Towards a Cost-Effective Interconnection Network Architecture with QoS and Congestion Management Support*

A. Martínez[1], P. J. García[1], F. J. Alfaro[1], J. L. Sánchez[1], J. Flich[2], F. J. Quiles[1], and J. Duato[2]

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

Abstract. Congestion management and quality of service (QoS) provision are two important issues in current network design. The most popular techniques proposed for both issues require the existence of specific resources in the interconnection network, usually a high number of separate queues at switch ports. Therefore, the implementation of these techniques is expensive or even infeasible. However, two novel, efficient, and cost-effective techniques for provision of QoS and for congestion management have been proposed recently. In this paper, we combine those techniques to build a single interconnection network architecture, providing an excellent performance while reducing the number of required resources.

1 Introduction

High-speed interconnection networks have become a major issue on the design of several computing and communication systems, including systems for parallel computing since they provide the low-latency and high-performance demanded by parallel applications. Unfortunately, the network is also becoming the most expensive and power consuming part of these systems.

On the other hand, networks have been traditionally overdimensioned in order to avoid high link utilization, but currently this is an expensive practice. Therefore, new and clever solutions for the problems related to high link utilization are needed.

One of these problems is network congestion. If not managed, congestion dramatically degrades network performance because it leads to blocked packets[3]

* This work was partly supported by the Spanish CICYT under grant TIC2003-08154-C06, by Junta de Comunidades de Castilla-La Mancha under grant PBC-05-005, by the Spanish State Secretariat of Education and Universities under FPU grant, and by UPV under Grant 20040937.

[3] We are considering lossless networks like InfiniBand, Quadrics, or Myrinet.

that prevent the advance of other packets stored in the same queue, even if they are requesting free resources further ahead. Moreover, a high utilization of the links may also degrade the performance observed by the users, which leads to the necessity of techniques to provide the traffic with quality of service (QoS). In this case, it is necessary to avoid interferences from best-effort traffic, which only demands a "deliver when possible" service, and guarantee that traffic with strict requirements is properly served.

Many techniques have been proposed both for provision of QoS and for congestion management. Unfortunately, most of them rely on the use of a considerable number of queues at the switches. For instance, the use of virtual output queues has been proposed for handling congestion, but it requires as many queues per switch port as end-points in the network. This increases switch cost due to the silicon area required for implementing such number of buffers.

Regarding QoS provision, the use of virtual channels (VCs) is a common solution, but, although current interconnect standards propose 16 or even more VCs, most commercial components do not offer so many VCs because it is too expensive in terms of silicon area. In fact, the trend followed nowadays by interconnect manufacturers in their new products is to increase the number of switch ports instead of increasing the number of VCs per port [1]. Note that for high-speed, single-chip switches, proposals requiring many queues could be considered if external DRAM is available for implementing the buffers. However, in this case, the low latencies demanded by QoS-requiring traffic could not be provided.

Recently, two novel, efficient, and cost-effective techniques both for provision of QoS and for congestion management have been proposed. The first one [2] consists in a full QoS support with only two VCs. The RECN [3, 4] congestion management strategy is the second one. The implementation of both proposals requires a very small number of queues per port, while they offer the same effectiveness as other more silicon-requiring techniques.

In this paper, we study in detail how these two techniques can be combined to form a single architecture that uses a reduced number of queues per port. We also show the performance achieved by the resulting switch architecture in comparison with the performance reached by more expensive solutions.

The rest of this paper is structured as follows. In Section 2, we review the techniques proposed for congestion management and, in particular, the RECN mechanism. Next, in Section 3 we review the proposals for QoS support in interconnection networks and specially the proposal that uses only two VCs. Section 4 presents the proposed interconnection network architecture, whose performance evaluation is presented in Section 5. Finally, in Section 6 some conclusions are drawn.

2 Dealing with Congestion in Interconnection Networks

The risk of congestion in interconnection networks is a well-known problem, and many strategies have been proposed to deal with it. The simplest of those strate-

gies are the network overdimensioning and the dropping of packets in congestion situations. However, none of them are suitable for modern interconnection networks: Overdimensioning the network implies a high cost and power consumption, while the dropping of packets implies packet retransmission that increases packet latency, so current interconnection networks are usually lossless.

Other more elaborated techniques have been specifically proposed for avoiding or eliminating congestion. For instance, proactive strategies are based on reserving network resources for each data transmission, requiring a traffic planification based on network status [5]. However, this status information is not always available, and the resource reservation procedure introduces significant overhead. On the other hand, reactive congestion management is based on notifying congestion to the sources contributing to its formation, in order to cease or reduce the traffic injection from those sources [6]. Unfortunately, these solutions are not quite efficient due to the delay between congestion detection and notification.

Other congestion management strategies focus on eliminating the main negative effect of congestion: The head-of-line (HOL) blocking. This phenomenon happens when a blocked packet at the head of a FIFO queue prevents the advance of other packets at the same queue, even if those packets require available resources. This effect may degrade network performance dramatically, since data flows not contributing to congestion may advance at the same speed than congested flows. In fact, an effective HOL blocking elimination would turn congestion harmless. In that sense, many HOL blocking elimination strategies have been proposed: virtual output queues (VOQs) [7], dynamically allocated multiqueues (DAMQs) [8], congestion buffers [9], etc. Most of these techniques rely on allocating different buffers for storing separately packets belonging to different flows.

In general, traditional HOL blocking elimination techniques are scalable or efficient, but not scalable and efficient at the same time. For instance, the use of VOQs at network level requires as many queues at each port as end-points in the network, being so an effective but not scalable technique. A variation of VOQ uses as many queues at each port as output ports in a switch [10]. So, this technique is scalable, but it does not eliminate completely HOL blocking, only the switch's internal HOL-blocking.

Recently, a new HOL blocking elimination technique has been proposed: RECN [3, 4]. RECN eliminates HOL blocking in a scalable and efficient way.

2.1 RECN description

RECN (Regional Explicit Congestion Notification)[3] is a congestion management strategy that focuses on eliminating HOL blocking. In order to achieve it, RECN detects congestion and dynamically allocates separate buffers for each congested flow, assuming that packets from non-congested flows can be mixed in the same buffer without producing significant HOL blocking. Therefore, maximum performance is achieved even in the presence of congestion.

RECN requires the use of some source deterministic routing in order to address a particular network point from any other point in the network. In fact, RECN has been designed for PCI Express Advanced Switching (AS) [11], a technology that uses source routing[4]. AS packet headers include a turnpool made up of 31 bits, which contains all the turns (offset from the input port to the output port) for every switch in a route. Thus, a switch, by inspecting the appropriate turnpool bits, can know in advance if a packet that is coming through one of its input ports will pass through a particular network point.

In order to separate congested and non-congested flows, RECN adds a set of additional queues (set aside queues, SAQs) to the standard queue at every input and output port of a switch. While standard queues will store non-congested packets, SAQs are dynamically allocated and used to store packets passing through a congested point. Every set of SAQs is controlled by means of a CAM (Content Addressable Memory). Every CAM line contains information required for identifying a congested point and for managing the associated SAQ.

Whenever an input or output standard queue receives a packet and fills over a given threshold, RECN detects congestion[5]. Then, a congestion notification is sent upstream to the packet sender port (an input port of the same switch or an output port of an upstream switch). These notifications include the turnpool required to reach the congested point from the notified port. Upon reception of a notification, a port allocates a new SAQ and fills the corresponding CAM line with the received turnpool. Since that moment, every packet received in this port will be stored in the allocated SAQ if it will pass through the associated congested point (this can be deduced from the packet turnpool). As non-congested packets are stored in different queues (the standard ones), congested packets cannot cause HOL blocking.

Furthermore, if any SAQ becomes congested, another notification will be sent upstream, and the receiving port should allocate a new SAQ. This procedure can be repeated until the notifications reach the sources. Therefore, there will be SAQs for storing congested packets at every point where otherwise these packets could produce HOL blocking. Moreover, congested packets cannot fill the port memory completely as RECN uses a SAQ-specific Xon/Xoff flow control.

RECN also detects congestion vanishment at any point, in such a way that the SAQs assigned to this point can be deallocated and later re-allocated for new congested points. This allows RECN to eliminate HOL blocking while using a reduced number of SAQs. Further details about RECN can be found in [3, 4].

3 QoS Support in Interconnection Networks

During the last decade, several switch architectures with QoS support have been proposed. Among the most recent proposals are the industry standards Infini-

[4] However, note that RECN could be applied on any network technology if it allows the use of source deterministic routing.

[5] Actually, in order to detect at input ports which output port is congested, RECN divides each standard queue into several small detection queues [4].

Band and PCI Express Advanced Switching (AS). The InfiniBand standard [12] considers up to 16 VCs, while the AS specification [11] incorporates up to 20 VCs (16 unicast, 4 multicast). However, the implementation of such number of VCs would require a significant fraction of silicon area and it would make packet processing a time-consuming task. Consequently, as far as we know, no implementation of these standards includes the full number of proposed VCs.

Several proposals have been presented in order to reduce the hardware required for QoS provision, some of them using only two VCs. For instance, the Avici TSR [7] is able to segregate premium traffic from regular traffic. However, it is limited to this classification and cannot consider more categories. Also, the architecture proposed in [13] maps multiple priority levels onto two queues. However, this proposal is aimed at a single-stage router based on a single buffered crossbar with small buffers at the crosspoints that are split into two VCs.

In contrast, the technique proposed in [2] uses two VCs while being simpler and more generic, as it is shown in the next section.

3.1 Full QoS support with 2 VCs

The key idea of the proposal explained in [2], is quite simple: Assuming that the links are not oversubscribed, all the traffic flows through the switches seamlessly. Therefore, it is possible to use only two VCs at the switch ports. One of these VCs is used for QoS packets and the other one for best-effort packets. In [2], a connection admission control (CAC) is used to guarantee that QoS traffic will not oversubscribe the links.

Another cornerstone of this proposal is to reuse at the switches the scheduling decisions taken at the network interfaces regarding the injection of traffic from the different classes. Specifically, it is assumed that packets are ordered at network interfaces according to a static priority criterion. In this way, every packet would be stamped with a priority (or service) 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 must merge these packet flows at the output ports. The 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.

Although it is assumed that QoS traffic does not oversubscribe any link, no assumption is made about best-effort traffic. However, network interfaces are still able to assign the available bandwidth (the fraction not consumed by QoS traffic) to best-effort traffic in the configured proportions. In this way, switches can still take into account the modest QoS requirements of this kind of traffic. Obviously, this is a coarse-grain QoS provision.

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

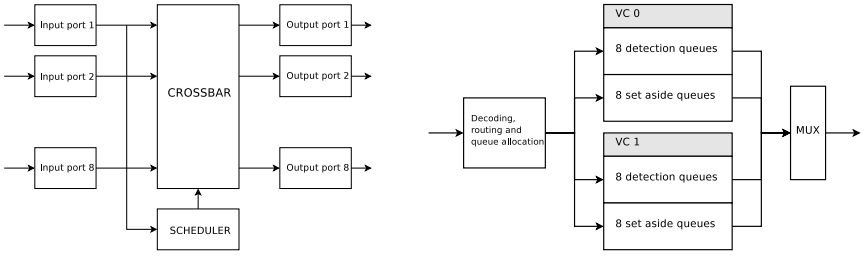
Note also that some aspects of this proposal could be simplified or improved if it is combined with the RECN strategy. For instance, instead of the CAC

applied to QoS traffic in [2], the RECN mechanism could detect if some traffic flows start producing congestion, and immediately segregate these flows from the non-congested ones. Moreover, regarding best-effort traffic, RECN can make important contributions. Specifically, it can guarantee the maximum throughput for best-effort traffic, avoiding also that congested flows affect non-congested traffic.

As RECN also requires a reduced number of resources, the combination of both techniques would allow to provide effective QoS support and congestion management at a low cost in terms of silicon area. The architecture we propose for combining both techniques is explained in the following section.

4 Proposed Interconnection Network Architecture

Our proposal consists in a interconnection network architecture able to support QoS and to cope with congestion while requiring reduced resources. The strategies we propose affect both network interfaces and switches.



(a) Switch organization (b) Input port organization

Fig. 1. Proposed architecture.

Figure 1 (a) shows a logical view of the switch organization, which consists in a combination of input and output buffering. Note that all the switch components are intended to be implemented in a single chip. This is necessary in order to offer the low cut-through latencies demanded by current parallel applications.

The innovations of our proposal are in the port design. The organization of an input port can be seen at Figure 1 (b). There are only two VCs: VC 0 is intended for QoS traffic, while VC 1 is intended for best-effort traffic. As can be seen, each VC is further divided into 16 queues. The first 8 queues of each VC are the detection queues, so each queue corresponds to each switch output port. The next 8 queues of each VC are the SAQs, where congested traffic is stored.

We include an additional field in the CAM lines used for managing the SAQs: The service level (SL). This new field will be used, in addition to the turnpool, for assigning a SAQ to a specific point and to a specific SL. Therefore:

– When a standard queue reaches the detection threshold, the corresponding congestion notification includes now, in addition to the turnpool, the SL of the packet responsible of the detection.
– Each allocated SAQ contains traffic of a single SL.
– A single turnpool may be replicated for several SAQs, with different SLs.

So, the congestion detection process is slightly different than in RECN without QoS support. As a detection queue may reserve several SAQs (each for a different SL), a bit mask is required to control which SLs have reserved a SAQ.

The output ports of the switch replicate the structure of the inputs, with two main differences. There is no need to decode the messages and detection queues refer to the outputs of the next switch. The network interface design would be very similar to this, although, in this case, a VC exists for each SL.

The scheduling at the switches goes as follows. There is a strict precedence of VC 0 (QoS traffic) over VC 1 (best-effort traffic): As long as there are ready packets of VC 0, no one from VC 1 is eligible. Among the queues inside each VC, a simple round-robin algorithm is applied. Note that RECN ensures that none of the SAQs will occupy all the buffer space and, therefore, this simple scheduling is sufficient.

The area requirements study for this design is based on the process detailed at [14]. We do not have yet a detailed Verilog switch design, but we can obtain good estimations by considering the area consumption of each individual component.

Table 1. Area consumption by components.

Module	Area 0.18 μm	Area 0.13 μm
Buffers ($32 \times 16 Kbytes$)	64 mm^2	32 mm^2
Crossbar and datapath	10 mm^2	5 mm^2
Scheduler	5 mm^2	3 mm^2
Total	79 mm^2	40 mm^2

In Table 1, the aforementioned estimations can be found. The memory area consumptions are taken from memory datasheets [15]. The number of buffers in the switch comes from 8 ports $\times$ 2 VCs $\times$ input and output. Keep in mind that at the placement and routing phase of the design process, the wiring introduced could increase these figures. Therefore, this design would take 100-150 mm^2 using 180 nanometers technology.

5 Performance Evaluation

We have evaluated the proposed architecture by means of simulations. In this section, we will detail the simulated scenarios and we will offer results showing the behavior of our proposal in comparison with the one of traditional switches.

5.1 Simulated architecture

We have supposed a workload of 8 SLs, with decreasing priority, such that SL 0 has the highest priority and SL 7 has the lowest. We have also assumed that SLs from 0 to 3 are QoS-requiring traffic, and share the same VC in the two-VC architecture. Moreover, we also suppose that SLs from 4 to 7 are best-effort traffic, and share the other VC in the two-VC scheme.

We have run simulations for three architectures. First, we have tested the performance of the ideal architecture, using VOQ at the network level combined with a VC per SL (*VOQ Net*). Also, we have tested a more realistic architecture, using VOQ at the switch level and as many VCs as SLs (*VOQ Switch*). Finally, our proposed architecture, combining the use of only 2 VCs at the switches with the use of RECN (*RECN 2 VCs*).

The network used in the tests is a folded (bidirectional) butterfly multi-stage interconnection network (MIN) with 128 ports (3 stages). We have chosen a MIN because it is a usual topology for computer clusters and IP routers. However, our proposal is valid for any other network topology, including direct networks. Other assumptions are based on the AS specifications [11]. For instance, maximum packet size is 2 Kbytes and link bandwidth is 8 Gb/s. Another assumption is the use of source routing, since it is needed by RECN.

The *VOQ Switch* and *RECN 2 VCs* architectures consider 8 ports and 32 Kbytes of buffer space per port. Note that the buffer space per VC in the *RECN 2 VCs* case is bigger than in the *VOQ Switch* case in a factor of 4. Also note that the scheduler considers 64 queues per port in the *VOQ Switch* design (8 VCs $\times$ 8 VOQs), while in the *RECN 2 VCs* case 32 queues (8+8 from each VC) are considered. For the sake of clarity, we do not consider in this study the saving in silicon area and the gain in scheduler speed due to these facts.

The *VOQ Net* architecture assumes a space of 2 maximum size packets per VC. This is the minimum required to assure a full throughput under a full load between two ports (one packet size plus a round-trip time, rounded to full packets). In a 128 end-points network, it requires 128 $\times$ 8 VCs $\times$ 4 Kbytes = 4 Mbytes per port of buffering. It is clearly too much for a single chip, and it should be implemented in external DRAM, but in that case the cut-through latency of the switch would be much higher. Nevertheless, in order to provide a reference of the ideal performance, we have not considered this additional delay in the *VOQ Net* simulations.

5.2 Traffic model

In all the tests we have used self-similar traffic. 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 [16]. 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 and the distribution of the bursts destinations is uniform. If the burst size is long, 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. Regarding burst length, we have used a long one of 30 Kbytes for the four best-effort classes and a shorter one of 5 Kbytes for the QoS classes.

5.3 Simulation results

We have considered two traditional QoS metrics in the performance evaluation: Throughput and latency. Packet loss is not considered because no packets are dropped due to the use of credit-based flow control. However, note that inappropriate results of latency may lead to dropped packets at the application level. For this reason, we also consider maximum latency.

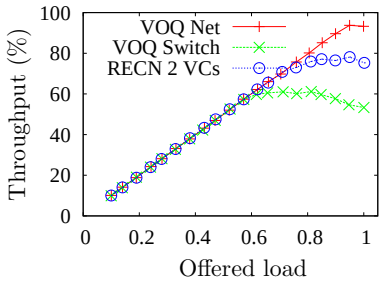


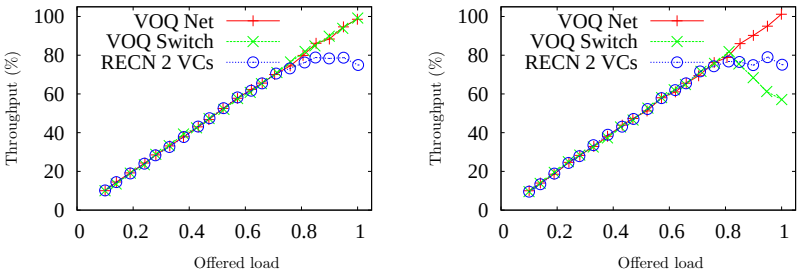
Fig. 2. Global throughput results for best-effort SLs.

We first analyze the results for the best-effort traffic classes, which are more likely to suffer from congestion. Figure 2 shows the global throughput of the network for the unregulated traffic. It can be seen that our *RECN 2 VCs* proposal, from an input load of 80% on, offers a 25% improvement over the *VOQ Switch* architecture and only losses a 5% from the ideal, infeasible *VOQ Net* architecture.

Figure 3 depicts the detailed throughput results for each one of the best-effort classes (SLs 4 to 7). While our proposal offers results very close to those of the ideal architecture, *VOQ Switch* offers a very poor performance for SLs 6 and 7.

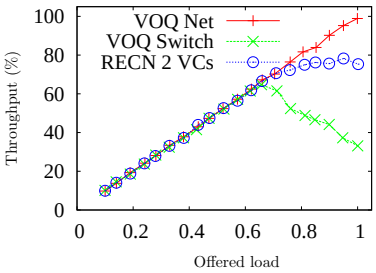
The next part of the evaluation deals with the QoS traffic. In this case, the three architectures offer 100% throughput for QoS traffic (not shown). Figure 4 shows the interesting average latency results for the QoS SLs. Note that these results are very similar for all the architectures. Although our proposal offers slightly worse results for the SL 0, the SL 3 benefits from the RECN technique. Almost identical results have been obtained for maximum latency (Figure 5). Due to space constraints maximum jitter results are not shown, but are also similar to those of latency.

These results show that QoS-requiring flows get the performance they need when using our architecture, although they are sharing a single VC. Therefore, our proposal is able to offer QoS support at the same level as a proposal that

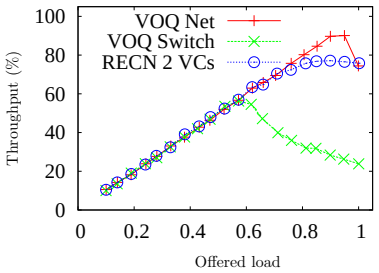


(a) SL 4

(b) SL 5



(c) SL 6



(d) SL 7

Fig. 3. Detailed throughput results for best-effort SLs.

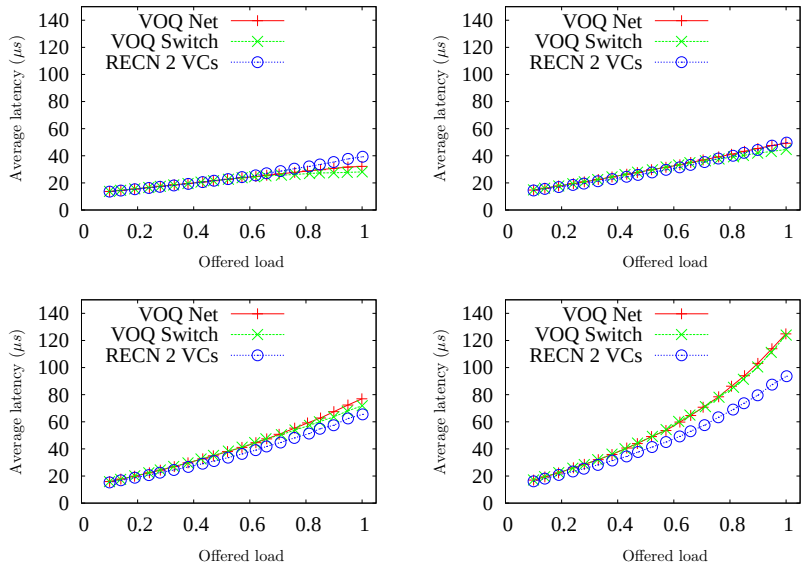
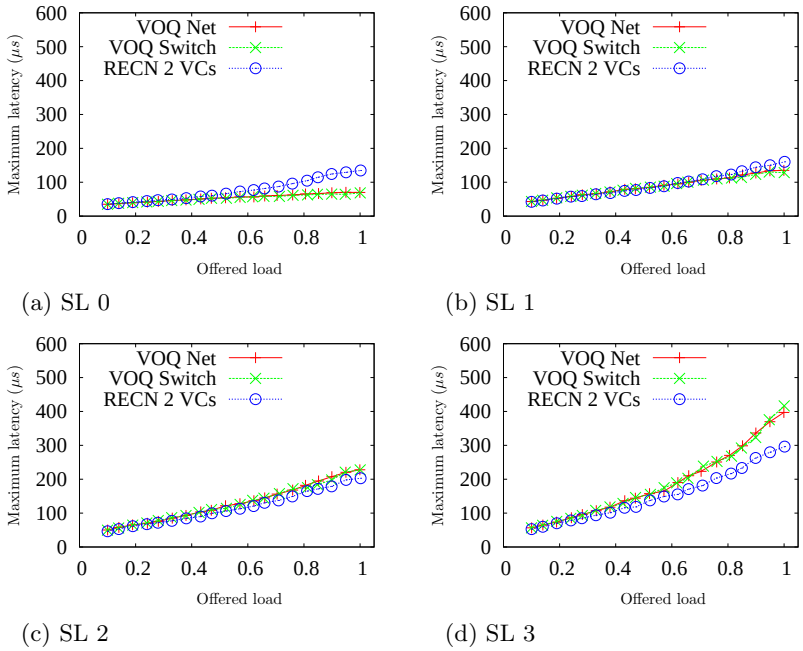


Fig. 4. Average latency results for QoS SLs.



(a) SL 0 (b) SL 1

(c) SL 2 (d) SL 3

Fig. 5. Maximum latency results for QoS SLs.

doubles the required number of queues (*VOQ Switch*) and even at the same level as an ideal and expensive architecture (*VOQ Net*). Moreover, in the case of heavy congestion in the QoS SLs, the *VOQ Switch* case will suffer strong degradation while the RECN mechanism included in our architecture would solve the problem.

6 Conclusions

Due to cost and power consumption constraints, current high-speed interconnection networks cannot be overdimensioned. Therefore, some solutions are needed in order to handle the problems related to high link utilization. In particular, both QoS support and congestion management techniques have become essential for achieving good network performance. However, most of the techniques proposed for both issues require too many resources for being implemented.

In this paper we propose a new network architecture able to face the challenges of congestion management and, at the same time, QoS provision, while being more cost-effective than other proposals. Our proposal is based on the combination of two novel techniques that provide congestion control and QoS support while requiring a reduced number of resources.

According to the results presented in this paper, we can conclude that our proposal can provide an adequate QoS while properly dealing with congestion.

We provide advanced techniques for the buffer management, which allow a good performance under heavy and unbalanced load, while still providing appropriate QoS levels. Since all this is achieved with a reduced number of resources, this architecture would also reduce network cost.

References

1. Minkenberg, C., Abel, F., Gusat, M., Luijten, R.P., Denzel, W.: Current issues in packet switch design. In: ACM SIGCOMM Computer Communication Review. (2003)
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) Available at `http://www.i3a.uclm.es/documentos/2/congresos/Congreso2_134_HiPC05.pdf`.
3. Duato, J., Johnson, I., Flich, J., Naven, F., García, P., Nachiondo, T.: A new scalable and cost-effective congestion management strategy for lossless multistage interconnection networks. In: Proceedings of the 11th Symposium on High Performance Computer Architecture (HPCA). (2005)
4. García, P., Flich, J., Duato, J., Johnson, I., Quiles, F., Naven, F.: Dynamic evolution of congestion trees: Analysis and impact on switch architecture. Lecture Notes in Computer Science (HiPEAC 2005) **3793** (2005) 266–285
5. Wang, M., Siegel, H.J., Nichols, M.A., Abraham, S.: Using a multipath network for reducing the effects of hot spots. IEEE Transactions on Parallel and Distributed Systems **6** (1995) 252–268
6. Thottetodi, M., Lebeck, A., Mukherjee, S.: Self-tuned congestion control for multiprocessor networks. In: Proc. of 7th. Int. Symp. on High Performance Computer Architecture. (2001)
7. Dally, W., Carvey, P., Dennison, L.: Architecture of the Avici terabit switch/router. In: Proceedings of the 6th Symposium on Hot Interconnects. (1998)
8. Tamir, Y., Frazier, G.: Dynamically-allocated multi-queue buffers for vlsi communication switches. IEEE Transactions on Computers **41** (1992)
9. Smai, A., Thorelli, L.: Global reactive congestion control in multicomputer networks. In: Proc. 5th Int. Conference on High Performance Computing. (1998)
10. Anderson, T., Owicki, S., Saxe, J., Thacker, C.: High-speed switch scheduling for local-area networks. ACM Transactions on Computer Systems **11** (1993) 319–352
11. Advanced Switching Interconnect Special Interest Group: Advanced Switching Core Architecture Specification. Revision 1.1. (2005)
12. InfiniBand Trade Association: InfiniBand architecture specification volume 1. Release 1.0. (2000)
13. Chrysos, N., Katevenis, M.: Multiple priorities in a two-lane buffered crossbar. In: Proceedings of the IEEE Globecom 2004 Conference. (2004)
14. 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) `http://archvlsi.ics.forth.gr/bufxbar/`.
15. Virtual Silicon Technology, Inc.: eSi-RAM/2P^{tm} Two-port register file SRAM. Data Sheet (2004)
16. Jain, R.: The art of computer system performance analysis: techniques for experimental design, measurement, simulation and modeling. John Wiley and Sons, Inc. (1991)