University of Castilla-La Mancha

A publication of the

Department of Computer Science

MMR: A MultiMedia Router Architecture to Support Integrated Workloads

by

Blanca Caminero, Carmen Carrión, Francisco J. Quiles,

José Duato, Sudhakar Yalamanchili

Technical Report **#DIAB-02-02-28** October 2002

A shorter version of this paper has been submitted to the *IEEE Transactions on Parallel and Distributed Systems*.
José Duato is with the Department of Information Systems and Computer Architecture (DISCA), Universidad Politécnica de Valencia.
Sudhakar Yalamanchili is with the School of Electrical and Computer Engineering, Georgia Institute of Technology (USA).

DEPARTAMENTO DE INFORMÁTICA
ESCUELA POLITÉCNICA SUPERIOR
UNIVERSIDAD DE CASTILLA-LA MANCHA
Campus Universitario s/n
Albacete - 02071 - Spain
Phone +34.967.599200, Fax +34.967.599224

MMR: A MultiMedia Router Architecture to Support Integrated Workloads

Blanca Caminero, Carmen Carrión, Francisco J. Quiles
Dept. of Computer Science. Escuela Politécnica Superior.
Universidad de Castilla-La Mancha. 02071 - Albacete, SPAIN
{blanca, carmen, paco}@info-ab.uclm.es

José Duato
Dept. of Information Systems and Computer Architecture (DISCA)
Univ. Politécnica de Valencia. 46071 - Valencia, SPAIN
jduato@gap.upv.es

Sudhakar Yalamanchili
School of Electrical and Computer Engineering.
Georgia Institute of Technology. Atlanta, 30332-0250 Georgia, USA
sudha@ece.gatech.edu

Abstract

During the past few years, the dramatic increase in performance/price experienced by desktop computers has boosted the use of multimedia applications. Many of these applications are inherently distributed, while others need the resources of a network of workstations to achieve good performance. In any case, multimedia applications need the network to provide Quality of Service (QoS) guarantees to support them properly.

The MultiMedia Router architecture arises as a solution that tries to provide adequate QoS support to multimedia data flows within compact network switching elements in local and cluster area environments. Also, conventional best-effort traffic should be taken into account, but it should not interfere in the QoS guarantees received by multimedia flows. In this paper, the MultiMedia Router architecture is described and the most relevant design trade-offs are discussed. Also, the main design parameters are tuned by means of extensive simulation tests.

Index Terms

Router architecture, Quality of Service (QoS), multimedia transmissions, Clusters of Workstations (COWs), Local Area Networks (LANs)

This work was partially supported by the Spanish CICYT under Grant TIC2000-1151-C07, and by the Consejería de Ciencia y Tecnología of the Junta de Comunidades de Castilla-La Mancha (regional government). The first author is in charge of correspondence.

I. Introduction and Motivation

The cost of both processor and network components has been continuously decreasing over the past few years. This has led to a tighter integration of computation and communication. As a consequence, an explosion of network-based applications, which are characterized by the processing and delivery of continuous data streams and dynamic media [1][2] in addition to the handling of traditional static data, has taken place. Numerous examples can be drawn from web-based applications, interactive simulations, gaming, visualization, virtual meetings, and collaborative design environments [3].

The changing nature of the workload and cost/performance trade-offs have prompted the development of a new generation of scalable media servers structured around networks of workstations (NOWs) interconnected by high-speed system/local area network (SAN/LAN) fabrics. Systems such as real-time media servers need to service hundreds, and possibly thousands, of clients. Typically, each client has its own distinct quality of service (QoS) requirements, such as packet dropping vs. reliable message transmission, low latency vs. jitter, or throughput vs. latency. Diverse service requirements must be concurrently met with the same set of hardware and software resources.

The need for higher communication bandwidth is becoming especially important in local and storage area environments. There are certain kinds of applications, like 3-D image rendering, that need the storage capacity or the computing power of a cluster of workstations in order to achieve good performance, thus increasing the need for higher communication bandwidth [4]. Moreover, the astonishing growth of the Internet in the last few years, and the increasing request for multimedia information, has caused the need for more powerful and sophisticated multimedia servers, usually implemented as clusters of workstations, or even clusters of processing elements together with storage elements [5]. This bandwidth must be provided over a network shared by multiple applications, i.e., not only multimedia flows but also traditional best-effort traffic generated by conventional applications.

Traditional router technology developed for high-speed multiprocessor interconnection networks or high-performance local area networks (LANs) are optimized for low latency for best-effort traffic [6][7]. These networks are not designed to permit concurrent guarantees of communication performance for multiple applications.

The recently proposed InfiniBand^{SM} architecture [8][9], an attempt for high performance LAN/SAN interconnection devised by a consortium of computing industry's leaders such as Compaq, Dell, Hewlett-Packard, IBM, Intel, Microsoft, Sun Microsystems and others alike, provides some sort of QoS capabilities within its switches. This is done by using a weighted round-robin virtual lane scheduling algorithm with two levels of priority, and providing different virtual lanes to different "service levels". Several studies are currently being carried out in order to provide proper support for the multimedia flows within the InfiniBand framework [10][11][12].

Besides InfiniBand switches, the only known approach to providing QoS with a simple router aimed at local environments is the MediaWorm Router [13]. This router uses wormhole switching for both multimedia and best-effort traffic, and uses Virtual Clock [14] for bandwidth allocation among the connections. The use of wormhole switching has several important drawbacks when applied to multimedia environments. First of all, the time a message can be blocked waiting for a busy resource to be released is not bounded, and this can produce high delay variations, i. e., high jitter values. And second, no admission control is performed and thus, bandwidth oversubscription can be produced. This will likely push the network workload beyond saturation, making unfeasible the provision of QoS guarantees.

On the other hand, ATM provides support for quality of service (QoS), but it has been optimized for wide area networks (WANs). As a consequence, most of the solutions adopted in ATM switches are not valid for LAN environments. In particular, ATM switches require large buffers to provide QoS guarantees. The reasons for using large buffers are twofold: 1) ATM does not implement flow control; therefore, large buffers are required to minimize data losses; and 2) ATM switches for WANs usually support thousands of connections concurrently. The only known compact implementation of ATM in a LAN environment is ATLAS I [15].

The main goal pursued by the MultiMedia Router (MMR) project [16][17] is to design compact single-chip routers able to support a large number of multimedia connections while allocating the remaining bandwidth to best-effort traffic. The MMR should handle this hybrid traffic efficiently, satisfying the QoS requirements of multimedia traffic, minimizing the average latency of best-effort traffic, and maximizing link utilization until network reaches saturation.

In order to fulfill these objectives, the designer has to carefully select several qualitative and quantitative network design parameters. Qualitative parameters include network topology, routing algorithm, admission control strategy, switching technique, bandwidth allocation algorithm, link/switch scheduling algorithm, buffer organization, buffer management, and crossbar organization. Quantitative parameters include network size, link bandwidth, router degree (number of ports), router clock frequency, buffer size, and number of virtual channels. In this work, the main trade-offs involved in the design of the MultiMedia Router will be discussed, and some evaluation results regarding the tuning of a few quantitative parameters will be presented.

The paper is organized as follows. First of all, the main distinguishing characteristics of the traffic generated by multimedia applications will be described. Next, the MultiMedia Router architecture will be explained by discussing its main features. Some algorithms used in resource scheduling within the proposed architecture will be introduced then. The evaluation results corresponding to the tuning of some key design parameters will follow. And finally, some conclusions and final remarks will be given.

II. Application Requirements

The traffic generated by multimedia applications exhibits a different behavior from other traditional applications, such as parallel computing, real-time computing, remote servers, and so on. The main distinguishing features of multimedia communication environments can be summarized as follows:

- **Very long data streams** Audio and video transmissions are usually composed of very long data flows, whose duration lasts from a few seconds to several hours. These data flows are best transmitted by using a connection oriented scheme.
- **Wide range of bandwidth requirements** Different connections can have very different bandwidth necessities. For example, video transmissions require much more bandwidth than audio transmissions. What is more, these requirements might vary throughout all the life of the connection. Audio and video transmissions need to transmit information frames at regular intervals, generating a bursty traffic pattern. Besides, video flows are usually compressed in order to reduce their bandwidth needs. As the compression rate depends on the input data, bandwidth requirements vary over time, thus generating VBR (Variable Bit Rate) traffic.
- **Large number of connections** Several multimedia applications might be simultaneously running at every network node. Besides, each of these applications might require the transmis-

sion of several data flows. As QoS requirements can be best satisfied if a connection oriented scheme is used, a large number of concurrent connections should be supported. However, this number is much lower than the one that WAN (Wide Area Network) routers, oriented to voice connections, must support. For a LAN environment, this number is around several hundreds of connections.

- **Jitter sensitive** Multimedia traffic is quite sensitive to variations in latency (jitter), i. e., video frames must be received at regular intervals. If a frame arrives too soon, it must be stored at the destination node, thus increasing the storage capacity needs. On the other hand, if it arrives too late, it cannot be displayed and is discarded. In this case, the previous frame must be shown again, and video quality is reduced. And what is more, bandwidth is wasted in transmitting an unusable video frame [18].
- **Latency tolerant** Generally, multimedia applications are not too sensitive to latency. Some of them, the less interactive ones, tolerate well even relatively high latencies, as video on demand. On the other hand, applications with a higher degree of interactivity, as the videophone, need a quicker system response [19][3]. Anyway, high latencies in the connection establishment phase are well tolerated.
- **Short control messages** Many multimedia applications need the transmission of short control messages, besides the long data flows. This traffic is not jitter sensitive, but performance of multimedia applications can be affected by the latency suffered by control messages. In particular, these messages should not be blocked by data flows.

Multimedia traffic may also coexist with best-effort traffic generated by conventional applications. An adequate support must be offered within the proposed router architecture for each kind of traffic.

III. The MultiMedia Router Architecture

As was pointed out before, the MultiMedia Router (MMR) architecture, initially proposed in [16], is targeted to the fulfillment of the following goals:

- To support a large number of multimedia connections, satisfying their Quality of Service (QoS) requirements.
- To allocate the remaining bandwidth to best-effort traffic, minimizing its average latency

- To maximize link utilization when the network reaches saturation

Figure 1(a) illustrates the architecture of the MMR. The following subsections describe some basic trade-offs made in the individual components.

A. *Switching Technique*

In order to be able to provide the necessary QoS to the multimedia flows, a connection oriented scheme must be adopted for their transmission. In this way, resources are allocated to the flows in the connection establishment phase, and, when data are effectively transmitted, delays caused by busy resources will be bounded. The overhead introduced by the connection establishment phase is paid off with the transmission of a large amount of information, as multimedia flows do. Examples of connection oriented schemes are circuit switching [20], pipelined circuit switching (PCS) [21], and ATM [22].

On the other hand, conventional best-effort traffic will coexist with multimedia flows. In this case, a connection oriented technique will increase latency unnecessary. Cut-through techniques, as wormhole [23] or Virtual Cut-Through (VCT) [24], are the most suitable ones to maximize the performance for this kind of traffic (high throughput, low average latency).

As there is no single switching technique suitable for both kinds of traffic, a good trade-off is to use a hybrid approach [25], where the most suitable switching technique is used for each kind of traffic: a connection oriented scheme for the multimedia flows, and a connectionless scheme for best-effort messages.

The selected connection oriented scheme is *Pipelined Circuit Switching (PCS)*, due to factors as its simplicity, its lower control overhead with respect to ATM, or its easy combination with wormhole or VCT. Also, PCS uses flow control to prevent data loss, and needs only small buffers associated to every virtual channel.

Regarding the connectionless scheme, the choice is *Virtual Cut-Through (VCT)* because although its behavior is exactly the same than wormhole when there is no congestion, when congestion occurs VCT can achieve better performance because blocked messages are extracted from the network.This benefits multimedia traffic because interference with data flows is reduced. Also, it has been shown that, when used in NOW/COW environments, a VCT router achieves the same or even better performance than a wormhole router, with a lower implementation complexity [26][27]. The control messages generated by many multimedia applications will also benefit from VCT switching, since they are short and need to be transmitted with a minimum latency.

The main drawback of VCT with respect to wormhole is that buffers large enough to store whole packets are needed. This is not a problem, because when transmitting long best-effort messages, they can be divided into small fixed-size packets. Moreover, modern messaging software levels, as AM [28] or FM [29], already split messages into small pieces, in order to pipeline their transmission over the network interface card, and increase performance. So, packetization does not introduce additional overheads.

Since best-effort traffic must not affect the QoS received by multimedia flows, different resources must be provided to each class of traffic. However, both types of messages should use the same pool of link and buffer resources at a node without partitioning of resources among switching classes. Thus, for both schemes, PCS and VCT, the flow control unit has the same size and will be referred to as a *flit*. Flits are synchronously forwarded through the crossbar.

Differentiation among traffic classes is implemented by assigning a different virtual channel to every multimedia flow, and a small set of virtual channels to best-effort messages. Also, a virtual channel will be dedicated to control messages, and they will be specially considered when performing traffic scheduling. It should be noted that all the virtual channels share the same structure, size and organization, independently of the kind of traffic they are assigned to.

B. Buffer Organization

The first main concern regarding buffer organization is whether they are attached to the input ports, the output ports, or somewhere else (i. e. central queues).

Central shared buffers are the best option in terms of storage capacity, but it has several important drawbacks, as the complexity of flow control, or the lack of isolation among flows. The last one is very important because the presence of a bursty source can exhaust buffer space, causing data loss or high delays to other flows. In any case, there would be a degradation of the QoS received by the applications. Thus, a router organization based on central buffers is not well suited to the MMR objectives.

An output buffered organization has been traditionally preferred in switches for LANs and WANs. It is usually more efficient than an input buffered one because it does not suffer from

HOL-blocking [30], and QoS provisioning is easier to implement. However, packets entering the switch must be processed immediately to decide the output buffer where they should be stored. In order to avoid data loss, the output buffers should admit as many concurrent packet writings as input ports has the switch, thus limiting switch scalability. Moreover, as current link speeds are near the limits imposed by technology, this organization has become practically unfeasible.

Input buffers have been usually deployed in wormhole-like router. This organization simplifies considerably flow control and provides low latencies for low and medium workloads. However, it presents two serious problems when used in a LAN/SAN environment. The first is that it suffers from HOL-blocking at medium-high loads, although there exist several solutions to this problem, as the use of a Virtual Output Queuing (VOQ) organization or per-connection buffering [31][32]. Another problem is that the round trip delay of flow control information makes necessary the use of large buffers in order to avoid data loss when links are long, if stop-and-go flow control is used [6]. This can be solved by using a credit-based flow control scheme [15]. In order to better amortize the overhead introduced by this flow control scheme, large flow control units should be used.

MMR router's buffers will be placed *at the input ports*, because this is the organization typically used with the selected switching techniques (PCS and VCT), flow control is easier to perform, and scales better than the other alternatives, specially with the current high speed links. In order to avoid HOL-blocking, per-connection buffering will be provided as a large set of virtual channels. This will help also to provide QoS guarantees to the multimedia flows.

When a connection is set up, a virtual channel in every link belonging to the path between source and destination is reserved. This allows to consider every connection independently from the others, and perform per-flow scheduling. In order to be able to support a large number of concurrent connections, needed in a multimedia environment, the buffers associated to each link must be organized in a large number of virtual channels.

Virtual channels have been traditionally organized as a set of queues linked by a multiplexer. As indicated in [33], router delays can increase substantially when a large number of virtual channels are multiplexed onto physical links. This is due in part to the multiplexer and virtual channel controller delays. Moreover, fully demultiplexed crossbars [34] (i.e., one virtual channel per crossbar port) become prohibitively expensive in silicon area as the number of virtual channels increases. Thus, it becomes necessary to devise an alternate buffer organization to support a large number of virtual channels.

Virtual channels in the MMR are organized as a set of RAM memory modules, interleaved with a simple pattern, in a way quite similar to the memory organizations used by vectorial and pipelined computers [35].

Every memory module has one input and one output port. Every flit is stored as a certain number of consecutive words in memory. Flits belonging to the same virtual channel are stored in adjacent sets of memory addresses. Before transmitting every flit, a control word with the number of the virtual channel to which the flit belongs is inserted. Control words are distinguished from data words by a control bit, as in Myrinet [6]. The number of virtual channel is used to address the corresponding buffer within the RAM. After storing every word, the address counter is automatically incremented. Interleaving is carried out through the least significant bits. The number of memory modules and the flit size are selected to match with the memory access time, in clock cycles.

The proposed buffer organization is depicted in Figure 1(b). Read addresses, needed to retrieve the flits, are generated by the link scheduler, once the scheduling for that cycle has been computed (see Section III-D).

It must be noted that a word can be read right after being stored, without awaiting for the reception of the full flit. In this way, transmission can be pipelined at the word level. Memory access time increments latency in a few clock cycles, but multimedia applications usually tolerate well latency.

C. Switch Organization

In most routers the internal switch fabric is implemented as a crossbar. Crossbars can be multiplexed, partially multiplexed or fully demultiplexed [34], depending on the number of ports. Although some routers use a fully demultiplexed crossbar [7], with as many ports as virtual channels, this organization becomes prohibitive when the number of virtual channels is large. Even for a relatively small number of virtual channels, some commercial routers use a multiplexed crossbar [36].

The MMR internal switch fabric is a *multiplexed crossbar*, with as many ports as communication links. It reduces silicon area by V and V^2, respectively, with respect to a partially multiplexed and a fully demultiplexed crossbar, where V is the number of virtual channels per link.

The main drawback of a multiplexed crossbar is that arbitration is needed every time an input link switches from one virtual channel to another. Arbitration is required at the input side to select a virtual channel from each input link. This problem is called *link scheduling*. Arbitration is also required at the output side because several virtual channels may request the same output link. This is known as *switch scheduling*. Nevertheless, this overhead can be hidden by overlapping arbitration with the transmission of a previous flit, if flits are large enough. Switch reconfiguration overhead can also be similarly hidden.

An interesting feature of multiplexed crossbars is that buffers are not required at the output side. As switch output ports are directly connected to output links, flits are directly transmitted through the switch and the corresponding output link. However, a few phit [1] buffers can be used to pipeline information through the switch and the link. Finally, serialization is required if internal data paths are wider than physical links.

D. Packet and Flit Transmission

Arbitration in a router with input buffers and multiplexed crossbar is difficult if requests arrive asynchronously, specially if QoS guarantees have to be offered. In this case, requests must be stored and synchronized in order to give preference to the highest priority requests, although they arrive later. Moreover, when a given input link wins arbitration to transmit the next flit, it has to wait until the corresponding output port is released, even if it has just finished the transmission of the previous flit. Thus, part of the link bandwidth is wasted.

Consequently, in order to fully utilize switch and link bandwidth, the MMR assigns *synchronously* switch ports and output links to the input virtual links with pending requests. This allows also to simplify router design.

Flit transmission is organized as a sequence of *flit cycles*. At the beginning of every flit cycle, all the input links with ready flits [2] transmit a control word with the identifier of the virtual channel to which the flit belongs. Then, they transmit synchronously one flit through the crossbar and the output links. Simultaneously, arbitration is carried out, in order to compute the output link and

[1] A *phit* is the amount of information that is transmitted in a single router clock cycle. It is also the wide of the internal datapaths. Sometimes it is also referred to as a *word*.

[2] Here, “ready flits” means the flits selected for transmission at the current flit cycle. There is at most one ready flit at each input link.

switch port assignments for the next flit cycle.

Once current flit transmission has finished, the crossbar is reconfigured for the next flit cycle. This operation needs only one clock cycle. During crossbar reconfiguration, no data flits are transmitted. The MMR makes most of this cycle forwarding pending routing probes, backtracking probes, and acknowledgments involved in the set up of connections. Once the crossbar has been reconfigured, transmission of the next flit starts, that is, the next flit cycle starts. It must be noted that, although transmission is synchronous inside every router, different routers work asynchronously.

Control messages and best-effort traffic are transmitted using virtual cut-through (VCT) switching, that is, no connection establishment phase is performed. When a message is larger than a flit, it is divided into small, fixed size packets, as explained before. The flow control unit in VCT is a packet. Given that the flits used for multimedia transmissions are large, the VCT packet size is made equal to a PCS flit. In this way, PCS and VCT flow control units have the same size, simplifying the router design.

Packet headers are routed as soon as they reach the router, through the routing unit. Then, the packets are stored at their corresponding virtual channels, until they are scheduled for transmission. Virtual channels that hold control messages and best-effort packets are considered also in the arbitration process, although in a different way. The virtual channels assigned to control messages are considered with the highest priority, since they need minimum latency transmission. On the contrary, the virtual channels carrying best-effort traffic are assigned the lowest priority, because this type of traffic should use only the remaining bandwidth not used by multimedia connections.

E. Routing Unit

The routing unit executes the routing algorithm. The purpose of this algorithm is to compute the paths that must be followed by the connection establishment probes, and by the control and best-effort messages, in their travel through the network.

There exist several routing algorithms, valid for wormhole networks with irregular topologies, that offer a high degree of adaptivity [37][38]. These algorithms can also be applied to both virtual cut-through and pipelined circuit switching.

But, due to the fact that in PCS the probes travel independently of the data, more flexible routing algorithms can be deployed in this case, i. e., backtracking algorithms. An example is the

Exhaustive Profitable Backtracking (EPB) algorithm [39], which performs an exhaustive search of the minimal paths in the network until a valid path is found or the probe backtracks to the source node.

The PCS connection establishment process will be slightly modified in the MMR. In PCS a probe is sent from the source to the destination, carrying control information and the destination address [40]. In the MMR, the probe will carry also the bandwidth requirements of the connection. At every node, when the probe is successfully routed, a virtual channel and its corresponding buffers are reserved, as in PCS. Besides, the requested bandwidth is allocated to the connection.

Finally, the routing unit keeps the channel mappings among input and output virtual channels for the established connections. Virtual channels are specified by the physical link where they belong and the number of virtual channel within it. Direct and reverse mappings are needed. The first ones are used for forwarding the data flits, and the second ones are used by the backtracking probes and the acknowledgments. These matchings are also used to propagate status information (credits, dynamic modifications in bandwidth allocation, . . .).

IV. Resource Scheduling Algorithms

The support for QoS guarantees within the MultiMedia Router is implemented as the solution to three basic problems, namely, bandwidth reservation, link scheduling, and switch scheduling.

The *bandwidth reservation scheme* allocates an adequate fraction of link bandwidth to every connection, when they are being established. *Link scheduling algorithms* must guarantee that the bandwidth reserved by every connection is available during data transmission. Regarding *switch scheduling algorithms*, their purpose is to minimize intern conflicts for crossbar ports, among the scheduled connections, thus maximizing crossbar utilization. A poor switch scheduling algorithm provides low bandwidth utilizations through the router, and also low link utilization.

That is, link and switch scheduling algorithms work closely to satisfy the QoS guarantees based on the reservation made by multimedia connections, and to utilize bandwidth in the most efficient way. They are the ones who are in charge to effectively distribute available bandwidth among the different connections, based on their reservation. As the design target is a compact mono-chip router, these strategies must be amenable to a compact and high speed implementation. Nevertheless, the use of large flits, and the fact of the concurrent computation of scheduling with flit transmission, allows for a relatively wide working margin.

In the next subsections, each of the strategies developed to cope with bandwidth reservation, and link and switch scheduling will be described.

A. Bandwidth Reservation

As was pointed out before, link and switch port bandwidth are divided into flit cycles. These cycles are grouped into *frames* or *rounds*. The number of flit cycles in a round is an integer multiple K ($K > 1$) of the number of virtual channels per physical link ($numVC$). That is:

$$round\,size = K \times numVC, \;\; K > 1 \tag{1}$$

Connection bandwidth is reserved as an integer number of flit cycles. These flit cycles will be assigned to the connection during every round, on a demand driven basis. Thus, a greater value of K provides a higher granularity for bandwidth reservation. However, jitter could increase, because the round takes more time to finish. So, the selected value for K must be a trade-off between flexibility and jitter.

When a connection is being set up, the source node produces a routing probe which carries information on bandwidth requirements. This probe tries to establish a connection by building a path between the source and destination nodes, reserving portions of link bandwidth and the corresponding buffers as it traverses successfully the intermediate nodes.

If resource reservation succeeds, the connection is established, and the request is admitted. If the necessary resources are not available at some point of the path, the connection fails, and all the resources that could have been reserved are released. If a backtrack search is used, other paths may be tried.

For *CBR (Constant Bit Rate) connections* this mechanism is easy to implement. The probe carries information about the bandwidth requested by connection c, expressed as a number of flit cycles per round (BW_m^c). Every output link L needs a register $resBW_L$ to keep track of the total number of flit cycles per round reserved up to this moment. This register will be incremented in BW_m^c cycles when a CBR connection c is admitted, and is decremented in the same amount when the connection c is teared down.

A CBR connection c can only be admitted if the following condition holds:

$$resBW_L + BW_m^c < round\,size \tag{2}$$

The problem is more difficult to solve for ***VBR (Variable Bit Rate) connections***. Some VBR connections send a continuous data flow, varying the bit rate over time. This could be the case of compressed audio flows. Other VBR connections send variable sized data bursts, but do not inject data between bursts. This would be the case of compressed video.

In order to be able to handle the different requirements of the connections, a probe to set up a VBR connection v will carry the permanent and peak bandwidths for that connection (BW_m^v and BW_p^v, respectively). The permanent bandwidth will usually correspond to the average bandwidth requested by the connection. It can even be equal to zero in the case of a purely bursty traffic source. The meaning of the peak bandwidth is straightforward in both cases. The permanent bandwidth of a connection will be added to the $resBW_L$ register of the corresponding output port, when a VBR connection is being established, and subtracted when the connection is eliminated. Besides, an additional register is needed at every output port to support VBR reservations. This register, denoted as $resBWp_L$, stores the total peak bandwidth requested by all the connections that use link L. It is incremented and decremented in BW_p^v, when connections are set up and teared down, respectively.

The conditions that must be checked to admit a VBR connection v are the following:

$$resBW_L + BW_m^v < round\,size \tag{3}$$

$$resBWp_L + BW_p^v < CF \times round\,size \tag{4}$$

The first condition is equivalent to the one applied to CBR connections. Regarding the second condition, it uses an additional parameter CF. This is a *concurrency factor*, whose value is initialized at router start-up. It must be noted that the proposed reservation mechanism does not guarantee the availability of the peak bandwidth to a connection in case it needs it. Such guarantee would waste a great part of bandwidth. However, the peak bandwidth will be available with some probability. This probability depends on the value of the concurrency factor CF. A greater value for CF means that link bandwidth will be shared by more VBR connections. Thus, the probability of several peaks from different sources to occur at the same time is greater. As there is not enough link bandwidth to absorb all of them, the probability to meet the QoS requirements of the connections decreases. Concurrency factor is therefore a trade off between QoS guarantees and link utilization.

Finally, it should be noted that control words can be used along a path to vary dynamically the bandwidth requirements of a connection. This process can be initiated in the source interface as a response to external events or to the actual performance experienced by the connection. For example, if a connection is suffering from congestion at a certain router in the network, its flits will eventually fill the buffers at that node. Due to the use of flow control and small buffers, this information is quickly propagated up to the source interface. Then, the source node can respond to the congestion situation in several ways: decrease data rate transmission, selectively discard data packets or even tear down the connection if it knows that it will not meet the QoS requirements. The idea is that network interfaces, or even the source node CPUs, implement more complex tasks related to bandwidth control, also increasing flexibility, in order to keep routers compact and fast.

B. Link and Switch Scheduling Algorithms

The MultiMedia Router is based on a input-buffered organization and uses a multiplexed crossbar as the switch fabric. This fact makes necessary to perform arbitration among the virtual channels that share a physical link, but also among virtual channels from different links. In the first case, a virtual channel must be selected to use a crossbar input port. In the second case, conflicts on the output ports must be solved. This idea is quite similar to the one proposed in [41][42]. The first arbitration is known as *link scheduling*, while the second is referred to as *switch scheduling*. Also, QoS necessities of every data flow must be taken into account when developing arbitration algorithms.

The objective of the link scheduling algorithm is to select one or more virtual channels from every physical link, and forward them to the switch scheduler. The selection will be carried out by considering the QoS requirements of every connection.

The purpose of the switch scheduler is to compute a conflict-free matching between input and output crossbar ports, from the set of "candidate" virtual channels, previously generated by link schedulers. This algorithm must maximize router throughput, but always taking into account that the QoS guarantees of the different connections must be respected.

The MMR has a link scheduler associated with each input link and a single switch scheduler (see Figure 1(a)). The proposed link and switch algorithms for the MMR are partitioned into three basic decisions:

1) **Candidate selection** Candidate selection corresponds to link scheduling, and is performed in parallel by all the physical links perform it. Every link scheduler generates a set of one or more virtual channels, the "candidates".
2) **Port ordering** Its purpose is to select an output port of the crossbar for matching, among the ones that were requested by one or more of the candidate.
3) **Arbitration** This step is only applied when the output port chosen in the previous phase has more than one requests. In that case, only one of them is selected.

This structure is motivated by the need for concurrent and pipelined operation to keep the link and switch schedulers operating at link speeds.

Several link and switch algorithms which follow this structure have been proposed by the authors [43][44][45][46][47]. In this paper, due to space limitations, only the link and switch schedulers initially proposed in [43] will be described. These are the ones used in the evaluation results presented at the end of the paper. The authors are preparing another paper which will describe with more detail the traffic scheduling algorithms developed for the MMR.

1) Link scheduling: A *candidate* on an input port is defined as the next flit in a virtual channel that is ready to be scheduled to an output port. Each link scheduler will produce a short list of candidates from the virtual channels on that link and record them in the input port *candidate vector*. Each entry in the vector records the scheduling data for the corresponding flit. Typically this is at least the crossbar output port and the priority value.

Candidate selection is based on the priority value. Based on this criterion, anywhere from 1 to N candidates are selected from each input port for an $N \times N$ switch (assuming that there exist active connections in all the ports). The candidates are ordered according to priority value and the position of a candidate on this order is the *level* of the candidate. Candidate vectors are then forwarded to the switch scheduler. Figure 1(c) shows an example candidate vector with four levels of candidates.

Flit priority is updated as a function of the QoS that a flit has experienced up to that point in time *relative* to the QoS requested by the corresponding connection. Such an update operation has been referred to as *priority biasing* [4][48][49] since the priority value is biased by the relative degradation of its service. The biasing operation couples the effect of the scheduler (e.g., queuing delay) with the QoS demand (e.g., jitter bound). This distinguishes this approach from priority update mechanisms such as age counters that do not distinguish connections based on their QoS requirements.

One of the proposed biasing functions is referred to as *Inter-Arrival Based Priority* (IABP) [43]. The priority of the head flit in an input queue is computed as the ratio of the queuing delay and the connection's inter-arrival time. The rate at which the priority of a flit increases depends on the bandwidth of the connection. Thus, different connections are biased at different rates, i.e., higher speed connections are biased faster. IABP couples the ongoing effect of the switch scheduler (queuing delay) with a measure of the demands made by the application (connection bandwidth).

2) Switch Scheduler: Each link scheduler provides a set of requests for output ports to the switch scheduler, stored in the candidate vector. The switch scheduling problem is one of matching input ports to output ports to maximize switch utilization and the QoS across connections. There are several approaches toward generating switch settings at high speeds [50][51]. However, the algorithms proposed in the literature are oriented to maximizing crossbar throughput, but fail in providing the necessary QoS to multimedia flows. There exist some recent proposals of switch scheduling algorithms with some QoS support, but they are or too complex to be implemented in a compact high-speed router, or lack the necessary flexibility to accommodate traffic with differing characteristics [52].

Following candidate selection, the results are organized into a *selection matrix*. This matrix is used to record the output port requests from all input ports. An example of a selection matrix is shown in Figure 1(d). The matrix has column dimension equal to the number of input ports, and row dimension equal to the product of the number of output ports and the number of candidates at each input port (each input port is assumed to provide the same number of candidates, if available). The columns are labeled with the corresponding input port, and the rows are labeled with output ports. The first N rows record the highest priority candidates from each input. For example, entry (i, j) is set if the highest priority flit from input j requests output i. The next N rows in the matrix record the level two candidates from each input port, and so on. For example, in Figure 1(d) there are two candidate flits in row 0 of the upper half- in column 1 and in column 3. This corresponds to the highest priority connections from input ports 1 and 3 having flits destined for output port 0.

From the selection matrix the switch scheduler creates a *conflict vector*. This is the number of non-null entries in each row. The conflict vector identifies the number of conflicts for an output port. Figure 1(d) also depicts the conflict vector corresponding to the matrix shown in the same figure. Note that the conflicts are naturally grouped by priority level.

Once the information has been conveniently organized in the selection matrix and the conflict vector, the two phases involved in switch scheduling are applied.

The *port ordering phase* rules the order in which output ports are examined to resolve port conflicts. This decision is captured in the *ordering function*. One of the ordering functions developed for the MMR is named *COA (Candidate Order Arbiter)* [43][46]. This function consists on selecting output ports first by level and then in increasing order of conflict within a level. Ties are broken by randomly selecting one of the ports. The rationale is that ports with the most conflicts should be matched last since those ports have the most opportunities to be matched to an input port.

Once an output port is selected, it might have more than one request, i. e., it could have more than one conflict. In this case, the *arbitration phase* is applied to select one of the input ports among the competing ones. The criterion is to pick the one with the highest priority value. In this way, QoS requirements of the flows are again taken into account. After finding every matching, data structures are updated, and port ordering and arbitration are applied again. The algorithm moves to the next candidate level when there are no more pending requests within a certain level, and it stops when there are no more pending requests at any level, i.e., the conflict vector is all zeroes. A detailed example showing how the COA algorithm works can be found in [44].

V. Tuning of the Main Design Parameters

In the MultiMedia Router architecture, the effect of several quantitative parameters on the QoS guarantees needed by multimedia flows need to be evaluated. More precisely, the optimum value for flit size, scheduler round size (value of the parameter K), and buffer size has to be determined. This section is dedicated to present the simulation tests carried out in order to tune these parameters. Evaluation results will be preceded by a brief description of the simulation environment and the workload models.

A. Simulation Conditions

Performance evaluation of the proposed architecture has been carried out by means of simulation. A discrete event C++ simulation tool has been developed. This tool models with great detail one single MMR router plus the NICs attached the input/output ports, configuration sometimes referred to as "workgroup router configuration" [53]. Flow control between the NIC and the MMR input ports has also been simulated. The purpose is to gain insight into the behavior of the architecture and the traffic scheduling algorithms.

Flits are generated by traffic sources following different patterns, depending on the type of source (the traffic sources used in the tests will be described later). These flits are stored in the NIC buffers, which are considered infinite. There is one buffer per connection in the NICs, as in the MMR input ports. The physical link controller located in the NIC only considers for forwarding to the router the flits corresponding to virtual channels which have credits available, that is, the flits that have free space in their corresponding MMR buffer. This link controller forwards the flits belonging to the different connections in a demand-driven round-robin fashion, i.e., it performs round-robin among all the connections with both flits and credits available. This simple scheme suffices for forwarding flits to the MMR while guaranteeing QoS. The reason is that when the scheduling algorithm guarantees QoS in the router, the NIC simply adapts flit forwarding to the needs of the router, thanks to the use of small buffers and flow control.

All the simulations presented in this Section were carried out for a 4×4 router, with 1.24 Gbps, 16 bit-wide links. This gives a router cycle of 12.9 nanoseconds. There are 128 virtual channels per physical link. The link and switch scheduling algorithms used are the ones described in Section IV-B, with four levels of candidates. Simulations have been carried out for 600 scheduling rounds for CBR traffic tests, and until 4 GOPs per connection have been transmitted, in the case of VBR traffic.

Workload is generated as a random mix of connections to fill a certain percentage of link bandwidth. Simulated percentages range from medium to high loads. The destination port for every connection is randomly generated. At the beginning of every simulation, connection requirements are read from an external file, and the bandwidth reservation scheme described in Section IV-A is applied. As a result, some connections might be rejected due to the lack of enough bandwidth at the requested output port, specially at high loads. After the connection setup phase, traffic sources start flit injection. During a few scheduling rounds, no statistical information is gathered, in order to get data only when the system is stable.

B. Workload Generation

For the tests presented in this Section, only multimedia workload has been considered. CBR and VBR traffic sources have been developed.

CBR traffic is synthetic, and consists on a set of connections which generate flits at a fixed rate. Three different types of CBR connections have been considered, depending on their bandwidth requirements: {64 Kbps, 1.54 Mbps, 55 Mbps}. These are representative of several applications as audio, video, and high-definition video transmissions, respectively. Besides, they impose a wide range of different bandwidth requirements to the MMR.

VBR traffic is generated from traces obtained from real MPEG-2 video sequences. This is a typical type of multimedia flow.

MPEG-2 video coding standard [54] encodes the video streams as a sequence of different frame types, I, P, and B, ordered with a prefixed and repetitive pattern, called GOP (Group Of Pictures). The GOP used is IBBPBBPBBPBBPBB. I frames encode independent frames, that is, I frames do not need any other information but themselves to be decoded. P frames need the previous I frame in the sequence to be decoded, because the data they hold is related to that on the I frame. Finally, B frames need information from both previous and following P or I frames to be decoded. The bandwidth needed for each type of frame is different. I frames are the most bandwidth consuming, because they carry more information, and B frames are the least bandwidth consuming. Every 33 milliseconds, a frame is injected. The traffic pattern generated by MPEG-2 video sequences is depicted in Figure 2(a). The size of every frame has been extracted from real MPEG-2 video traces.

A frame is composed of a number of flits. These flits must be injected within a frame time, i. e., 33 millisecs. Flits belonging to a frame can be injected in a number of ways. Two different injection models have been developed:

- **Back-to-Back (BB) model** In this model, all the flits are transmitted at a peak bandwidth, common to all the connections. The *peak bandwidth* is such that it allows the injection of the largest frame among all the connections within 33 millisecs. Transmission of each frame starts at a frame time boundary. All the flits that compose the frame are injected at the selected peak rate, and then, the source becomes idle until the next frame time boundary. The peak bandwidth in the experiments is 50 Mbps for all the connections. Figure 2(b) depicts this model. IAT_p stands for the Inter-Arrival Time (IAT) related to the peak bandwidth.

- **Smooth-Rate (SR) model** In this model, the flits that compose a frame are transmitted with a different IAT for every frame, in such a way that the flits of a frame are evenly distributed within the frame time. The IAT has been computed as the ratio between 33 msecs and the number of flits that compose the frame. A graphical representation is shown in Figure 2(c).

C. *Tuning Flit Size*

Flit size influence is twofold. First, every flit is preceded by a control phit. This control phit carries the virtual channel identifier to which the flit belongs to. Second, when a flit is forwarded a phit carrying flow control information is sent to the previous node in the path. In order to better amortize these overheads, large flits are preferred. One advantage of the use of large flits is that the time available to compute link and switch schedules is greater. Recall that scheduling is carried out simultaneously with flit transmission. Thus, the time available to compute it is one flit cycle.

On the other hand, the use of large flits may have two drawbacks: the flits might not be efficiently filled with useful load, and the increase in latency. The first drawback is not applicable, due to the long flows generated by multimedia applications (see Section II). This feature makes the percentage of flits that might not be fully occupied with payload negligible. Regarding the increase in latency, flit transmission can be pipelined at the phit level, thus improving transmission efficiency. Nevertheless, it must be noted that multimedia applications are not too latency sensitive. This fact leaves a certain margin for tolerance. Anyway, tests must be carried out it order to assess this point.

For the tests carried out in this section, the round size was configured with $K = 16$, and buffer size was set to 1 flit per buffer. Due to space limitations, only results for CBR traffic are included. The conclusions obtained for VBR traffic were qualitatively the same, and can be found in [17].

Two flit sizes were considered: 128 bits and 1024 bits. As phit size is fixed to 16 bits, the respective flit cycles are equal to 9 and 65 router clock cycles, considering also the transmission of the control word associated to every flit (see Section III-D). This control word is transmitted once every 8 data words, when 128 bit flits are used, and once every 64 data words, if flits have 1024 bits. The control word is an overhead, since it does not carry payload, but it also must be considered in order to accept connections. This means that when a connection is being set up, the overhead introduced by control words must be added to the effective data bandwidth when checking the availability of bandwidth at the requested output port. The consequence is that, when workload is moderate to high, the amount of rejected bandwidth increases when using small flits. This effect can be seen in Figure 3(a), which depicts the total bandwidth corresponding to the set of accepted connections, related to the total requested bandwidth. Up to a workload level of 76% of link bandwidth, there is no difference in the behavior of both flit sizes. Beyond that point, the use of large flits allows the admission of more bandwidth. For example, the maximum admitted load when using 128 flits is 86%, while it reaches 95% if flits are 1024 bit long.

Figure 3(b) shows the average crossbar utilization by data, that is, the percentage of utilization due to control words is discounted. It can be clearly seen as bandwidth use is much more efficient with 1024 bit flits.

Now, performance in terms of the delay suffered by the flits has to be analyzed. The measure selected for this purpose is the distribution of flit delay since generation, which is computed by counting the number of flits whose delay is lower than a set of thresholds. Thresholds are related to the Inter-Arrival Time (IAT) of every connection, that is, stricter thresholds are considered for the more demanding connections. This is done for every workload level, and every type of connection. Results are presented in Figure 3(c)-(f) for different connections, and different workload levels.

The plots shown in Figure 3(c) and (d) show flit delay distribution for low (64 Kbps) and high (55 Mbps) bandwidth connections, for the highest common workload level reached by both configurations. It can be seen as larger flits allow more flits to fulfill with stricter deadlines, in 55 Mbps connections. Flits from low bandwidth connections perform the same with both flit sizes.

The next workload point is different for 128 and 1024 bit flits, since now the set of accepted connections is different. Distribution of delays for the most demanding connection at these workload points are shown in Figure 3(e) and (f), for 128 and 1024 bit flits, respectively. It can be seen as the percentage of flits which fail to fulfill deadlines around the IAT of the connection has increased noticeably with respect to the previous workload level when 128 bit flits are used. This performance degradation is much lower when 1024 bit flits are used. Also note that the workload point for 1024 bit flits is higher that the one for 128 bit flits. Nevertheless 1024 bit flits still beat 128 bit flits in terms of maximum delay achieved by the flits.

The conclusion that can be extracted from this study is that large flits are preferable, since they help not only to amortize control overheads and to compute switch schedules, but also performance of the multimedia applications is enhanced, and useful workload range is enlarged. The maximum size of the flit would be rather determined by the availability of silicon area to implement buffer space.

D. Tuning Scheduling Round Size

The scheduling round size depends on the value of the parameter K. Recall that the size of the round is computed as $K \times numVC$, with $K > 1$, being $numVC$ the number of virtual channels per physical link in the MMR. The size of the round determines the bandwidth reservation granularity. With greater values for K, link bandwidth is divided into smaller pieces, which are allocated to different connections. In this way, reserved bandwidth (a discrete magnitude) approximates better the real amount of bandwidth occupied by connections (a continuous magnitude). So, link bandwidth utilization is improved. However, as the scheduling round takes more time to complete when K is greater, latency or jitter could be negatively affected. This influence must be checked.

The tests carried out in this section have been configured with a buffer size of 1 flit, and 1024 bit flits. Three values for K have been tested: 4, 16 and 64. Given that the number of virtual channels per physical link is 128, the size of the scheduler round turns out to be 512, 2048, and 8192 flit cycles, respectively. This leads to a bandwidth reservation unit of 2.42, 0.60, and 0.15 Mbps, respectively. It can be observed as, the lower K is, the coarser the granularity of bandwidth allocation is.

Figure 4(a) depicts the percentage of load accepted in the simulator versus the total requested bandwidth, for CBR traffic (similar results are obtained when using a VBR-only workload). It can be noticed that, when the requested load is over 75% of the link bandwidth, the behavior for $K = 4$ differs from the other two values. This is due to the granularity factor introduced by K. For example, a value of $K = 4$ implies to reserve 2.42 Mbps for a 64 Kbps connection, thus wasting lots of bandwidth. When workload is low to medium, the effect of this “over-reservation” does not show, because there is still enough bandwidth for all the connections. When workload is moderate, this effect starts to show, as is reflected in Figure 4(a). When K is increased from 16 to 64, the effect of granularity is negligible, at least for the set of connections used in the experiments. Also, the average crossbar utilization is shown in Figure 4(b) . It can be seen as the router does not reach saturation in the workload range common to the three configurations.

As a measure of application performance, Figures 4(c) and (d) show the distribution of flit delay for the most and least demanding CBR connections, respectively, for a workload level near saturation. It can be seen a very slight improvement in the percentage of flits that achieve delays lower than the tighter thresholds, i.e., $\leq IAT$ for $K = 4$, while there is no difference in the results obtained for $K = 16$ and $K = 64$. Regarding VBR traffic, performance measures are related to the application data units, i.e., the frames, rather than to individual flits. Figures 4(e) and (f) depict the average frame jitter for VBR MPEG-2 connections, with both injection models (SR and BB). Also, negligible differences appear when changing the value of K. In any case the values are in the range of a few microseconds. Note that there are tecniques to absorve at destination jitter values of up to several milliseconds [55].

As a conclusion, it has been shown as the use of small values for K implies a coarser granularity in bandwidth reservation. Consequently, more connections are rejected when workload is moderate. Regarding application performance in terms of delay and jitter, the impact of the value of K is negligible. The reason is that the scheduling round size is effectively used only when reserving bandwidth. During flit transmission, scheduling decisions are taken at every flit cycle, and not for the whole scheduling round. Results suggest the use of the intermediate value for K, i.e., $K = 16$.

E. Tuning Buffer Size

The size of the MMR buffers is closely related to the feasibility of a compact router design. In that sense, minimum sized buffers are preferred, due to the large amount of virtual channels supported by the MMR. The use of small buffers does not imply flit loss, because the MMR applies flow control. Thus, in case of congestion, flits would be eventually stored in the buffers placed at the network interface card (NIC). Then, most of the flits from a congested connection would be stored at its source node. So, the source can easily take decisions to control the connection behavior, such as selectively discarding flits, or even aborting the connection in a extreme case.

In the latter case, a minimum amount of flits from the aborted connection would remain in the network, thus minimizing the amount of wasted resources.

On the other hand, the fact that flits would be mainly stored at NICs instead of routers may make resource scheduling difficult. This could lead to an increase in flit delay, thus degrading QoS. This negative impact would be more noticeable when VBR traffic is present, due to its bursty pattern. Switches have traditionally used large buffers to absorb bursts, and to avoid flit loss and performance degradation [56]. However, the use of flow control to avoid flit loss may make the use of large buffers not necessary. Thus, the minimum router buffer size must be determined, in order to improve the compactness of the design, but without degrading the QoS required by the applications.

The tests carried out in this section have been configured with 1024 bit flits, and $K = 16$. First, simulations were carried out with a CBR only workload, and several buffer sizes, ranging from one flit to 1024 flits. The results were quite predictable, and showed no difference when varying the amount of buffer size. The reason is that CBR sources inject their flits evenly distributed over time and always according to their specified rate. Thus, while there is no saturation, flits are able to traverse the router without forming queues. Therefore, the increase in buffer size does not affect the performance of CBR traffic, and a buffer with capacity for just one flit is enough.

Thus, the study has mainly focused on the performance of the router with VBR traffic. Only some results with the SR injection model will be presented here. The interested reader is referred to [57] for a complete analysis with both injection models.

First of all, average crossbar utilization is shown in Figure 5(a). Results with buffer sizes of 1, 4 and 1024 flits are shown. The reason for testing such a wide range of sizes is to check if a buffer with capacity for a whole video frame would impact performance significantly. The largest video frame used in these simulations contains 1000 flits, approximately. Also, small buffer sizes were tested in order to establish the minimum buffer size necessary to guarantee QoS to MPEG-2 flows.

The behavior for buffer sizes of 1 and 4 flits is very similar. In both cases, the network enters saturation at 82% load (approximately) without showing any performance degradation beyond saturation. The sources generated up to 100% of workload but, beyond 82% of utilization, flow control makes the flits that do not fit into the MMR buffers (which are full) stay at the NIC buffers. Surprisingly, utilization drops heavily for buffers with 1024 flits. One reason for this is the degradation of the priority mechanism (which is based on queue delays) when the queues in the MMR become long. Note that priority takes into account the queue delay, that is, there is a counter associated to every flit in every buffer that stores the delay experienced by the flit in the MMR queues. Consider the case when contention occurs and flits start to form queues. When buffers are large, flits are more likely to wait for their turn for transmission within the MMR input buffers. This implies that their queue delay will be greater than the one experienced if buffers were smaller. In the latter case, flits would spend more time in the NIC buffers, and their queue delay in the router would not be increased as much. So, when using large buffers, those counters are more likely to overflow, as queue delay can reach higher values, thus corrupting the priority mechanism. Besides, the Inter-Arrival Time (IAT) for the connection, also considered in the IABP scheduling algorithm for computing priorities, changes for every video frame injected with the SR model. The modification on the IAT for the connection is transmitted across routers as a control message. These messages are transmitted independently from the flits belonging to the connection, and can arrive at the router before all the flits from the previous frame have been forwarded. When the router is near saturation, and the flits stay longer at the MMR queues, flits belonging to an I frame are treated as if they belonged to a B frame. So, their priority increases more slowly, and they are not forwarded as soon as they should.

Next, the distribution of frame delays will be analyzed in order to know how many frames suffer from delays shorter than a certain deadline. The deadlines considered are related to a frame time (FT, 33 milliseconds), and are set as multiples and sub-multiples of FT, ranging from $FT/16$ to $64 \times FT$ (from 2.06 millisecs to 2.212 secs). Results are shown in Figure 5(b)-(d). The plots show the distribution of frame delay since generation for the workload levels labeled in Figure 5(a) as A, B, and C, which are close to or beyond saturation.

For the A workload point, that is, right before saturation, and with all the tested buffer sizes, all the frames suffer delays lower than the stricter threshold (2.06 millisecs). For workload levels B and C, it can be seen that best performance is obtained for small buffer sizes. The reason of this behavior is the degradation on the priority mechanism used in the resource scheduling algorithm for large buffers, when the router is near saturation, and flits enqueue in the router buffers. For the workload point labeled as B, the router is clearly saturated for 1024 flit buffers, but performance is still tolerable for one and four flit buffers. In that case, all the frames suffer delays lower than 66 millisecs, that is, twice the frame time. For the higher workload point considered, C, the less strict deadlines (132 millisecs, $4 \times FT$) can still be met if small buffers are used. These results are quite reasonable, because a typical deadline for MPEG-2 video transmission is one second between endpoints. This is the value for the CTD (Cell Transfer Delay) recommended by the ATM Forum for video distribution services, using MPEG-2 [58].

VI. Conclusions and Final Remarks

During the last few years, new applications have imposed new demands from the network in cluster and local environments. These demands, or Quality of Service (QoS) requirements, are still not well supported by high-performance state-of-the-art interconnects. The MultiMedia Router tries to be a solution to the provision of QoS in those environments, while retaining a compact and simple design.

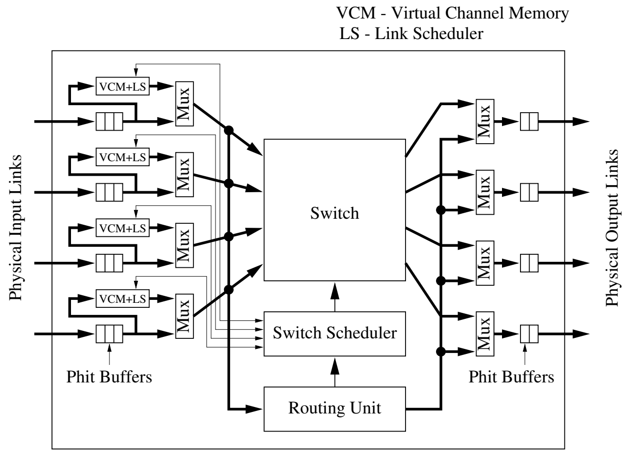
In this work, the main features and design trade-offs related to the MultiMedia Router architecture have been thoroughly described and explained: the use of a hybrid switching technique, an input buffered organization, credit-based flow control, multiplexed crossbar, large flits, ...

Also, evaluation results to show the tuning of the main quantitative design parameters have been presented. The efficiency of the use of large flits in the MMR has been assessed. They help to hide control overheads, to amortize crossbar reconfigurations, and facilitate the scheduling tasks, without degrading QoS. It has also been shown how the size of the scheduler round does not impact application performance, although small rounds do have a negative impact on the amount of accepted bandwidth, due to a coarser bandwidth allocation granularity. Finally, VBR traffic performance is noticeably enhanced when as small as one flit buffers are used, due to the effect of the scheduling algorithms on the flits queued in the MMR buffers. Small buffers are of key importance for the compactness of the router design. They will also help in performing a source-controlled traffic management, thanks to the use of flow control.

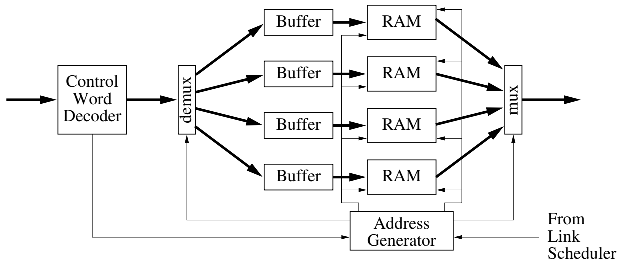
Although a brief glimpse to the algorithms developed for traffic scheduling has been introduced for the sake of completeness, more extensive work has been carried out by the authors on this issue. They are preparing another paper where focus will be placed on the algorithms devoted to traffic scheduling in the MultiMedia Router. These algorithms are aimed at making an effective use of the switch and link bandwidth, while providing the necessary QoS to the multimedia flows.

APPENDIX

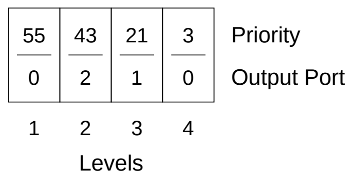
FIGURES



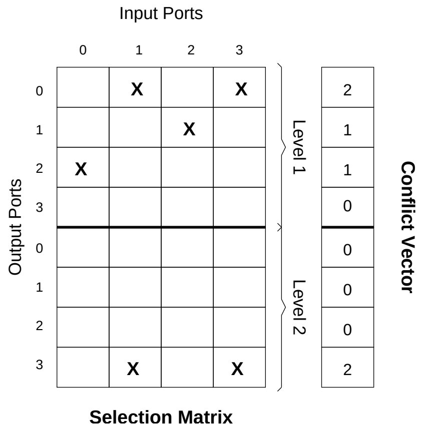
(a) Router organization



(b) Virtual channel memory organization

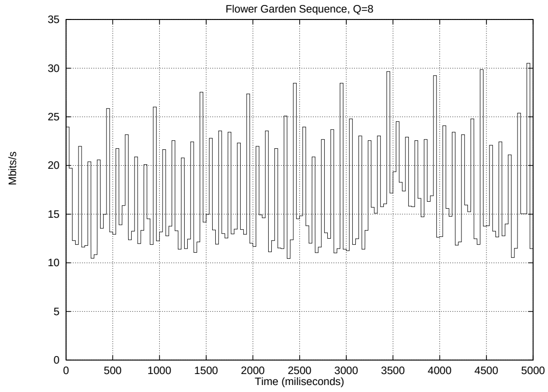


(c) Candidate vector

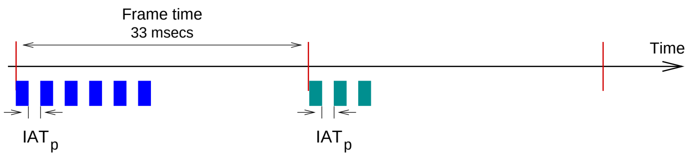


(d) Selection matrix and its corresponding conflict vector

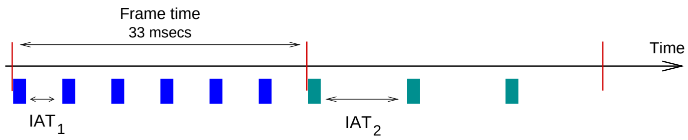
Fig. 1. MultiMedia Router architecture and structures used in link and switch scheduling.



(a) Traffic pattern of a typical MPEG-2 video sequence

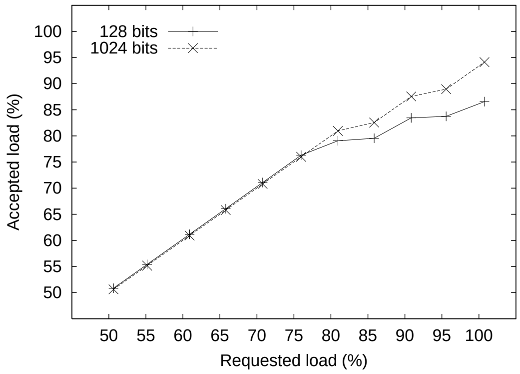


(b) Back-to-Back (BB) injection model

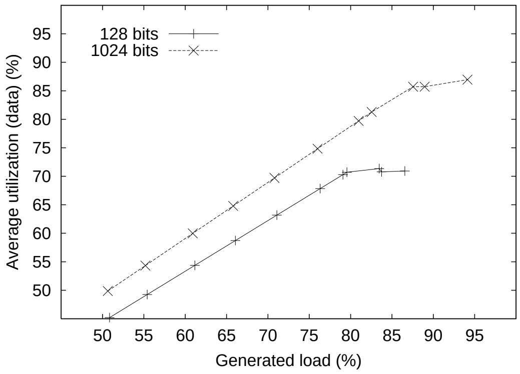


(c) Smooth-Rate (SR) injection model

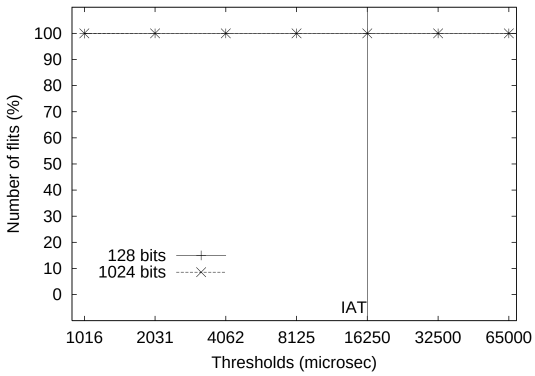
Fig. 2. Example MPEG-2 sequence and frame injection models.



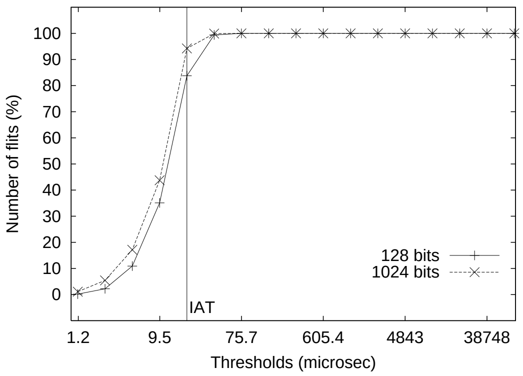
(a) Accepted vs. requested load



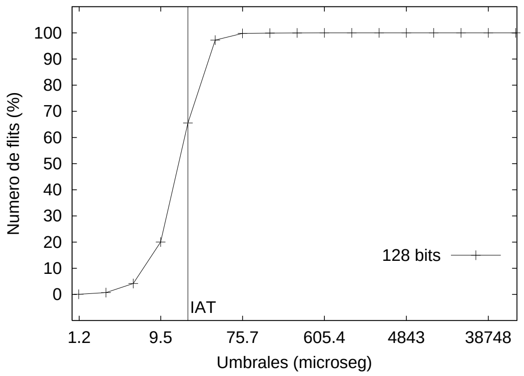
(b) Average crossbar utilization (data)



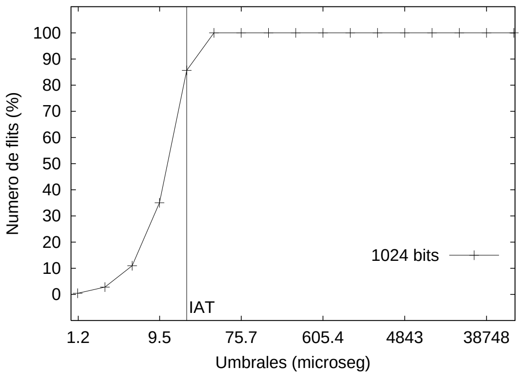
(c) Flit delay distribution
(64 Kbps, workload = 76%)



(d) Flit delay distribution
(55 Mbps, workload = 76%)

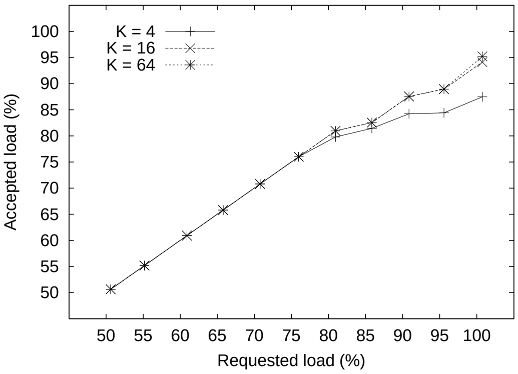


(e) Flit delay distribution
(55 Mbps, workload = 80%)

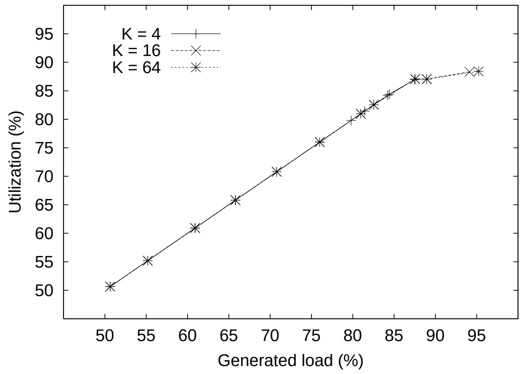


(f) Flit delay distribution
(55 Mbps, workload = 83%)

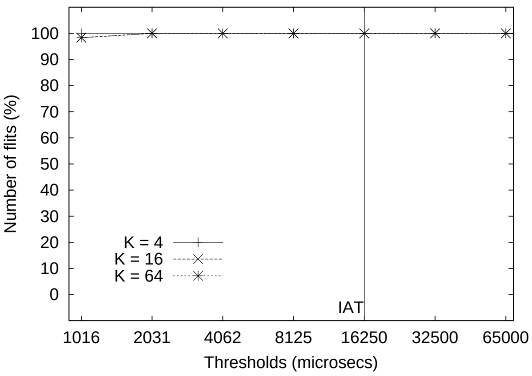
Fig. 3. Impact of flit size - CBR traffic



(a) CBR traffic - Accepted vs. requested load

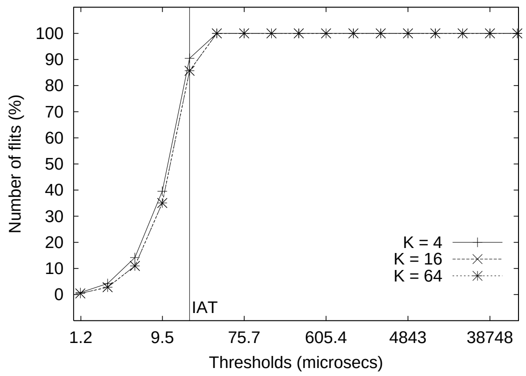


(b) CBR traffic - Average crossbar utilization



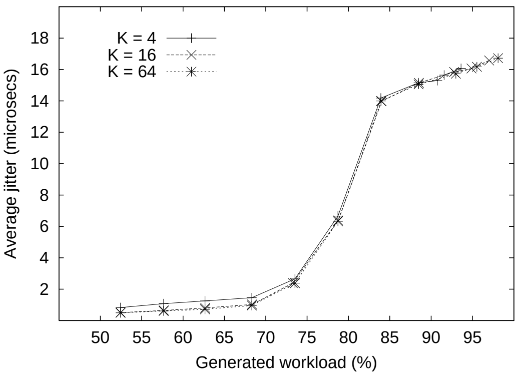
(c) CBR flit delay distribution

(64 Kbps, workload = 82%)



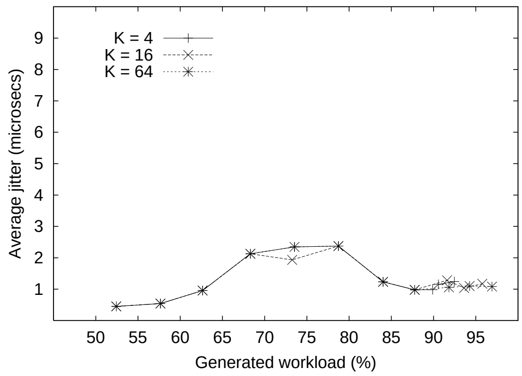
(d) CBR flit delay distribution

(55 Mbps, workload = 82%)



(e) VBR traffic - Average frame jitter

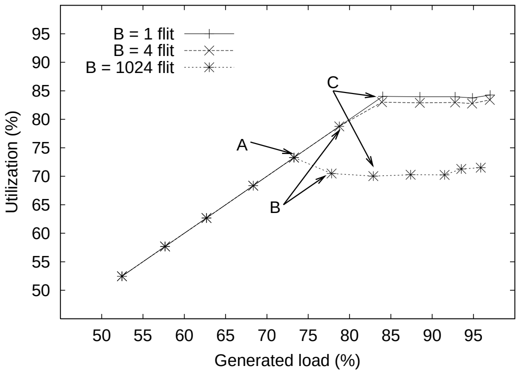
(SR injection model)



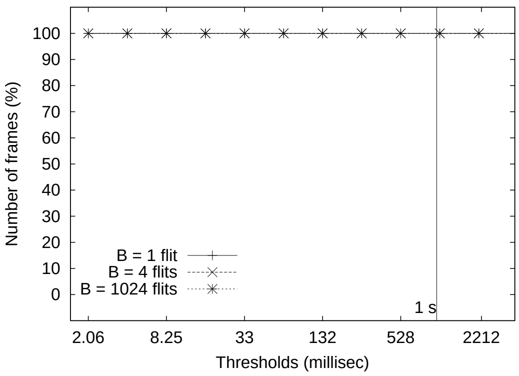
(f) VBR traffic - Average frame jitter

(BB injection model)

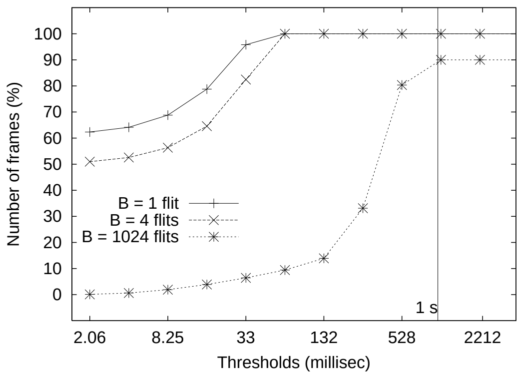
Fig. 4. Impact of scheduler round size



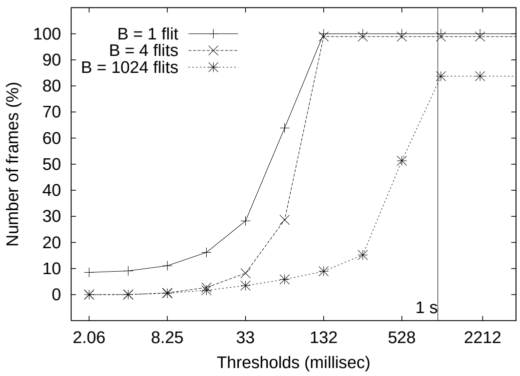
(a) Average crossbar utilization



(b) Distribution of frame delay, workload level A (%)



(c) Distribution of frame delay
workload level B (%)



(d) Distribution of frame delay
workload level C (%)

Fig. 5. Impact of buffer size - VBR traffic

References

[1] K. Dienfendorff and P. Dubey, "How multimedia workloads will change processor design," *IEEE Computer*, vol. 30, no. 9, September 1997.

[2] C.E. Kozyrakis and D. Patterson, "A new direction for computer architecture research," *IEEE Computer*, vol. 31, no. 11, November 1998.

[3] J. F. Kurose and K. W. Ross, *Computer Networking: A top-down approach featuring the Internet*, Addison-Wesley, 2000.

[4] A. A. Chien and J. H. Kim, "Approaches to quality of service in high-performance networks," in *Parallel Computer Routing and Communications Workshop (PCRCW)*, 1997.

[5] E. Pendery and J. Eunice, "InfiniBand Architecture: Bridge over troubled waters," White Paper, available at `http://www.infinibandta.org/newsroom/whitepapers/`, 2000.

[6] N. Boden et al., "Myrinet: A gigabit per second LAN," *IEEE Micro*, February 1995.

[7] W.J. Dally et al., "Architecture and implementation of the Reliable Router," in *Hot Interconnects II*, 1994.

[8] InfiniBand Trade Association, "InfiniBand Architecture specification, volume 1, release 1.0," October 2000.

[9] G. Pfister, *High Performance Mass Storage and Parallel I/O*, chapter 42: An Introduction to the InfiniBand Architecture, IEEE Press and Wiley Press, 2001.

[10] J. Pelissier, "Providing quality of service over Infiniband^{TM} architecture fabrics," in *Hot Interconnects*, 2000.

[11] F.J. Alfaro, J.L Sánchez, and J. Duato, "A strategy to manage time sensitive traffic in InfiniBand," in *Workshop on Communication Architecture for Clusters (CAC'02)*. April 2002, IEEE Computer Society.

[12] F.J. Alfaro, J.L Sánchez, J. Duato, and C.R. Das, "A strategy to compute the InfiniBand arbitration tables," in *International Parallel and Distributed Processing Symposium (IPDPS)*. April 2002, IEEE Computer Society.

[13] K. H. Yum, A. Vaidya, C. R. Das, and A. Sivasubramanian, "Investigating QoS support for traffic mixes with the MediaWorm router," in *High-Performance Computer Architecture (HPCA-6)*, 2000.

[14] L. Zhang, "Virtual Clock: A new traffic control algorithm for packet switching networks," *ACM Transaction on Computer Systems*, May 1991.

[15] M. Katevenis et al., "ATLAS I: A single-chip ATM switch for NOWs," in *Workshop on Communications and Architectural Support for Network-based Parallel Computing (CANPC)*, 1997.

[16] J. Duato, S. Yalamanchili, M. B. Caminero, D. Love, and F. J. Quiles, "MMR: A high-performance multimedia router - Architecture and design trade-offs," in *International Symposium on High Performance Computer Architecture (HPCA-5)*. 1999, IEEE Computer Society.

[17] M. B. Caminero, *Design of a Router Oriented to Multimedia Traffic in LAN Environments*, Ph.D. thesis, Escuela Politécnica Superior. University of Castilla-La Mancha, July 2002, in Spanish.

[18] G. Karlsson, "Asynchronous transfer of video," *IEEE Communications Magazine*, August 1996.

[19] I. Dalgic, *Performance of Ethernet and ATM Networks Carrying Video Traffic Based on Accurate Characteristics of Video Sources*, Ph.D. thesis, University of Stanford, August 1996.

[20] Intel Corporation, *iPSC/2 and iPSC/860 User's guide*, 1991.

[21] B. V. Dao, J. Duato, and S. Yalamanchili, "Configurable flow control mechanisms for fault-tolerant routing," in *22nd International Symposium on Computer Architecture (ISCA)*, June 1995.

[22] M. Prycker, *Asynchronous transfer mode: solution for broadband ISDN*, Ellis Horwood Limited, 1991.

[23] W. J. Dally and C. L. Seitz, "The Torus routing chip," *Journal of Distributed Computing*, vol. 1, no. 3, October 1986.

[24] P. Kermani and L. Kleinrock, "Virtual Cut-Through: A new computer communication switching technique," *Computer Networks*, vol. 3, 1979.

[25] J. Duato, P. López, F. Silla, and S. Yalamanchili, “A high performance router architecture for interconnection networks,” in *International Conference on Parallel Processing (ICPP)*, August 1996.

[26] J. Duato, A. Robles, F. Silla, and R. Beivide, “A comparison of router architectures for virtual cut-through and wormhole switching in a NOW environment,” in *13th International Parallel Processing Symposium, IEEE Computer Society*, 1998.

[27] J. Duato, A. Robles, F. Silla, and R. Beivide, “A comparison of router architectures for Virtual Cut-Through and Wormhole switching in a NOW environment,” *Journal of Parallel and Distributed Computing*, , no. 61, 2001.

[28] T. von Eicken, D. E. Culler, S. C. Goldstein, and K. E. Schauser, “Active Messages: A mechanism for integrated communication and computation,” in *International Symposium on Computer Architecture (ISCA)*, 1992.

[29] S. Pakin, M. Lauria, and A. Chien, “High performance messaging on workstations: Illinois Fast Messages on Myrinet,” in *Supercomputing*, 1995.

[30] M. J. Karol, M. G. Hluchyj, and S. P. Morgan, “Input versus output queuing on a space division packet switch,” *IEEE Transactions on Communications*, December 1987.

[31] H. Obara and T. Yasushi, “An efficient contention resolution algorithm for input queuing ATM cross-connect switches,” *International Journal of Digital and Analog Cabled Systems*, vol. 2, no. 4, October-December 1989.

[32] M. J. Karol, K. Y. Eng, and H. Obara, “Improving the performance of input-queued ATM packet switches,” in *IEEE INFOCOM*, 1992.

[33] A. A. Chien, “A cost and speed model for k-ary n-cube wormhole routers,” in *Hot Interconnects’93*, August 1993.

[34] W. J. Dally, “Virtual-channel flow control,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 3, no. 2, March 1992.

[35] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A quantitative approach*, Morgan-Kaufmann, 1996.

[36] J. Carbonaro and F. Verhoorn, “Cavallino: The tera-flops router and NIC,” in *Hot Interconnects Symposium IV*, August 1996.

[37] F. Silla, *Routing and flow control in networks of workstations*, Ph.D. thesis, Universidad Polit’ecnica de Valencia, 1998.

[38] F. Silla and J. Duato, “High-performance routing in networks of workstations with irregular topology,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 7, July 2000.

[39] P. T. Gaughan and S. Yalamanchili, “Adaptive routing protocols for hypercube interconnection networks,” *IEEE Computer*, May 1993.

[40] P. T. Gaughan and S. Yalamanchili, “A family of fault-tolerant routing protocols for direct multiprocessor networks,” *IEEE Transactions on Parallel and Distributed Systems*, May 1995.

[41] D. Stiliadis and A. Varma, “Providing bandwidth guarantees in an input-buffered crossbar switch,” in *IEEE INFOCOM*, 1995.

[42] D. Stiliadis, *Traffic scheduling in packet-switched networks: Analysis, design and implementation*, Ph.D. thesis, University of California at Santa Cruz, 1996.

[43] D. Love, S. Yalamanchili, J. Duato, M.B. Caminero, and F.J. Quiles, “Switch scheduling in the Multimedia Router (MMR),” in *International Parallel and Distributed Processing Symposium (IPDPS)*. May 2000, IEEE Computer Society.

[44] M. B. Caminero, F. J. Quiles, D. Love, S. Yalamanchili, and J. Duato, “Switch scheduling in the MultiMedia Router (MMR),” Tech. Rep. DIAB-00-02-10, Computer Science Department. University of Castilla-La Mancha, February 2000, available from `http://raap.info-ab.uclm.es/`.

[45] M.B. Caminero, C. Carrión, F.J. Quiles, J. Duato, and S. Yalamanchili, “A cost-effective hardware link scheduling algorithm for the Multimedia Router (MMR),” in *Lecture Notes on Computer Science: International Conference on Networking (ICN’01)*. July 2001, vol. 2094, Springer-Verlag.

[46] M.B. Caminero, C. Carrión, F.J. Quiles, J. Duato, and S. Yalamanchili, "Investigating switch scheduling algorithms to support QoS in the Multimedia Router MMR," in *Workshop on Communication Architecture for Clusters (CAC'02)*. April 2002, IEEE Computer Society.

[47] M.B. Caminero, C. Carrión, F.J. Quiles, J. Duato, and S. Yalamanchili, "A new switch scheduling algorithm to improve QoS in the MultiMedia router," in *IEEE International Workshop on Multimedia Signal Processing (MMSP'02)*. December 2002, IEEE Computer Society, to appear.

[48] D. Garcia and D. Watson, "Servernet II," in *Workshop on Parallel Computer Routing and Communication*, June 1997.

[49] J. H. Kim, *Bandwidth and latency guarantees in low-cost, high-performance networks*, Ph.D. thesis, University of Illinois at Urbana-Champaign, 1997.

[50] T. E. Anderson et al., "High speed switch scheduling for local area networks," *ACM Transactions on Computer Systems*, 1993.

[51] Y. Tamir and H.C. Chi, "Symmetric crossbar arbiters for VLSI communication switches," *IEEE Transactions on Parallel and Distributed Systems*, vol. 4, no. 1, 1993.

[52] G. Nong and M. Hamdi, "On the provision of Quality-of-Service guarantees for input queued switches," *IEEE Communications Magazine*, December 2000.

[53] Cisco Systems Inc., *Network Design and Case Studies*, Cisco Press, 2nd edition, 2000.

[54] "Generic coding of moving pictures and associated audio. Recomendation H.262. Draft International Standard ISO/IEC 13818-2," 1994.

[55] M. Perkins and P. Skelly, "A hardware MPEG clock recovery experiment for variable bit rate video transmission," Tech. Rep. ATM94-0434, ATM Forum, 1994.

[56] P. Cuenca, L. Orozco-Barbosa, F. J. Quiles, and A. Garrido, "Loss-resilient ATM protocol architecture for MPEG-2 video communications," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, June 2000.

[57] M.B. Caminero, C. Carrión, F.J. Quiles, J. Duato, and S. Yalamanchili, "Tuning buffer size in the Multimedia Router (MMR)," in *Workshop on Communication Architecture for Clusters (CAC'01) - International Parallel and Distributed Processing Symposium (IPDPS)*. April 2001, IEEE Computer Society.

[58] M. Schwartz and D. Beaumont, "Quality of Service requirements for audio-visual multimedia services," Tech. Rep. ATM94-0640, ATM Forum, 1994.