EAST Note 95-05, June 22 1995.

FIBRE CHANNEL PERFORMANCES WITH IBM EQUIPMENT

Carlo Miron, Massimo Bernaschi
ECSEC-IBM, Rome
miron@vaxlnl.lnl.infn.it, massimo@vnet.ibm.com

Fabrice Chantemargue, Susana Munoz
CERN, Geneva
chante@sunrans.cern.ch, munoz@sunrans.cern.ch

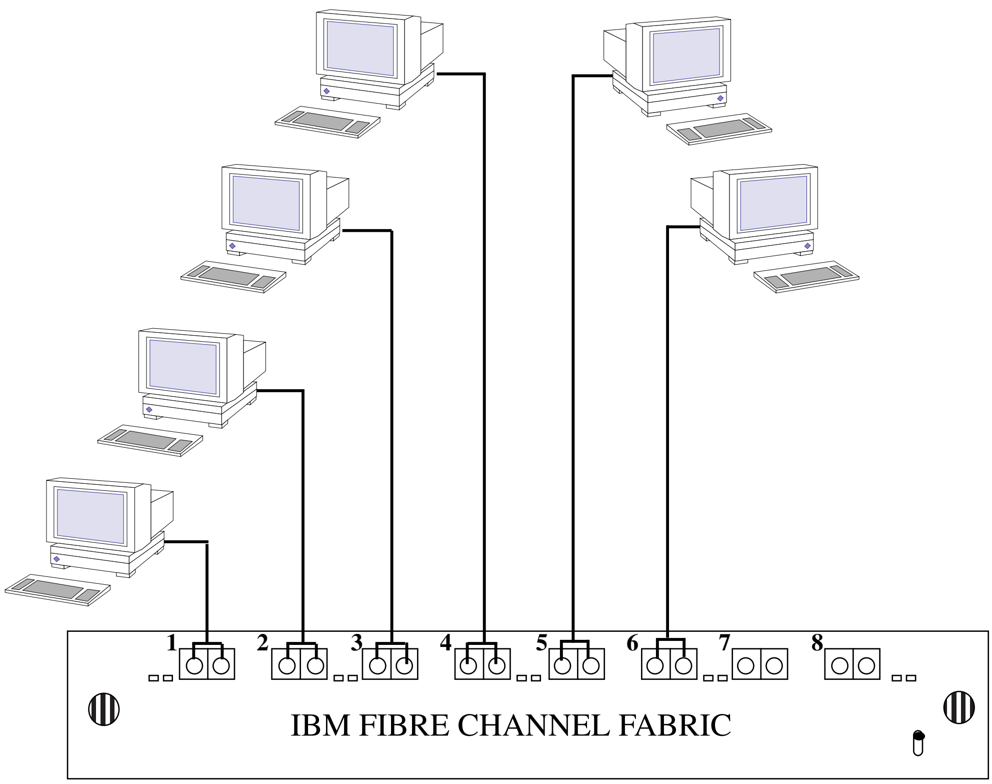
Gaetano Maron
INFN, Legnaro
maron@vxcern.cern.ch

I. Introduction

This work has been done in the framework of High and Low Energy Physics applications (real-time trigger and event building), in which we are concerned in the installation of standard commercial processing nodes interconnected by Fibre Channel communications and switching equipment. High Energy Physics applications need a real-time computing network whose major challenge is exchange of short messages with very low latency, coupled with high performance processors. On the contrary, the new generation of Low Energy Physics experiment has not a critical timing, since the trigger scheme is usually performed by dedicated hardware devices, but needs performant switching capabilities and high bandwidth. These applications are described in paragraphs II and III.

The aim of this work was to measure communication performances at the user level when exchanging data between IBM workstations via an eight port IBM Fibre Channel Fabric (or switch). Similar tests have already been performed at CERN with different kinds of equipment on heterogeneous systems (IBM and HP workstations with different power, ANCOR and IBM Fibre Channel interfaces, ANCOR Fibre Channel Fabric) with different kernel configuration [1], [2], [3] and [4].

Thanks to IBM-ECSEC at Rome, the test with the following homogeneous hardware configuration has been made feasible:



The six workstations were RS/6000-560 machines with a POWER architecture running at 50 Mhz. The IBM Fibre Channel interfaces to the MCA bus (RS/6000 IBM workstation) are 32 bit bus Master capable of data transfer rates up to 40 MBytes/s: they do not represent a bottleneck in the system, where the speed of each 50 micron optic fibre is 266 Mbits/s. The Fibre Channel equipment is implementing version 4.3.

The commercial driver available from IBM [5] for the Fibre Channel adapters offers two different API (Application Programming Interface):

- TCP/IP interface,
- Raw Socket interface.

We performed transmission tests using both APIs and results are reported hereinafter.

II. High Energy Physics application

The High-energy Physics community has decided to build over the coming decade at CERN the future Large Hadron Collider (LHC). This accelerator will operate with two counter rotating proton beams at the highest possible energy. Physics will start with a primary event rate of 10^8 to 10^9 collisions per second, and the task is to find, in real-time, a small number of interesting physics events, not more than a few per second. Each event is characterized by several Megabytes of information, were it to be fully digitized. A succession of event selection algorithms will be applied, in several stages, each more selective, and requiring higher algorithmic complexity, than the previous one. The subdivision into stages, or “levels”, avoids dead time (data jams) as much as possible. In the later stages of event selection these algorithms will be implemented in processors of some generality; close to the detectors they will have to run on custom-made or specifically adapted hardware, with limited or no programmability.
As part of the preparatory program for LHC experimentation, various Research and Development projects have been set up around detectors, associated electronics, and data acquisition. In this proposed joint project, we are directly concerned with the activities in the R&D project EAST (RD-11), presently involved in several pilot projects to demonstrate the critical components of such a selection (“trigger”) system at an input rate of 100 kHz. EAST has evolved the basic Region of Interest concept (RoI) for Second level Triggering. A RoI represents a spatial area in the subdetector, in which the previous level of triggering (i.e. first level) has identified candidates for interesting phenomena to be triggered.
In this scheme, EAST has decomposed the Second level Trigger problem into three different phases, in order to implement it. Phase 1 deals with front-end buffering and RoI collection. Phase 2, named Feature Extraction, is relative to some Local Processing of data inside a RoI of a subdetector. Phase 3 is devoted to Global Decision: Physics features will have to be collected from all subdetectors and from all RoIs in order to allow to take an overall decision on the entire event.
Basic choices exist in expressing the implementation of these three phases as architectures, for which we consider processor architectures based on more or less commercial massively parallel computing elements and high-bandwidth data transmission, on image processing devices, and on FPGA-based architectures. One possible architecture choice is based on the well known farm-approach and will use only standard commercial devices, general purpose processors and network components. A major asset of considering general purpose products (processors and network components) is an easy technology update, not so obvious when considering dedicated products.
In the simplest scheme (see figure 24), there are two layers of processors performing local Feature Extraction (Phase 2) and Global Decision (Phase 3) respectively. Both are managed according to a farm-approach, i.e. individual processors take longer to execute the algorithm than the frequency would require, and have to be sufficiently numerous to allow, by suitable scheduling, to follow the data flow in average. Local processors receive their data from intelligent devices through a switching network. It is assumed in our data model that all data packets have the same length of 1 Kbyte. The global processors receive data from local processors, again through a switch. The data message length here is assumed to be 64 bytes.
Up to now, numerous studies have been devoted to implementing and benchmarking relevant physics algorithms on different processors. As some new products relative to the so-called High Speed Networks emerge on the commercial market, like ATM, Fibre Channel for instance, some studies begin for investigating the potentiality of these new promising technologies, which are all potential candidates and therefore will compete for the final implementation.

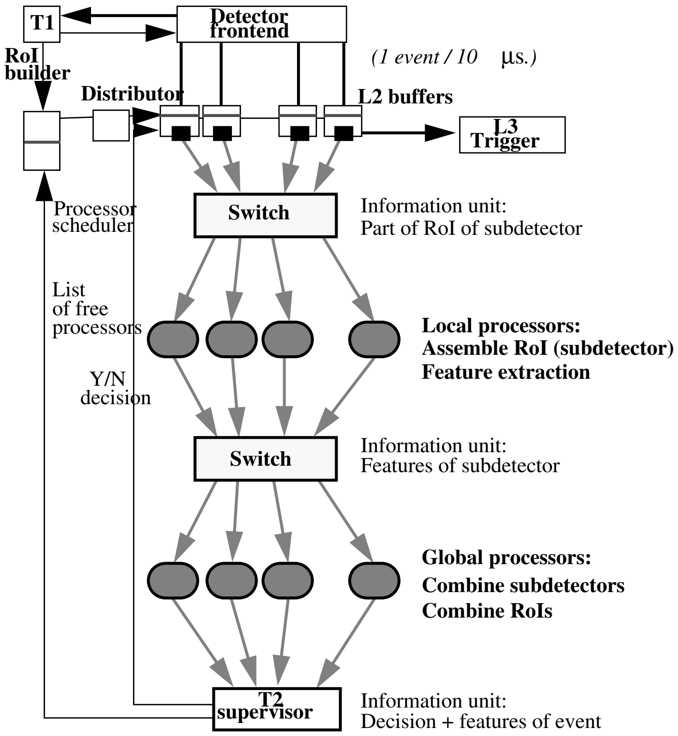


Figure 24: Farm-based implementation.

The aim of all the tests performed at ECSEC-Rome, is to evaluate today what is the performance of such general purpose network components and processors, and to try to extrapolate for tomorrow whether it can be suitable to our application or not, and what performance can be expected. For more details on physics applications, refer to [6], [7], [8] and [9].

III. Low Energy Physics application

A new gamma spectrometer for low energy heavy ions reactions called EUROBALL has been designed by european physicists working in this field. The spectrometer is composed by a sphere of about 300 hyperpure Ge crystals surrounded by anticompton scintillators to detect with an excellent resolution gamma photons. Ancillary detectors can be added to the spectrometer to characterize also the charged and neutral particles produced in the reaction. The experiment has a very simple multiplicity trigger and is able to produce up to 100 KHz of interesting events. A switching network based on Fibre Channel Standard has been proposed, both as event builder and normal communication network for this experiment. The ECSEC test, in this case, is aimed at measuring the point to point bandwidth and estimating the behavior of the switch when it is loaded (i.e. when several communications are performed at the same time). Moreover, as the readout units, which are based on VME boards, do not support TCP/IP protocols, this test will give an assessment on the functionality and the performances of an available light-weight protocol, viz the Raw Socket protocol.

IV. Communication Benchmarks

All measurements, whose aim is to determine overheads, communication latencies and throughputs at the user level, have been achieved on the sender side. We defined overheads, communication latencies and throughputs as follows:

- Overhead: it is defined as the time from when an application sends a message to the time when it is ready to send another message. In practice, it corresponds to the measurement of the time taken for performing the write of a one byte message (the smallest message that can be written).

- Communication latency: it is defined as the time from when an application (on the sender side) sends a message to the time when another application (on the receiver side) receives this message. In practice, we measure a round trip (RT) communication latency for a one byte message, and then divide it by two.

- Throughput (or transmission rate): it is defined as the amount of data that can be sent within one second.

The program used to test out communication performances, when using *TCP/IP* protocols, was *Netperf*. On all machines, we configured the High-Speed TCP/IP Extension (*rfc 1323*) to increase the performance of TCP/IP over Fibre Channel. Other appropriate network options were modified to fully utilize the advantages of *rfc 1323,* like for instance the maximum socket buffer size (sb_max), the window size on both receiver and sender sides (tcp_sendspace, tcp_revcspace), ... [see appendix on how to configure network options].
The *TCP_no_delay* option (*-D*) was set on the sender side: it is necessary for relevant measurements because it disables the so-called Nagle mechanism for the buffering (it avoids buffering and grouping messages before sending). Sometimes, we got some troubles when using this option turned on and we were obliged to turn it off (only in case of sending large messages with large socket buffers).

For the light-weight protocol (Raw Socket), no program was available from the public domain; therefore we used and ran an IBM hand-made application, which was evolved jointly by Carlo Miron and Massimo Bernaschi.

A. Using TCP/IP protocols

The first experiment was aimed at estimating the latency introduced by the IBM Fibre Channel Fabric. For that, we measured overheads, communication latencies and throughputs for a large number of message sizes when two workstations were back to back connected and when the Fabric was located in-between.
The second experiment was aimed at estimating the scalability of the fabric when performing several communications in parallel.

1. Estimation of the IBM Fibre Channel Fabric Latency

Figures 1 and 2 (left) display the performances for transferring data from memory to memory with and without the Fabric when using two different socket buffer sizes (1 Kbytes, 2 Kbytes).
Figures 2 (right), 3, 4 and 5 display the performances for writing data on the sender side with and without the Fabric. For small messages (less than 1Kbytes), the socket buffer size was set to 1 Kbytes for this test. For large messages, the socket buffer size was set to 1 Mbytes and the -D option (TCP_nodelay) was set off.

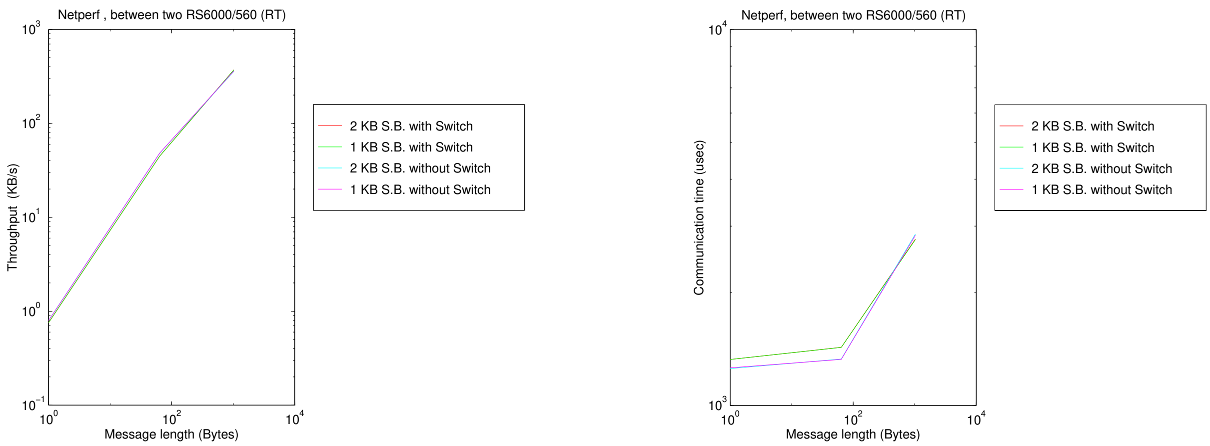


Figure 1: From memory to memory: (left) Throughput, (right) Communication latency.

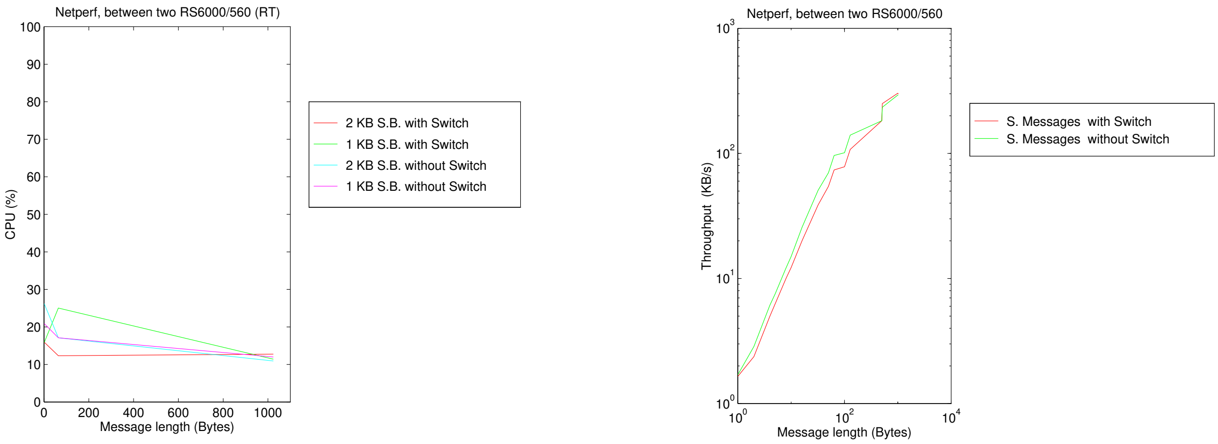


Figure 2:(left) CPU (From memory to memory), (right) Throughput (small messages).

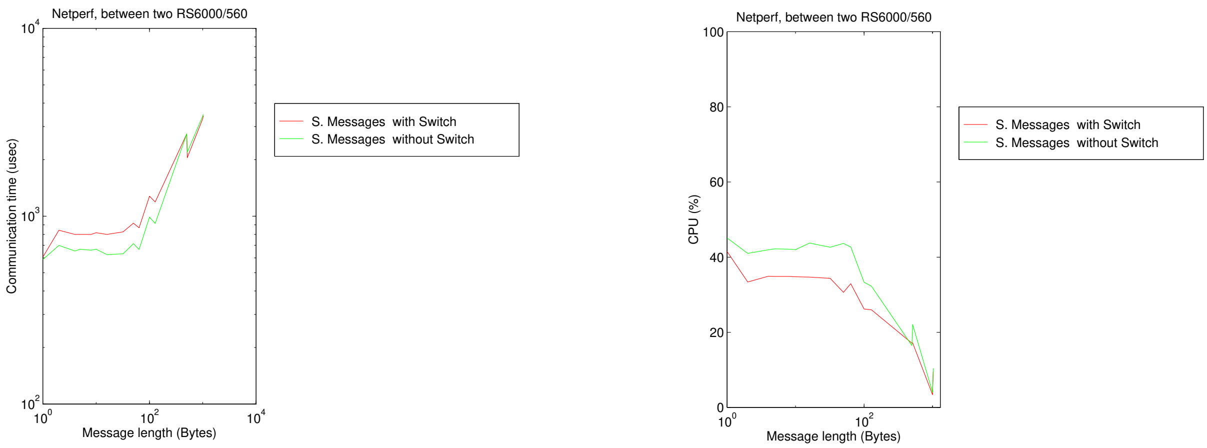


Figure 3: (left) Overhead (small messages), (right) CPU (small messages).

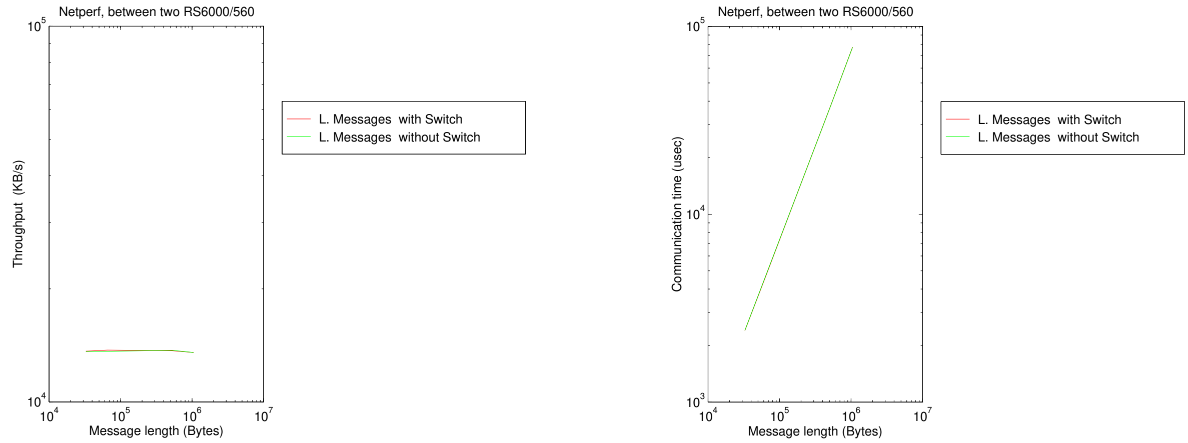


Figure 4: (left) Throughput (large messages), (right) Overhead (large messages).

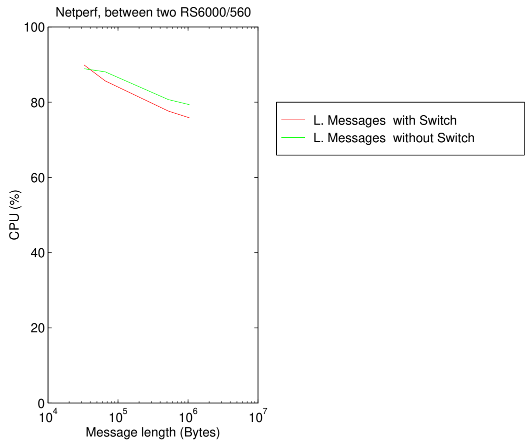


Figure 5: (CPU) (large messages).

From memory to memory performance measurements (see figure 1 (right)), we can infer the following Fabric latency:

66 µs (for 1 byte), 101 µs (64 bytes) and -73 µs (1024 bytes).

The last value (the negative one) is strange and difficult to explain but is found consistently. In other words, it means that when the Fabric is located between two workstations, it improves the communication performance for messages greater than 512 bytes.

Hereinafter is the summary of the critical performances that have been measured:

- Communication latency for 1 byte: 1325 µs (CPU=16%) with the switch
- Communication latency for 1 byte: 1259 µs (CPU=21%) without the switch
- Overhead for 1 byte: 606 µs (CPU=41%) with the switch
- Overhead for 1 byte: 588 µs (CPU=45%) without the switch
- Throughput for 64 Kbytes: 13.6 Mbytes/s (CPU=90%) with the switch
- Throughput for 64 Kbytes: 13.6 Mbytes/s (CPU=89%) without the switch

For large messages, there is practically no difference in performances when the communication is performed with or without the Fabric. On the contrary, for small messages, the best performances are measured for back to back communications.

The influence of the switch is that:

- it introduces a degradation in performance for small messages and improves the performance for messages greater than 512 bytes.

- in most cases, it permits to reduce the percentage of CPU used on the sender side for performing communications.

2. Measurements when the Fabric is loaded

The aim of this test was to estimate the behavior of the fabric when several communications were achieved in parallel. This test has been performed when using four or six workstations. Either one or two couples of workstations were devoted to the load of the fabric: we tested two kinds of load. In the first test, either one or two couples of workstations were exchanging in uni-directional way large messages (1 Mbytes) using large socket buffer sizes (1 Mbytes). In the second test (done with only four workstations, due to machine availability problems), the load consisted of transmitting small messages (64 bytes) using small socket buffer sizes (1 Kbytes). In parallel between the two remaining workstations, we re-executed the same kind of test as described in the previous section (estimation of overhead, communication latency and throughput).

a) First type of load (large messages)

Performances issued from the load processes were constantly checked during the test and as no drop has been noted, they are not represented.
Figures 6 and 7 (left) display the performances for transferring data from memory to memory versus the message length, according to the number of loads. The socket buffer size for this test was set to 1 Kbytes.
Figures 7 (right), 8, 9 and 10 display the performances for writing data on the sender side versus the message length, according to the number of loads. For small messages (less than 1Kbytes), the socket buffer size was set to 1 Kbytes. For large messages, the socket buffer size was set to 1 Mbytes and the -D option was set off.

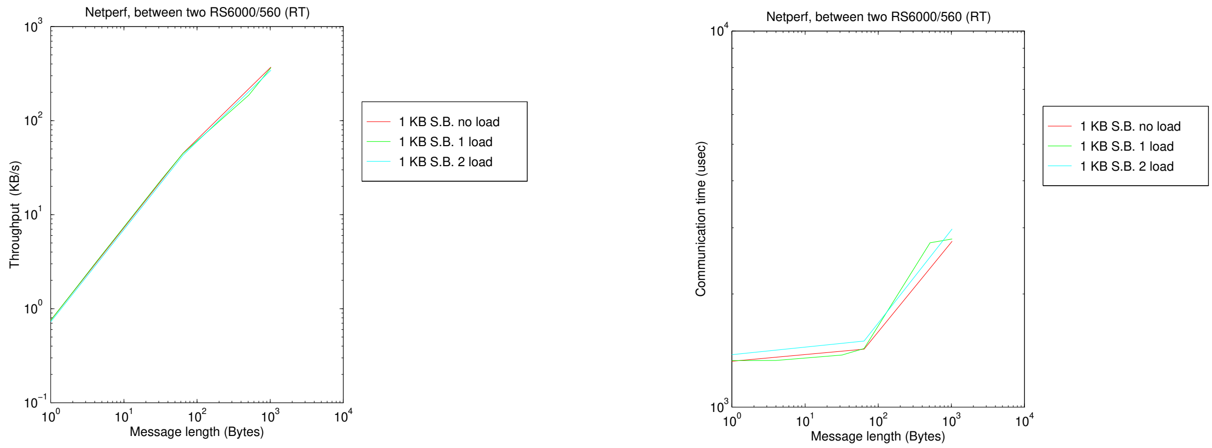


Figure 6: From memory to memory: (left) Throughput, (right) Communication latency.

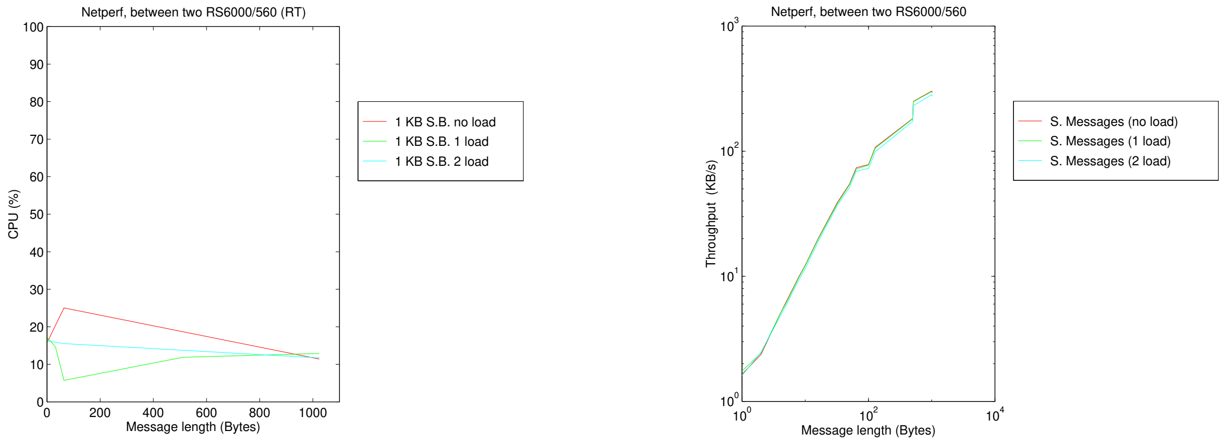


Figure 7:(left) CPU (From memory to memory), (right) Throughput (small messages).

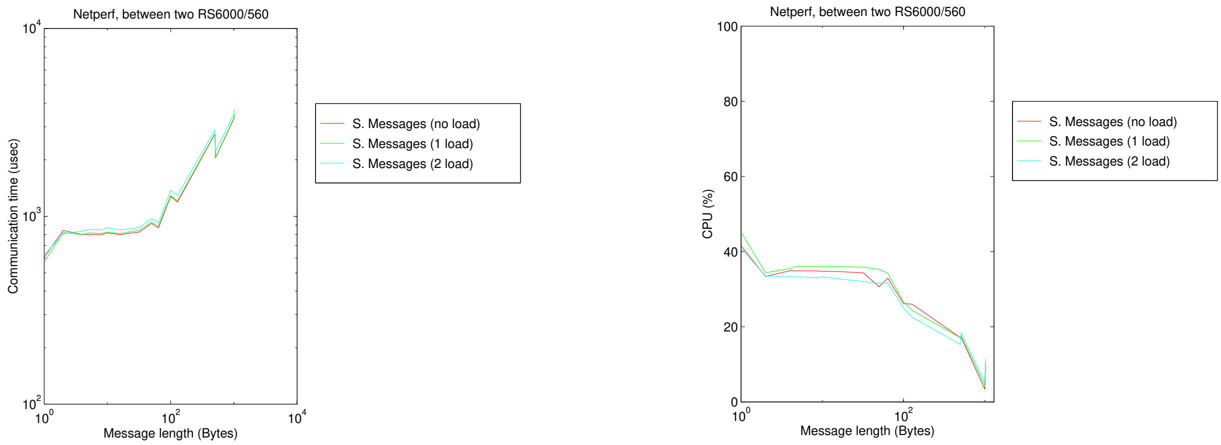


Figure 8: (left) Overhead (small messages), (right) CPU (small messages).

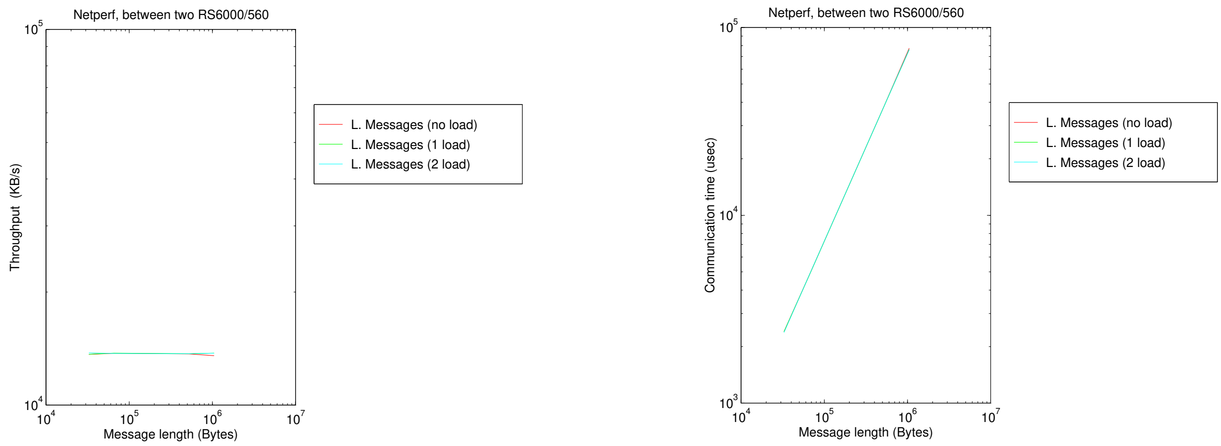


Figure 9: (left) Throughput (large messages), (right) Overhead (large messages).

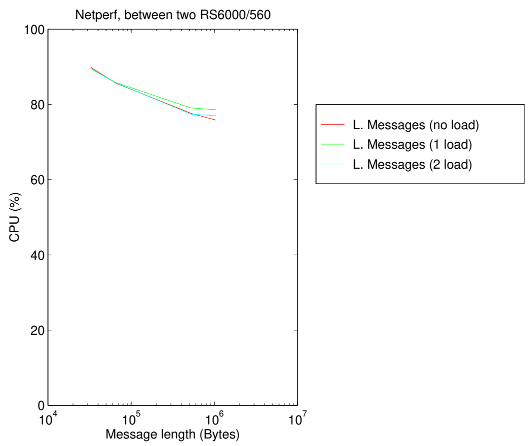


Figure 10: (CPU) (large messages).

The load, when it consists in sending large messages, does not degrade the performance on other communications, as long as these communications are made of large message exchanges. When they are made of small message exchanges, a small degradation (few percents) has been reported. As this degradation is weak, it is difficult to be sure that this is not resulting of statistical fluctuations.

b) Second type of load (small messages)

Figures 11 and 12 (left) display the performances for transferring data from memory to memory versus the message length, according to the number of loads. The socket buffer size for this test was set to 1 Kbytes.

Figures 12 (right), 13, 14 and 15 display the performances for writing data on the sender side versus the message length, according to the number of loads. For small messages (less than 1Kbytes), the socket buffer size was set to 1 Kbytes. For large messages, the socket buffer size was set to 1 Mbytes and the -D option was set off.

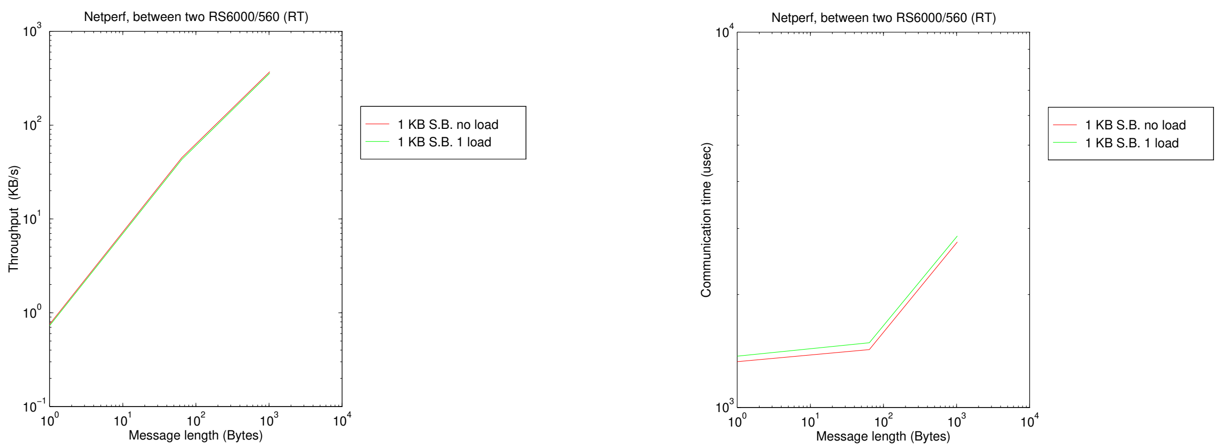


Figure 11: From memory to memory: (left) Throughput, (right) Communication latency.

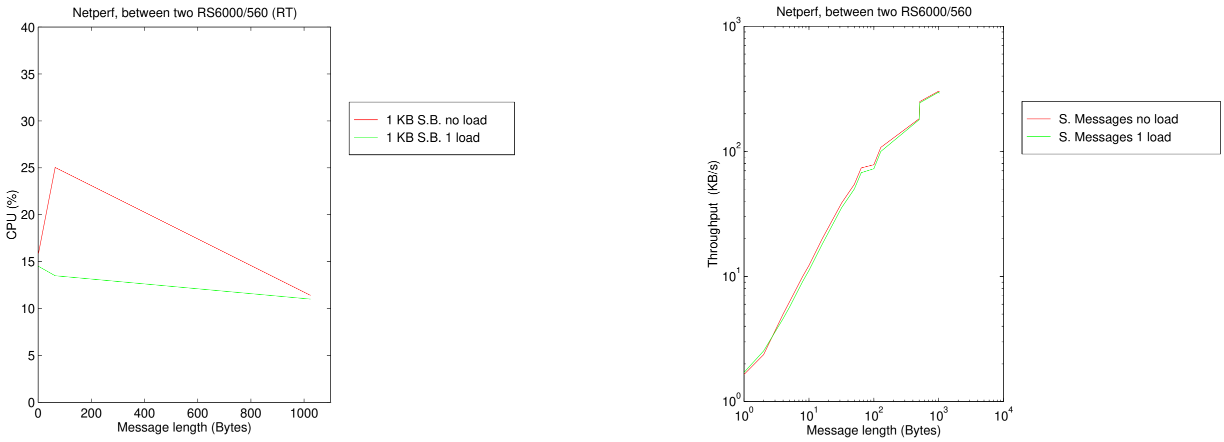


Figure 12:(left) CPU (From memory to memory), (right) Throughput (small messages).

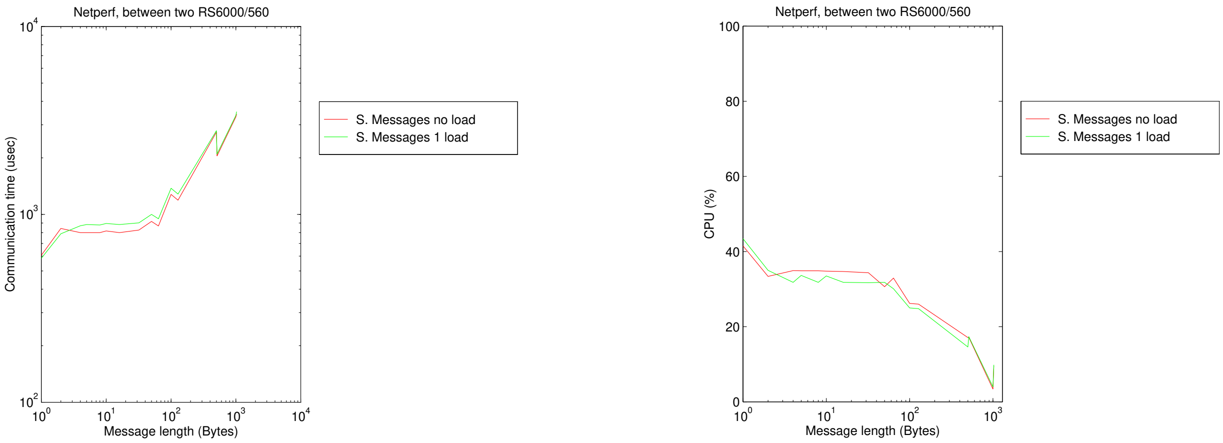


Figure 13: (left) Overhead (small messages), (right) CPU (small messages).

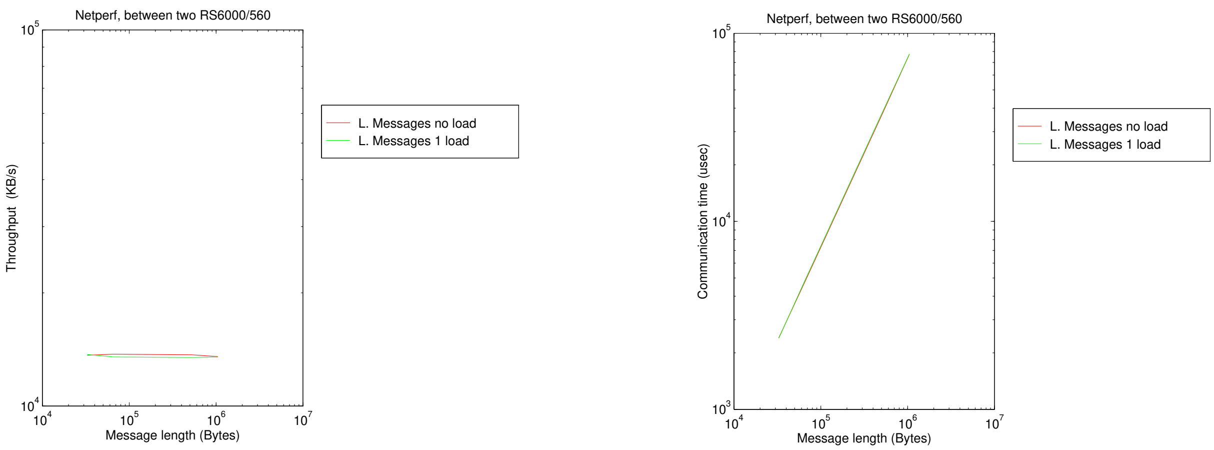


Figure 14: (left) Throughput (large messages), (right) Overhead (large messages).

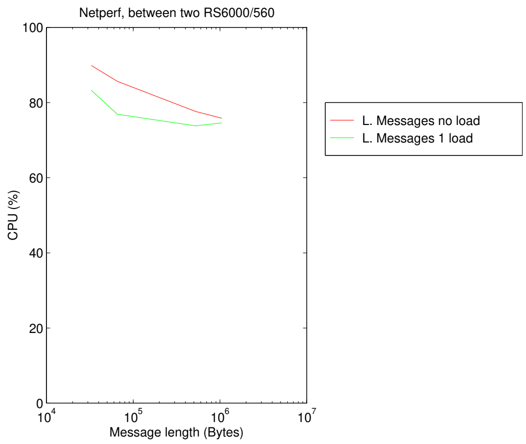


Figure 15: (CPU) (large messages).

Figure 16 displays the degradation that has been reported on the load communications, versus the duration of the test, according to the type of test that was running in parallel.

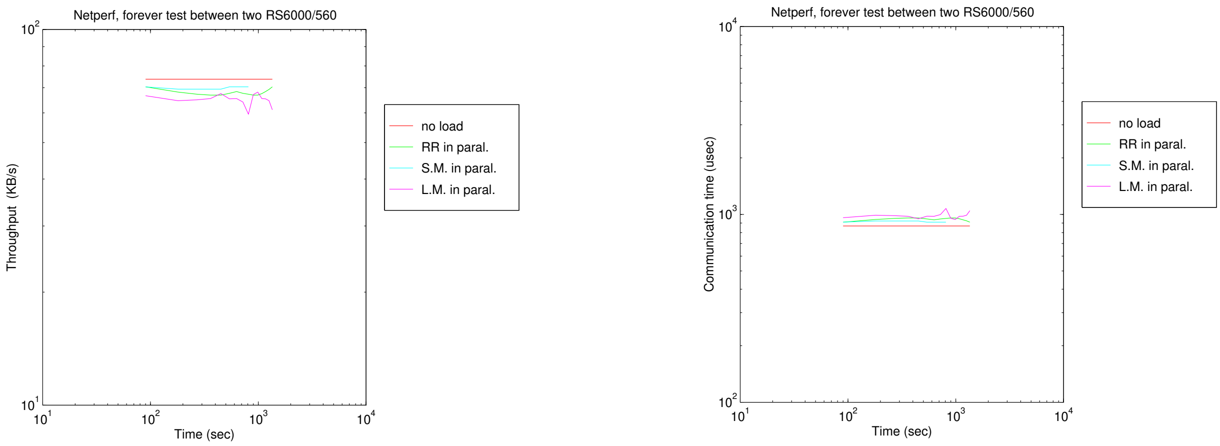


Figure 16: (left) Throughput on the load, (right) Overhead on the load.

When the load is made of the transmission of small messages, we pointed out a degradation on all communication performances of around 5-10 percent.

B. Using Raw Socket: a light weight protocol

This test is similar to the one described in paragraph IV-A, i.e. it is aimed at measuring communication benchmarks, but now when using Raw Socket protocol. With this protocol, the user is responsible for the good delivery of packets. All the tests have been achieved with the following convention: one acknowledgment is sent back after reception of 16 packets (messages).

As before, we performed this test when using two, four or six workstations (two workstations for measuring performances without any interference due to other connections, but with the fabric in-between). For the test including the loading, either one or two couples of workstations were devoted to the load of the fabric: we tested the same two kinds of load as described in paragraph IV-A-2, both implemented with TCP/IP communications. Let us just briefly recall the tenet of the loads. In the first test, either one or two couples of workstations were exchanging in uni-directional way large messages (1 Mbytes) using large socket buffer sizes (1 Mbytes). In the second test (done with only four workstations, due to technical problems), the load was consisting in transmitting small messages (64 bytes) using small socket buffer sizes (1 Kbytes).

a) First type of load (large messages)

Performances issued from the load processes were constantly checked during the test and no drop has been noted. That is the reason why they are not represented.

Figures 17 and 18 display the performances (throughput, overhead and CPU utilization) versus the message length, according to the number of loads.

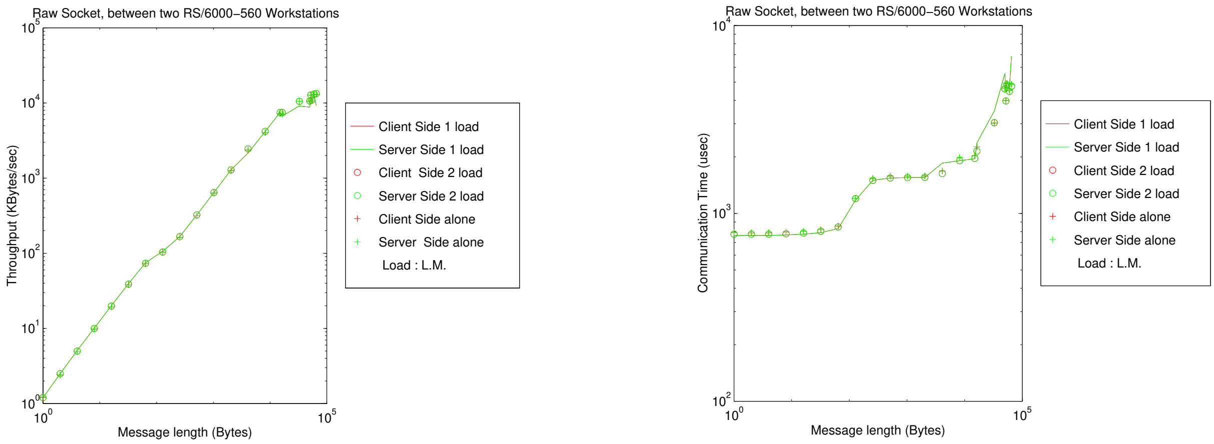


Figure 17: (left) Throughput, (right) Overhead.

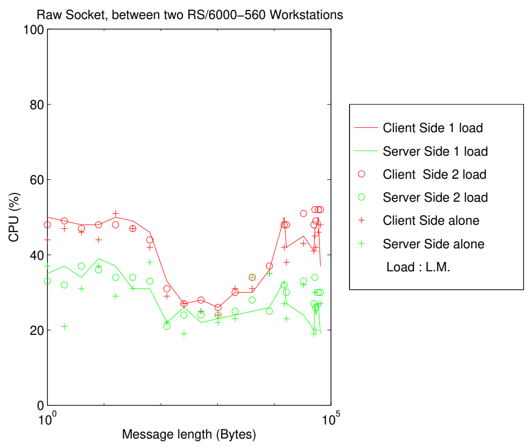


Figure 18: CPU.

Hereinafter is the summary of the critical performances that have been measured without any load:

- Overhead for 1 byte: 780 µs (CPU=44 %)
- Throughput for 65000 bytes: 13.1 Mbytes/s (CPU=48%)

As seen before in paragraph IV-A, the load, when it consists in sending large messages, does not degrade the performance on other communications, as long as these communications are made of large message exchanges. When they are made of small message exchanges, a small degradation (few percents) has been reported. As this degradation is weak, it is difficult to be sure that this is not resulting of experiment fluctuations.

b) Second type of load (small messages)

Figures 19 and 20 display the performances (throughput, overhead and CPU utilization) versus the message length, according to the number of loads.

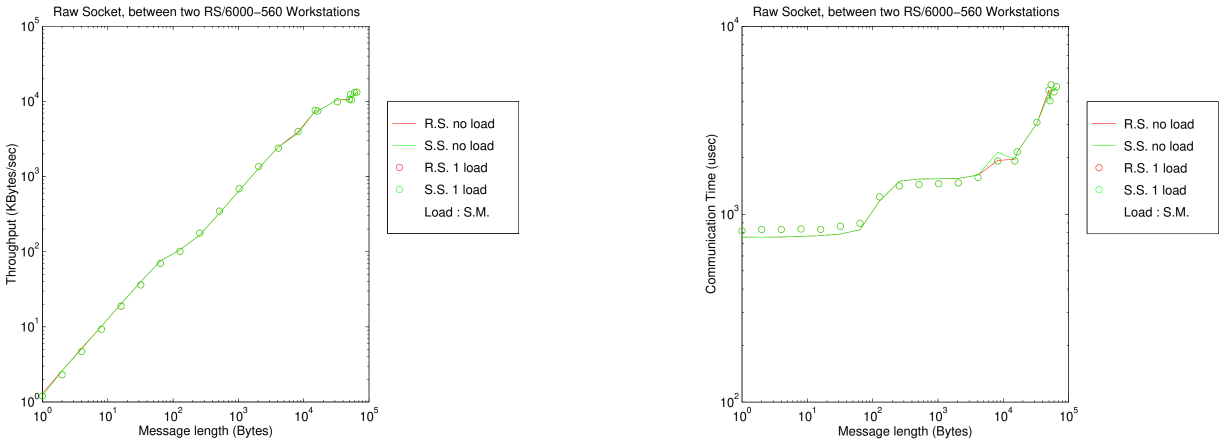


Figure 19: (left) Throughput, (right) Overhead.

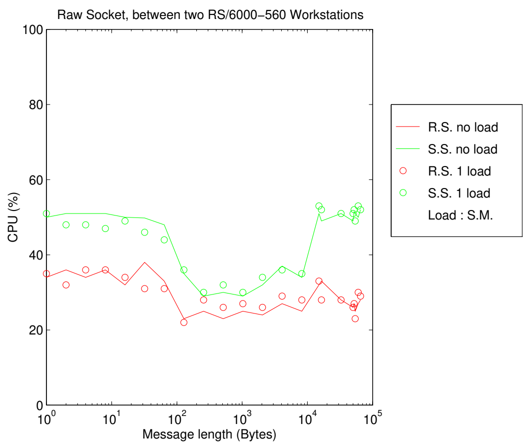


Figure 20: CPU.

When the load is made of the transmission of small messages, we pointed out a degradation on all communication performances of around 5-10 percent.

Figure 21 displays the degradation that has been reported on the load communications versus the duration of the test, when the Raw Socket test was performed in parallel.

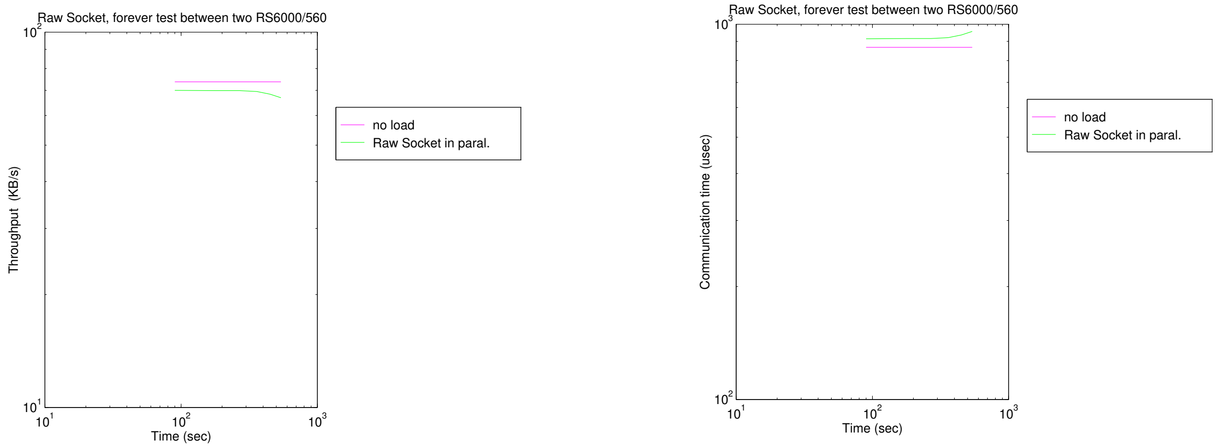


Figure 21: (left) Throughput, (right) Overhead.

c) Influence of the number of packets received before acknowledged

When using the Raw Socket protocol, it is up to the user to check the good delivery of packets during a transmission. Figures 22 and 23 show the performance versus the message length, according to the number of packets received before acknowledged (1, 8, 16, 24, 32).

Let us just recall that all the tests previously reported with Raw Socket protocol have been performed with one acknowledgment every 16 packets.

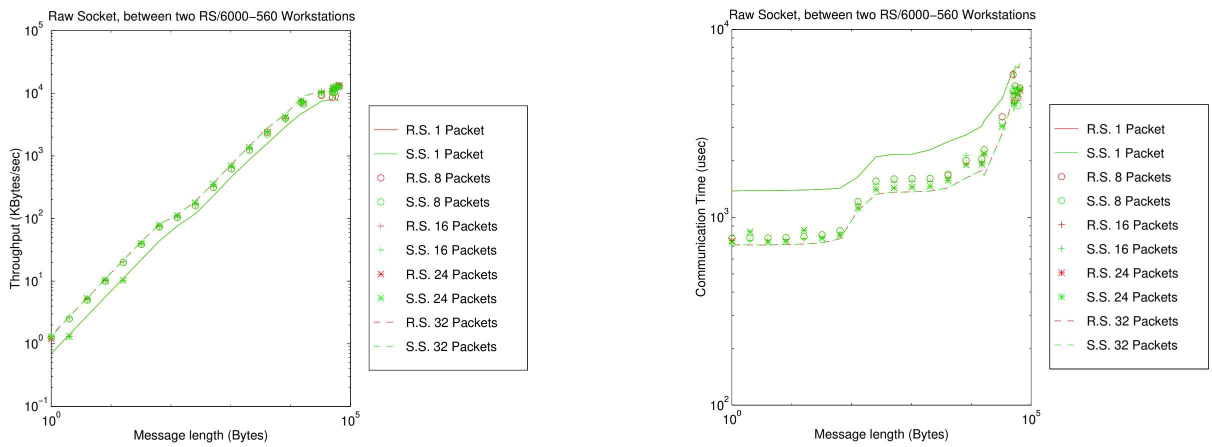


Figure 22: (left) Throughput, (right) Overhead.

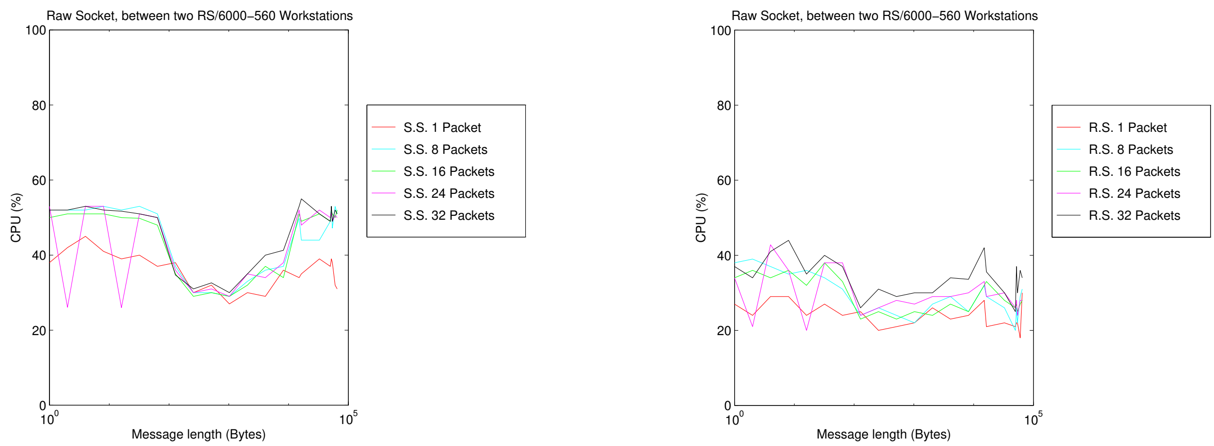


Figure 23: (left) CPU on the sender side, (right) CPU on the receiver side.

V. Concluding Remarks

Conclusions that can be inferred from these measurements are not the same according to the kind of Physics application.

For application in 100 KHz trigger, the conclusion of these measurements is that today, Fibre Channel does not satisfy our application's requirements. We measured a minimum overhead for a parallel application in which processing tasks will interleave with communication processes. Assuming zero processing time, we can state for the combination of hard- and software under scrutiny (based on general purpose workstations like IBM RS/6000-560s interconnected to a Fibre Channel Fabric) a maximum aggregate working frequency of around 1.5 Khz (it corresponds to an overhead of around 600 µs. measured with TCP/IP). It is also important to stress the fact that the communication performances are correlated with the power of the host workstation inside which the Fibre Channel adapter is located. Working with IBM RS/6000-590s, we already measured an overhead reduced to 375 µs [5]. In that case, it would imply a working frequency of 2.7 KHz.

For Low Energy Physics applications, the conclusions are more optimistic, since the bandwidth that was measured for a point to point communication satisfies the requirements of the experiment. This experiment will create indeed very short events (about 500 bytes with fragments of about 50 bytes), which can be grouped into bigger events (called super events) allowing to fully exploit Class 1 service of the Fibre Channel Standard (dedicated connection). Connectionless services (Class 2 and Class 3) could help the design of the software and probably make more efficient the use of the switch, but unfortunately it was not possible to test it with the equipments and the software that was available.

The overhead is a critical issue for applications in which small messages have to be transmitted, like for instance in our High Energy Physics application (remember that our largest message is only 1 Kbyte). We will not use TCP/IP for our eventual implementation, but some light-weight protocol, expected to improve considerably the overhead (mostly for High Energy Physics) and the bandwidth (for both applications). Using Raw Socket as protocol, we were a bit disappointed, because we did not observe any improvement over TCP/IP, and moreover, this protocol did not seem to be very robust. Of course, the Raw Socket application we used was a preliminary version and we can expect a performance improvement in the next release, since Raw Socket is by-passing TCP and IP layers. Improvements are mandatory for both physics applications. The only advantage in favour of the Raw Socket protocol (versus TCP/IP), at that time, is that the CPU utilization is half when transmitting large messages: for instance for sending around 64 Kbytes, the CPU utilization was around 90% with TCP/IP and less than 50% with Raw Socket.

When several independent communications were performed in parallel (loading of the switch of approximately 30 percent), we observed a maximum degradation of 5-10 percent on the performance of each communication when it was relative to the transmission of small messages. In case of large messages, no degradation was measured. Such a test should be extended to a fully loaded switch.

We would like to have confidence in the future evolution of the technology for improving considerably performances. This seems easier for high throughput, more speculative for hypothetical very low overhead. For short term future, commercial products like Fibre Channel may more typically be needed in general-purpose applications, for which high throughput is of greater interest than low overhead.

VI. References

[1] Fabrice Chantemargue: “Communication benchmarks with the Ancor Fibre Channel Fabric”, CERN/EAST note 94-22, July 1994

[2] Fabrice Chantemargue: “Communication benchmarks with HPs and IBMs connected around an Ancor Fibre Channel Fabric”, CERN/EAST note 94-24, August 1994

[3] Fabrice Chantemargue: “High Performance TCP/IP communication benchmarks with HPs and IBMs connected around an Ancor Fibre Channel Fabric”, November 1994

[4] Fabrice Chantemargue, Susana Munoz, Gaetano Maron: “Fibre Channel Equipment under Test”, March 30 1995.

[5] Fibre Channel Adapter/266, User’s Guide and Reference, IBM.

[6] Rudy Bock & al.: “Status Report: Embedded Architectures for Second-level Trigger (EAST)”, CERN/DRDC 94-20, 9 May 1994.

[7] CMS Technical Proposal, CERN/LHCC/94-38.

[8] ATLAS Technical Proposal, CERN/LHCC/94-43, 15 December 1994.

[9] Rudy Bock & al.: “Programmable active memories in real-time tasks”, Nucl. Inst. Meth., A 356 (1995) 457.

Some notes are available on the **Web** in html and poscript formats under the following location:

- notes [1], [2], [3] and [4]: **http://www.cern.ch/HSI/fcs/applic/rd11.htm**.
- note [6]: **http://www.cern.ch/RD11/eastnotes/EASTNOTES.html**.
- note [7]: **http://cmsinfo.cern.ch/cmsinfo/Welcome.html.**
- note [8]: **http://www.cern.ch/pub/Atlas/TP/tp.html.**

VII. Appendix

The example below represents a script that can be used to set all the appropriate network options to fully utilize the advantages of rfc 1323 (see [5]).

```
/etc/no -o sb_max=6000000 -o rfc1323=1 -o thewall=6000 -o lowclust=200 -o mb_cl_hiwat=2000
/etc/no -o tcp_sendspace=655360
etc/no -o tcp_revcspace=655360
etc/no -o udp_sendspace=655360
etc/no -o udp_revcspace=655360
```