Performance Analysis of Storage Area Network Switches

Andrea Bianco, Paolo Giaccone, Enrico Maria Giraudo,
Fabio Neri, Enrico Schiattarella *
Dipartimento di Elettronica - Politecnico di Torino - Italy
e-mail: {bianco, giaccone, giraudo, neri, schiattarella@mail.tlc.polito.it}

Abstract— **This paper focuses on a switching architecture designed for Storage Area Network (SAN) applications, with a crossbar switching fabric and an aggregate bandwidth of hundreds of Gbps. We describe the architecture and adopt an abstract model of the flow-controlled, credit-based, packet transfer around the switching fabric. The major effects on performance of the credit-based flow control are investigated under different system parameters.**

I. Introduction

In recent years, Storage Area Networks (SANs) have emerged as the key solution to allow servers to access promptly and reliably large amounts of data. The traditional Directly Attached Storage (DAS) paradigm, in which each server has a dedicated connection to its storage devices (disks, tapes, CD libraries, etc.), has shown severe limitations in terms of performance, scalability, reliability and ease of management. The SAN is a dedicated network infrastructure that connects servers to storage devices, with very different requirements from the LAN and the WAN. The intermediate nodes of the network are packet switches that support high-throughput, low latency and loss-free communication. This paper is focused on packet switching architectures specifically designed for SAN applications.

Today, most SANs are based on Fibre Channel technology [1], [2], [3], which has been specifically designed to interconnect peripheral devices to computing systems. The Fibre Channel standards define a simple and clean data path, suitable for fast implementation in hardware. To allow end-nodes to process incoming frames at very high speed and to avoid large retransmission and reassembly buffers, the network guarantees that transmitted frames are never lost, duplicated, or reordered. A credit-based flow control mechanism is used, both on each link and end-to-end, to prevent buffer overflow in case the receiver cannot sustain the data rate of the sender. Switches can actively use such mechanism to regulate traffic entering from external links, but must manage their internal resources to avoid frame losses under any circumstances.

* The authors would like to thank Emilio Leonardi for his precious comments on this work.

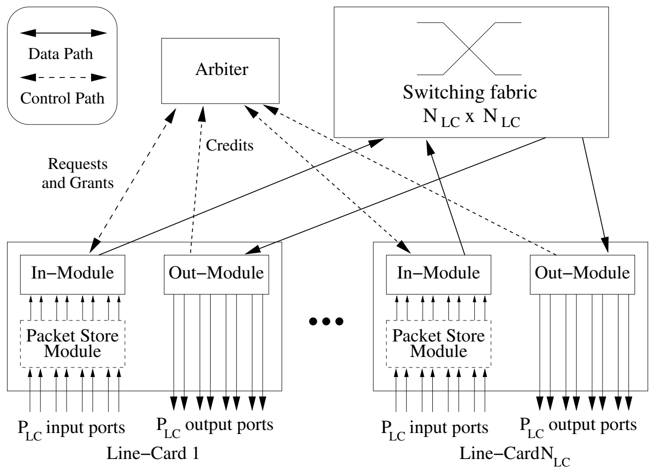

Fig. 1. Logical architecture of a $N \times N$ switch: the switching fabric is a CIOQ architecture with single queue per input; virtual output queues are present in the Packet Store Module and in the In-module.

We present a possible design and study the performance of a combined input output queued (CIOQ) SAN switch. The main differences with respect to LAN/WAN switch designs are the addition of a centralized arbiter, that provides flow control with fine granularity, and a number of backpressure mechanisms and buffer management techniques that guarantee loss-free operation and improve fairness. For the sake of better scalability, the system is fully asynchronous; this also allows a native way of switching variable-size packets, without the need for segmentation into fixed-size data units and reassembly.

We focus mainly on the flow control and backpressure mechanisms in such CIOQ switch, and on their effects on performance. Although we present results considering a particular choice of parameters, (number of ports, external and internal link bandwidths and de/multiplexing factors) derived from implementation constraints, the main observations drawn from performance results hold in general for switches with similar architectures.

II. The switching architecture

Figure 1 shows the logical architecture of the considered SAN switch, when drawing two linecards. Packets enter the linecard through an input port and are multiplexed to share the access link to the switching fabric (*uplink phase*). After traversing the switching fabric, they reach their destination linecard and finally the proper output port after demultiplexing (*downlink phase*). Following the data path, it is possible to distinguish three stages: the first one, internal to the linecard, where data flows are multiplexed to form a higher speed aggregate flow; the second stage, inside the fabric, that switches the aggregate flows between linecards; the third stage, internal to the linecard, where the aggregate flows are demultiplexed.

Four main buffering stages can be identified. In the first stage, two storage areas are available: first, packets are stored in the Packet Store Module according to a virtual output queue (VOQ) architecture; later, they are transferred to the shared memory contained in a high-speed interface towards the switching fabric, called In-module. In the second stage, a small FIFO queue is also available internally to the switching fabric at each input and output port (CIOQ architecture) to cope with high-speed asynchronous links between linecards and the fabric, and with the moderate speedup $K = 2$ available in the switching fabric; since just one queue for each input is present, a simple round robin mechanism at the output is sufficient to schedule the packets and provide fair service. In the third stage, the Out-module stores the packets received from the switching fabric and adapts their format for the transmission to the output ports. The link speeds are set fast enough to avoid bandwidth bottlenecks in the uplink phase.

Two control mechanisms are available to meet the requirement of lossless behavior dictated by the SAN environment: distributed *backpressure* and centralized *credit-based flow control*. Explicit backpressure, providing a coarse form of loss prevention, is implemented in each buffering stage: when a buffer becomes full, a backpressure signal is sent backwards to block each source that could flood that buffer. In addition to backpressure, the access to the switching fabric is governed by a central arbiter, which receives individual packet transmission requests from the linecards (the In-module sends such requests for each received packet) and grants the transmissions depending on the availability of the buffers at output ports. The main flow control mechanism is between the Out-modules and the In-modules. The phases in the flow control loop, shown in Fig. 2, are the following: (1) when a buffer is available at the output port, the Out-module sends a credit to the arbiter for that output port; (2) the arbiter compares the ungranted requests from the In-modules with the available credits; (3) the arbiter grants the In-module for the transmission of a packet to an available output port; (4) the In-module sends to the switching fabric the packet at the head of the queue corresponding to the received grant; (5) the switching fabric transfers the packet to its destination linecard; (6) the Out-module receives the packet and generates a new credit whenever the packet has been completely sent to the output port. Since buffers in Out-modules are associated with output ports, this flow control scheme is based on an arbitration granularity of (input port, output port) pair, to allow a fair and more flexible access to output resources. As a consequence, all the packets generated by the same input port and directed to the same output port belong to the same flow. The number of available credits X for each output port, is determined by the amount of memory available per port in the Out-module.

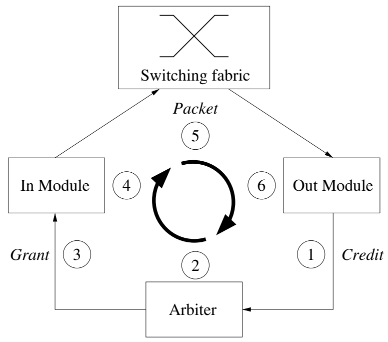

Fig. 2. Credit-based flow control loop managed by the central arbiter

We assume that N input-output ports are divided among N_{LC} linecards. If P_{LC} is the number of ports in each linecard, then $N = P_{LC} \times N_{LC}$. We assume $N = 256$ ports running at 2 Gbps, aggregated into $N_{LC} = 16$ linecards, each with $P_{LC} = 16$ ports. Hence, the switching fabric runs 16 ports at 32 Gbps, and the overall switching bandwidth is roughly half terabit. We assume that the fabric is implemented in a single ASIC chip, as this was shown to be feasible with current technology.

All main modules (linecards, switching fabric and arbiter) run in an asynchronous way, so that the uplink and downlink phases occur concurrently and independently. In a distributed implementation each module can be built on a dedicated ASIC chip. The packet transfers between modules are per-byte, to exploit the maximum transfer bandwidth available. These choices allow native support for transferring variable-size packet, without the need of fragmentation/reassembly modules. The memory in the In-module and the Out-module is instead organized in fixed-size memory segments, dimensioned for the maximum transfer unit (MTU) of the network (in the Fibre Channel case, 2 Kbytes); this coarse segmentation, which allows a higher memory speed, leads to inefficiencies in memory usage, but does not affect the maximum system throughput and available bandwidth thanks to the per-byte transfer across the linecards. Note that the architectural choice of keeping the modules asynchronous eases multi-rack implementations, where it is difficult to distribute the same time and phase signal to all linecards because of the different propagation delays.

III. Performance study

We developed a simulation program to study the system performances, modeling the switch architecture presented above. We concentrate our performance study mainly on the credit mechanism managed by the central arbiter, to assess its behavior under different types of traffic.

A. Simulation settings

Table I summarizes the settings adopted in the simulation. The main variable we consider to assess the performance is X, i.e. the number of credits for each output port.

Parameter	Symbol	Value
Input - Output ports		
Input - Output ports per linecard	P_{LC}	16
Linecards in the system	N_{LC}	16
Overall number In-Out ports	N	256
Link speeds for data and signalling		
Input - Output ports (data path)	V_{PO}	2 Gbps
Linecard $\leftrightarrow$ crossbar (data path)	V_{CR}	32 Gbps
Linecard $\leftrightarrow$ central arbiter (control path)	—	2 Gbps
Packet size		
Minimum packet dimension	—	64 bytes
Maximum packet dimension	MTU	2048 bytes
In-module & Out-module		
Number of credits per output	X	*(variable)*
In-module shared buffer size	—	100 MTU
Switching Fabric		
Internal speedup	K	2
Input fabric buffer size	—	20 KBytes
Output fabric buffer size	—	40 KBytes

TABLE I
Summary of the main architecture parameters

The chosen parameters refer to a specific implementation, but the main results of this study can be extended also to other switching fabric architectures providing lossless and high throughput switching.

B. Traffic model

Traditional traffic models employed in the study of switching systems are not suitable for flow-controlled switches, since flow control mechanisms interact also with the sources (in a SAN, through the end-to-end or buffer-to-buffer flow control of Fibre Channel). Since the sources are controlled, traditional open loop sources, like Bernoulli or on-off/bursty, cannot be employed.

The whole system sources-plus-switch is lossless and traditional concepts like 100% throughput should be carefully defined. We adopt the following traffic model. A generic open loop source generates packets with traffic matrix $\tilde{\Lambda} = [\tilde{\lambda}_{ij}]$, where $\tilde{\lambda}_{ij}$ is the *gross* average rate of input i directed to output j, with $1 \leq i, j \leq N$. The source can be active or inactive, depending of the flow control signal received. If it is active, the packet generated by the source is actually sent to the input port. If the source is inactive, because of a flow control signal, the generated traffic is lost. Hence, the *net* average arrival rate of traffic entering the switch is given by $\Lambda = [\lambda_{ij}]$, with $\lambda_{ij} \leq \tilde{\lambda}_{ij}$. In stationary conditions, also the average service rate is λ_{ij}. Note that, with this traffic model, the transmission queue at the transmitter is not implemented, i.e. its size is zero; this does not affect the results since in our simulations we set the gross load per input equal to 1 (i.e., $\sum_j \tilde{\lambda}_{ij} = 1$).

We define the throughput as the ratio between the actual traffic entering (and exiting) the switch and the gross offered traffic, i.e. $\lambda_{ij}/\tilde{\lambda}_{ij}$ for input-output relation i to j. In this paper we present simulation results only under uniform traffic: $\tilde{\lambda}_{ij} = 1/N$, for $1 \leq i, j \leq N$.

We consider two possible distributions of the packet size:

- fixed packet size, equal to either 64 (for small packets) or 2048 bytes (for large packets), which are the minimum and maximum size supported in the switching architecture;
- variable packet size, multiple of 64 bytes and distributed uniformly between 64 and 2048 bytes.

C. Performance under uniform traffic

This scenario is critical since all output resources are contended among all input ports, especially inside the switching fabric. Two facts can affect negatively the final throughput: *starvation* and *backpressure*. Starvation is present in the In-module for all packets waiting for credits from the Out-module, and it is due to the lack of credits (hence of buffer space in the Out-module). Backpressure is generated by the fabric FIFO input queues when they saturate, and block the corresponding In-module.

The intrinsic contention generated by the traffic inside the switching fabric also affects the final throughput. Indeed, when packets directed to the same Out-module (linecard) are present at the head of different fabric input queues, just one of them can be transferred. The fabric queues are FIFO and this fact prevents the packets behind the heads of the input fabric queues to access the crossbar. This is the well known "head-of-line (HOL) blocking" problem, which increases the occupation of the input fabric queues. Note that an internal speedup $K > 1$ mitigates the effects of the HOL-blocking.

The number of credits X available for each output port has doubled-sided effects on the final throughput. When X grows, the starvation in the In-module decreases thanks to the higher sending rate from the In-module. Since also the input fabric queues grow, this may lead to backpressure. At the same time, large X implies a larger HOL blocking among the contending fabric FIFO

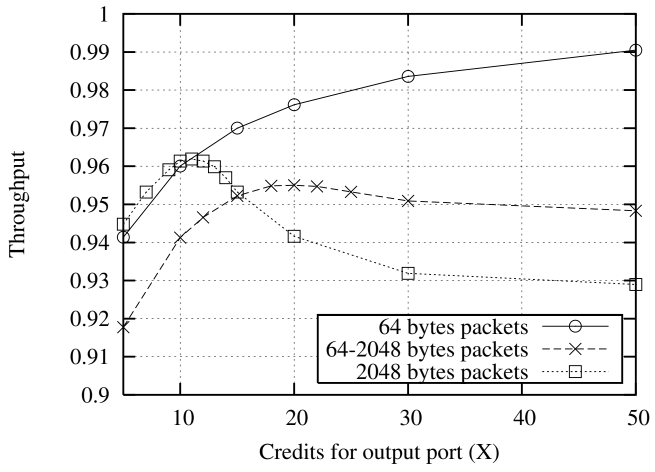

Fig. 3. Throughput under uniform traffic and different packet size distribution

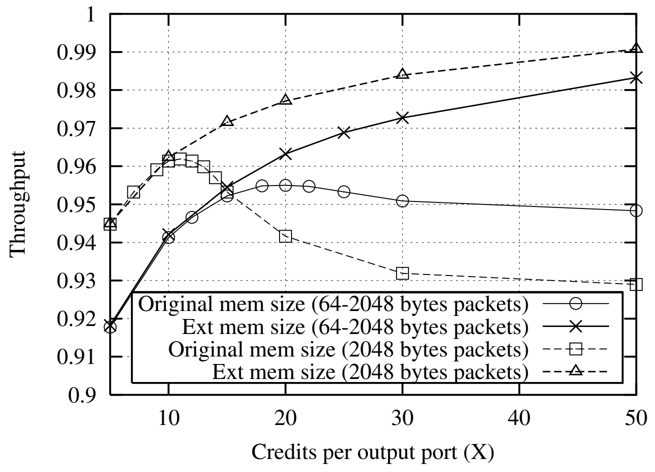

Fig. 4. Throughput in the two scenarios with original and extended memory size

input queues. Indeed, small values of X can shape the incoming traffic to reduce the HOL blocking.

To intuitively understand this fact, it is convenient to consider just the crossbar, with FIFO queues and speedup one. Assume now to allow just only one credit for each output of the crossbar, i.e. at most one packet for each output is allowed to reach the switching fabric, and no contentions occur at the FIFO input queues: hence, the maximum throughput is achieved. On the contrary, if the number of credits is large, the switching fabric behaves as an input queue with a single FIFO, and the throughput for speedup 1 is known to be about 58% [4], under uniform traffic.

In other words, large X increases HOL blocking, thus decreasing the throughput. But, at the same time, large X compensates the control credit loop delay, thus increasing the throughput. Hence, we would expect an optimal value of X for which the throughput is maximized; this expectation will be met by the results of our simulations.

The dimension of the packets plays here an important role on performances. We have highlighted this role by studying three different scenarios, the first with small packets only, the second with large packets only and the third with variable-size packets. Figure 3 shows the throughput performances for different values of X, under uniform traffic scenario and for the three packet length distributions.

1) Performance with small packets: The throughput under uniform traffic is affected by the contention among the packets. Thanks to the small dimension of the packets, the occupation of the fabric queues cannot saturate and the backpressure is never activated, so the throughput curve is monotonically increasing.

2) Performance with large packets: With 2048 bytes packets, it is interesting to observe the the throughput reduction for number of credits larger than 11. This is coherent with the effects of large X, discussed at the beginning of Section III-C. A detailed analysis of the internal states of the switch shows a larger occupation of the fabric queues, which activates more often the backpressure and this decreases the throughput for larger X, as expected.

3) Performance with variable-size packets: Figure 3 shows the performance under uniform traffic and variable-size packets. With respect to the case with fixed-size packets only, the starvation and the backpressure effects are between the case with only small packets and the case with only large packets. This is coherent with the fact that this is a mixed scenario among the other two ones.

In Figure 3, when comparing the final throughput performance for the three cases, for low X both scenarios with fixed-size packets behave the same, and better than the scenario with variable-size packets. This is due to the fact that large packets, having a long transmission time on output links, may hold credits needed by small packets waiting to access the switching fabric. For large X, the variable-size scenario is again between the two fixed-size scenarios.

4) Performance with extended memory size in the switching fabric: We increased the memory size of the fabric queues to avoid backpressure; this scenario is called "extended memory size". Since the arbiter receives at most $X \times P_{LC}$ credits per linecard, in the worst case the maximum occupancy allowed in an output fabric queue will be: $X \times P_{LC} \times MTU$. In this case, $P_{LC} = 16$ and $MTU = 2048$ bytes and we dimension the size of each output fabric queue equal to $32768 \times X$ bytes. Furthermore, we set each input fabric queue equal to half of the output fabric queue, that is equal to $16384 \times X$ bytes.

Figure 4 shows the throughput achieved with the original memory size and with the extended memory size, in the case of variable and fixed-size packets. With

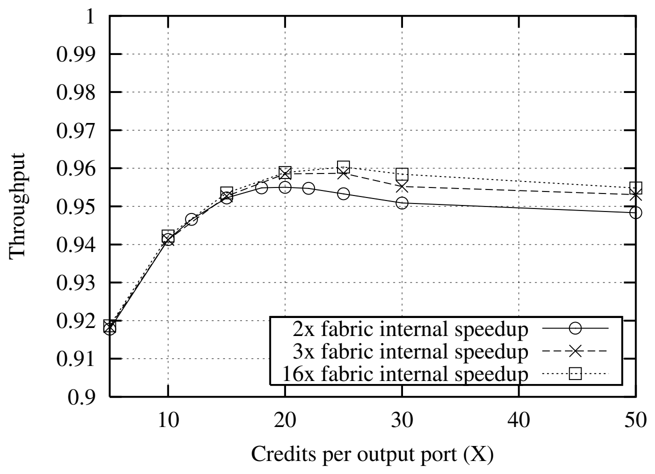

Fig. 5. Throughput in a variable-size packet scenario with increasing internal speedup in the switching fabric

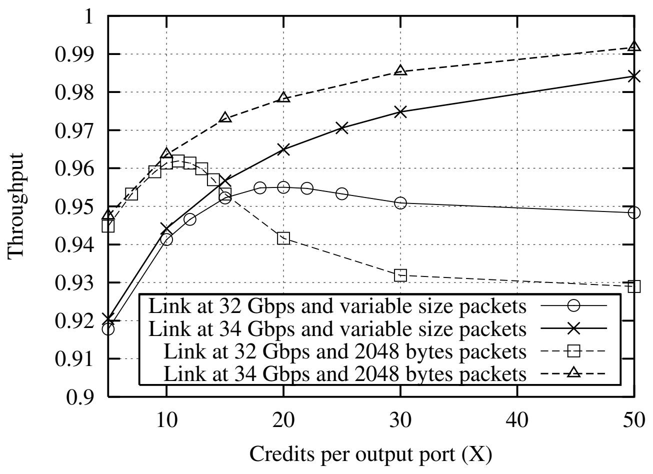

Fig. 6. Throughput with link-speedup between the switching fabric and the linecards

extended memory, the throughput is always increasing with the number of credits for output port. It can be observed that the backpressure is never active and the starvation effect decreases (as expected) with X. The queue occupation is always very small, since the queues are dimensioned for the worst case of large packets only and, of course, this is not efficient for variable-size packets.

5) Performance with increased internal speedup in the switching fabric: When the internal speedup K of the switching fabric is increased, the negative effects of HOL blocking are reduced. The occupancy of output fabric buffers grows and fabric output link utilization becomes higher. However, if $K \gg 1$, fabric output buffers fill up, backpressure blocks packets from fabric inputs and any speedup becomes useless. Figure 5 compares the performance achieved for increasing values of K, in a scenario with uniform traffic pattern and variable-size packets: as the internal fabric speedup increases, the overall throughput improvement remains limited to 1%.

6) Performance with link-speedup between the switching fabric and the linecards: We explored the effects of a link-speedup in the communication rate between the switching fabric and the linecards, while keeping the switching fabric internal speedup $K = 2$.

Figure 6 shows the throughput comparison in the case of communication rate of 32 and 34 Gbps. With a very small link-speedup factor (equal to $34/32 = 1.06$), the throughput asymptotically grows from 0.947 (for 32 Gbps) to 0.992 (for 34 Gbps) with an improvement of about 5%. It is possible to show that the link-speedup decreases the queue occupation inside the switching fabric and the generation rate of the backpressure signals: any throughput reduction is (almost) only due to the starvation.

7) Summary: With respect to the original scheme based on the default parameters, by adding a small link-speedup between the switching fabric and the linecards, it is possible to achieve almost the maximum throughput, for large enough number of credits. On the contrary, an internal speedup larger than 2 is useless. Note that using extended memory size is less efficient than using link-speedup, and it is probably not practically implementable since the memory size inside the switching fabric is usually strictly limited by technological factors.

IV. Conclusions

We presented a high-level study of a switch architecture for SAN applications. The switch employs a credit-based flow-control scheme to guarantee fair access to system resources and backpressure to prevent buffer overflows. The main contribution of our work is the study of the effects of these mechanisms on switch performance. We have discussed the impact of starvation and backpressure on switch throughput and the conditions under which they occur.

Our simulation results illustrate the behavior of the switch in different traffic scenarios and provide guidelines for the dimensioning of system parameters.

References

[1] ANSI INCITS 373-2003, "Fibre Channel - Framing and Signaling (FC-FS)", rev. 1.90, April 9 2003

[2] R.W. Kembel, R. Cummings, *The Fibre Channel Consultant : A Comprehensive Introduction*, Northwest Learning Associates, January 2000

[3] R.W. Kembel, *Fibre Channel Switched Fabric*, Tucson, AZ, USA, Northwest Learning Associates, 2001

[4] Karol M., Hluchyj M., Morgan S., "Input versus output queuing on a space division switch", *IEEE Trans. on Communications*, vol. 35, n. 12, Dec. 1987, pp. 1347-1356