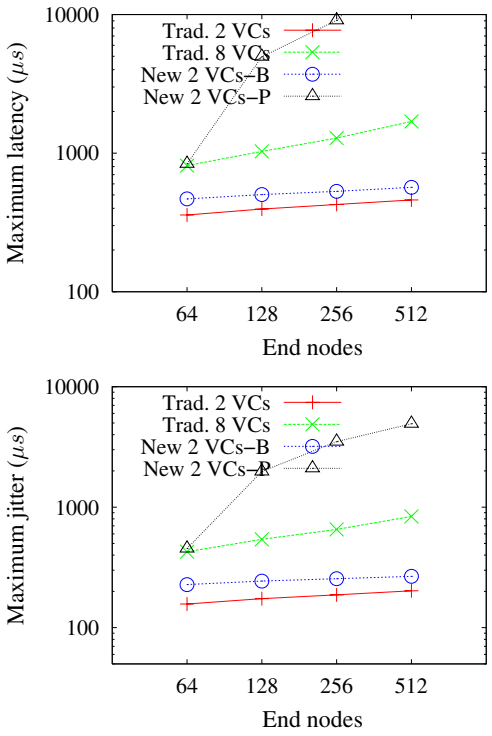

Figure 4. Performance of QoS traffic, worst-case scenario ($k = 1$, $B = 60$).

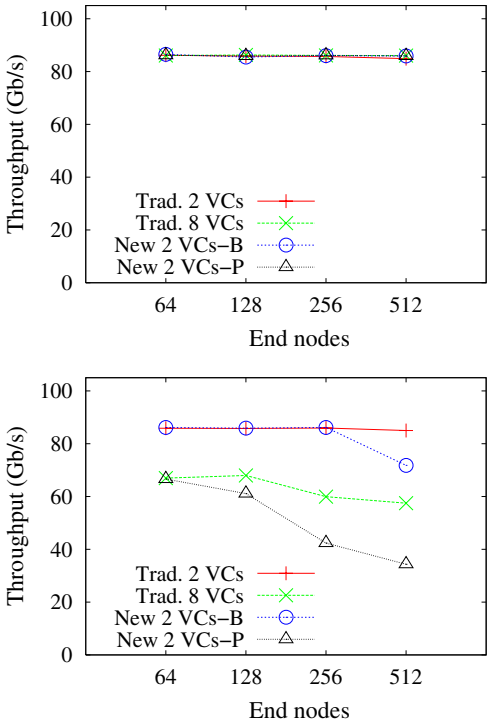

Figure 5. Throughput of SL 4 (top) and SL 5 (bottom), uniform scenario ($k = 0$, $B = 1$).

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 for the best-effort traffic with the lowest priority because it has less buffer space.

Figure 7 summarizes the trade-offs of using our *New 2 VCs-P* proposal. As can be seen, there is a very noticeable reduction in chip and links counts and therefore, in the associated power consumption of the interconnection network. On the other hand, the global throughput achieved is also lower, but the reduction on the component count would lead to a much more cost-effective solution.

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 VCs allows us to offer two alternative switch designs: The first keeps the expenses but produces a higher global throughput and better latency and jitter results by using more flexible buffers. The second greatly decreases the cost and power-consumption of the interconnection network with a small degradation in performance (around 10% less global throughput than *Traditional 8 VCs* in the worst-case scenario).

6. Conclusions

In [11] we presented a proposal to use only two VCs at each switch port to provide QoS support. The first VC of the two proposed 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. We also showed preliminary results using multimedia traffic in an uniform scenario for a small network size.

In this paper, we have evaluated this proposal's scalability and we have found that the performance is good even when network size increases. We have also evaluated two alternative switch designs benefiting from our proposal. The first design, although does not reduce the cost of the network, improves the performance under bursty traffic. Moreover, we have found that the other design, with more of ports per switch, obtains a noticeable reduction on the component count of the network. This would lead to a cheaper and less power-consuming interconnetion network.Finally, our evaluation has considered a worst-case scenario, which serves us to find the limits of our proposals. Note that this scenario is so hard that even a switch model with the full number of VCs has problems to offer good results.

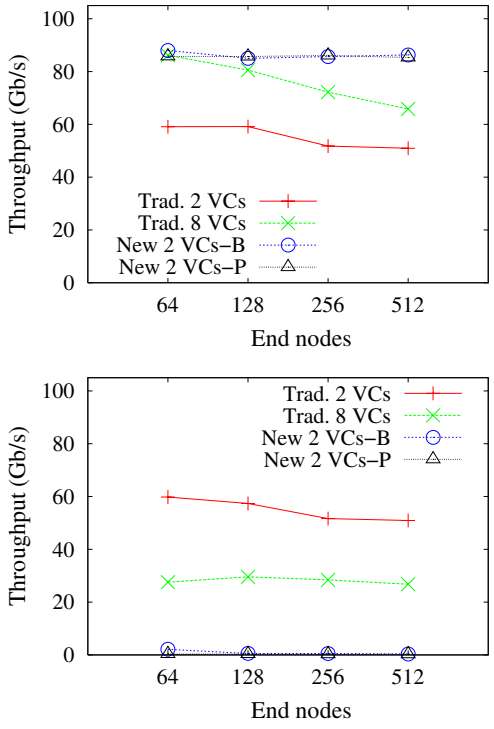

Figure 6. Throughput of SL 4 (top) and SL 5 (bottom), worst-case scenario ($k = 1$, $B = 60$).

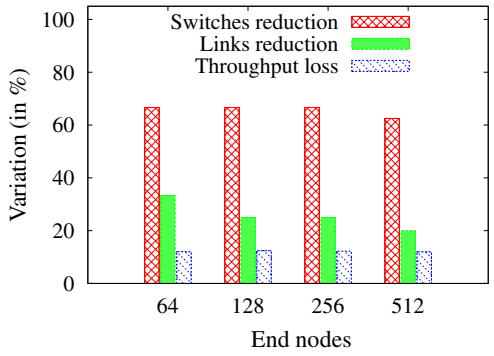

Figure 7. Summary of trade-offs of *New 2 VCs-P* proposal, compared with *Traditional 8 VCs* case.

References

[1] Advanced switching core architecture specification. Technical report, available online at `http://www.asi-sig.org/specifications` for ASI SIG members.

[2] F. J. Alfaro, J. L. Sánchez, and J. Duato. QoS in InfiniBand subnetworks. *IEEE Transactions on Parallel Distributed Systems*, 15(9):810–823, Sept. 2004.

[3] N. Chrysos and M. Katevenis. Multiple priorities in a two-lane buffered crossbar. In *Proceedings of the IEEE Globecom 2004 Conference*, Nov. 2004.

[4] W. Dally, P. Carvey, and L. Dennison. Architecture of the Avici terabit switch/router. In *Proceedings of the 6th Symposium on Hot Interconnects*, 1998.

[5] J. Duato, S. Yalamanchili, M. B. Caminero, D. Love, and F. Quiles. MMR: A high-performance multimedia router. Architecture and design trade-offs. In *Proceedings of the 11th Symposium on High Performance Computer Architecture (HPCA)*, Jan. 1999.

[6] J. Duato, S. Yalamanchili, and N. Lionel. *Interconnection networks. An engineering approach*. Morgan Kaufmann Publishers Inc., 2002.

[7] I. Elhanany, D. Chiou, V. Tabatabaee, R. Noro, and A. Poursepanj. The network processing forum switch fabric benchmark specifications: An overview. *IEEE Network*, Mar. 2005.

[8] C. Huang, J. Wang, and Y. Huang. Design of high-performance CMOS priority encoders and incrementer/decrementers using multilevel lookahead and multilevel folding techniques. *IEEE Journal of Solid-State Circuits*, 1(37):63–76, Jan. 2002.

[9] InfiniBand Trade Association. *InfiniBand architecture specification volume 1. Release 1.0*, Oct. 2000.

[10] R. Jain. *The art of computer system performance analysis: techniques for experimental design, measurement, simulation and modeling*. John Wiley and Sons, Inc., 1991.

[11] A. Martínez, F. J. Alfaro, J. L. Sánchez, and J. Duato. 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)*, Dec. 2005. Available at `http://www.i3a.uclm.es/documentos/2/congresos/Congreso2_134_HiPC05.pdf`.

[12] C. Minkenberg, F. Abel, M. Gusat, R. P. Luijten, and W. Denzel. Current issues in packet switch design. In *ACM SIGCOMM Computer Communication Review*, Jan. 2003.

[13] D. Miras. A survey on network QoS needs of advanced internet applications. Technical report, Internet2 - QoS Working Group, 2002.

[14] L. Shang, L. S. Peh, and N. K. Jha. Dynamic voltage scaling with links for power optimization of interconnection networks. In *Proceedings of the 9th Symposium on High Performance Computer Architecture (HPCA)*, pages 91–102, February 2003.

[15] D. Simos. Design of a 32x32 variable-packet-size buffered crossbar switch chip. Technical Report FORTH-ICS/TR-339, Inst. of Computer Science, FORTH, July 2004. `http://archvlsi.ics.forth.gr/bufxbar/`.

[16] G. K. Zipf. *The Psycho-biology of Languages*. Houghton-Miffin, MIT, 1965.