Characterizing the Real-Time Behavior of Prioritized Switched-Ethernet[1]

Paulo Pedreiras, Ricardo Leite, Luis Almeida

DET-IEETA, University of Aveiro
Aveiro, Portugal
pedreiras@det.ua.pt, a17315@alunos.det.ua.pt, lda@det.ua.pt

Abstract

Ethernet is, today, the de facto standard in the Local Area Networks general domain. Despite having been designed for the office environment, it has been appropriately modified or adapted in order to fulfill the requirements of many other fields, including large distributed embedded systems and industrial automation. One typical requirement in such application fields is the need to deliver time-constrained communication services, which cannot be efficiently met using the original CSMA/CD medium access control. Among many possible solutions, either software or hardware-based, the one that became more popular, recently, is the use of switches. However, this does not enforce timeliness per se. In this paper we show a set of practical experiments that reveal the weaknesses of switched Ethernet in what concerns real-time behavior. The results point to the need for further traffic control, at the data sources, if a predictable behavior of the switch is desired.

1. Introduction

During the past two decades, several communication protocols were designed specifically for distributed computer control systems (DCCS), found either in process control, factory automation or large distributed embedded systems, to interconnect sensors, actuators and controlling equipment. These have been generally called fieldbuses, an expression probably inherited from the process control domain. However, a considerable effort was also devoted to analyze the possibility of using in such application domains general-purpose protocols employed in other areas (e.g. Ethernet, ATM, FDDI). This effort has been justified mainly with the difficulties that traditional fieldbuses have in supporting the growing bandwidth demand felt in some DCCS applications and in supporting a seamless vertical integration in more complex systems composed by layered network architectures [2, 1, 5, 7].

Particularly, Ethernet has been received a considerable and growing interest from the scientific and industrial communities. Despite not being originally designed to deliver time-constrained communication services, some of its properties, such as its easy integration with Internet, inherent compatibility with the networks used at higher levels in hierarchical industrial systems, and low price, make its use very appealing [2, 1, 7]. However, its destructive and non-deterministic arbitration mechanism (CSMA/CD) remains the main obstacle for the required timeliness support [2].

In the quest for achieving real-time behavior in Ethernet several approaches and techniques have been experienced, which range from modifications of the medium access control to the addition of transmission control layers above Ethernet [6]. More recently, the use of switches has been advocated as a solution to this problem by creating a collision domain per station and thus avoiding collisions. However, switches also have limitations that may lead to unpredictable performance, jeopardizing their real-time capabilities.

The aim of this paper is to expose such weaknesses of switched Ethernet raising the awareness for those misbehaviors. For this purpose, several practical experiments are carried out, based on a common off-the-shelf Ethernet switch. Particularly, we evaluate the impact on real-time traffic caused by memory exhaustion in port queues, by blocking due to lower priority traffic, by the low number of different priority levels and by the switch processing load.

2. Switched Ethernet

Ethernet switches provide a private collision domain for each one of their ports. Thus, unless induced intentionally for management purposes (e.g. flow control), collisions are completely avoided as long as full-duplex communication is used. Moreover, IEEE 802.1p gives switches the ability to prioritize messages (up to 8 distinct traffic classes). Another interesting standard is IEEE 802.1q that allows creating virtual LANs (VLANs) within the same switch, supporting traffic isolation between logically different networks. This feature is particularly important to constrain broadcast and multicast packets to the respective VLAN, not affecting other VLANs.

[1] This work has been partially supported by the European Union, under contract IST-2001-34820 (ARTIST).

Unfortunately the use of a switch in an Ethernet network, by itself, is not enough to make it real-time in the general case. For instance, if the traffic is sent to an output port at a higher rate than its capacity, messages must be queued. If queuing occurs in an unbounded way, the switch memory may be exhausted, causing message losses. In DCCS applications, this situation can occur more often than desired, caused by the use of broadcast message transmission associated to the typical producer/ consumer co-operation model. This perturbation is usually referred to as broadcast storm. Another important problem concerning the use of switched Ethernet is the lack of enough priority levels to support efficient priority-based scheduling [2]. The impact of network topology and message scheduling strategies inside the switch has also been recently addressed [4]. Besides those problems, it is also important to remember that switches are based on hardware and software components that have their own limitations, such as switch fabrics and CPU bandwidth and memory organization and size (fig 1).

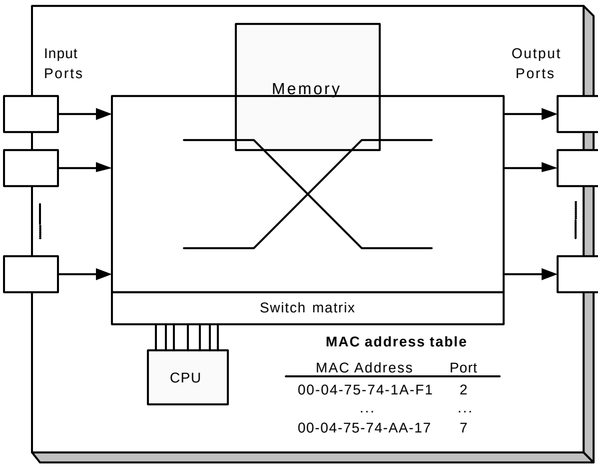

Figure 1 – Usual switch internal architecture.

3. Experiments

The experimental set-up consists on an Ethernet switch, Allied Telesyn model AT-8024, and 5 Personal Computers with CPU's ranging from Pentium MMX at 200MHz to Pentium III at 500MHz. The network adapters were configured to 10Mbps. The traffic generated used a mixed between unicast and broadcast addresses. The former ones were, however, unknown for the switch in order to cause flooding. This allowed increasing the traffic load in the switch while maintaining the possibility of filtering packets in the stations. All stations generating periodic traffic were running the SHaRK [3] real-time operating system. Other stations used to generate non-strictly periodic load were running Windows. The temporal measurements were carried out in a monitoring station running SHaRK with an instrumented network driver that time stamped the received packets with microsecond resolution.

The switch was configured so that all management protocols were disabled. Moreover, flow control was also disabled in the switch as well as in all stations. This switch supports 802.1p with two priority levels, only.

Along the remainder of the text, the expressions *packet* and *message* will be used interchangeably.

3.1 - Precision of measurements

Either the time stamping as well as the periodic generation of real-time traffic was, however, influenced by the limitations of the respective station and of the SHaRK operating system. These could also insert extra delays and jitters, either in packet generation as well as in reception. Therefore, a very simple network setup was used in order to determine that influence and consequently, the precision of the subsequent measurements. Thus, the generator and the monitoring stations were connected to each other via a hub, and the network load was that described in table 1.

Table 1 – Load for experiment 3.1

VLAN	Mesg	Add	Data size	Period	
1	1	Brd	100B	1ms	*

* monitored stream.

In this circumstance, the hub should not introduce any significant jitter and thus, whatever is measured should be caused by the stations, only. The measured jitter was low, as shown in table 2 (150000 samples), except for very rare larger variations (<36μs). Figure 2 shows the histogram of the measured periods.

Table 2 – Results for experiment 3.1

Period (μs)				Relative jitter (μs)			
Min	max	avg	std	min	max	avg	Std
989	1012	1000.1	0.97	0	36	1.09	1.13

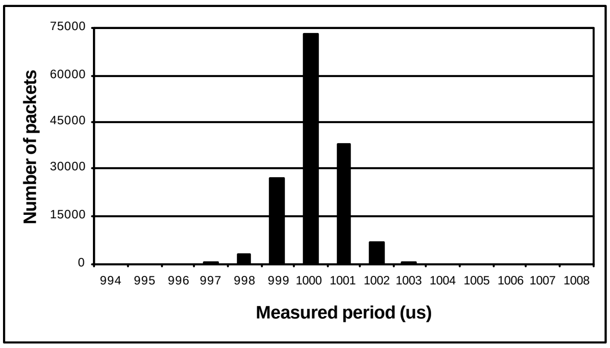

Figure2 - Distribution of measured periods using the timestamps of consecutive packets (experiment 3.1).

3.2 - Switch latency

The same experiment as the previous one was carried out but using the switch. The results were very similar to those in table 2 showing that a switch, in the conditions referred before, does not introduce additional jitter.

The latency was measured with both a hub and the switch, using a pair of pulses in the stations parallel ports, one set in the generator right before issuing the transmission request and another in the monitoring station, just before the reception time stamping. This global latency, which included queuing/dequeuing delays, interrupt latency, transmission time and hub/ switch latency, was then measured in an oscilloscope. With the hub this value varied between 188µs and 200µs while with the switch the variation was from 200µs to 210µs (+/-0.5µs in both cases). The difference between both situations is compatible with the cut-through message forwarding technique used in the switch, that requires queuing the Preamble, Start of Frame delimiter and Destination address (14 bytes) before starting to forward the message. Those 14 bytes take 11.2µs to be transmitted, which are the cause for the additional delay induced by the switch with respect to the hub.

3.3- Interference between VLANs

Although from a conceptual point of view the traffic in distinct VLANs should be completely isolated, the switch CPU, memory and switch fabric are shared resources, and thus some degree of interference is expected. In this context, two sets of experiments were carried out. In the first one, VLAN 2 was added to the traffic in table 1, as described in table 3, causing a strong overflow in internal queues.

Table 3 – Load for experiment 3.3A

VLAN	Mesg	Add	Data size	Period
1	1*	Brd	100B	1ms
2	2	Uni	46-1500B	**
2	3	Uni	46-100B	**
2	4	Uni	46-1000B	**
2	5	Uni	500-1400B	**

* monitored stream ** continuous tx

The results obtained are still similar to those in table 2, showing that message 1 did not suffer any significant interference from the overload in VLAN 2. The reason is that the traffic in VLAN 1 did not require queuing, thus not requesting access to the switch internal memory and, consequently, not suffering from the overload. However, a very small increase in the average jitter from 1.09µs to 1.12µs shows that there is still a small interference caused by the extra CPU load incurred by the larger traffic handling effort. This effect is, nevertheless, negligible when compared with the impact of other factors as those shown in the following experiments.

The second experiment used a higher load in VLAN 1 (table 4), so that the traffic in this VLAN would also require queuing. It was carried out in two parts, firstly with VLAN 1 only, and then with VLANs 2 and 3.

The results with VLAN 1, only (table 5), show a higher jitter in message 1 as expected due to blocking induced by the lower priority messages 2 and 3 in the same VLAN. However, despite rare, a few larger blockings were detected that need another explanation since they are longer than the longest lower priority message. This issue will be investigated in future work.

The results with all VLANs are similar but there are now 0.45% (in 150000) lost packets in message 1. This shows that the VLANs are still isolated in terms of temporal behavior but, when queuing is required, the memory overflow caused by the traffic in VLANs 2 and 3 propagates to VLAN 1, causing lost packets.

Table 4 – Load for experiments 3.3B

VLAN	Mesg	Add	Prio	Data size	Period
1	1*	Brd	H	46B	1ms
1	2	Uni	L	50-200B**	0.4-0.6ms**
1	3	Uni	L	46B	1ms
2	4	Uni	-	46-100B	***
2	5	Uni	-	46-100B	***
3	6	Uni	-	46-100B	***
3	7	Uni	-	46-100B	***

* monitored stream ** uniformly distributed *** continuous tx

Table 5 – Results for experiment 3.3B

Measured period (µs)				Relative jitter (µs)			
min	max	avg	std	min	max	avg	Std
67	2269	1000.1	91.3	0	2202	103	110.5

3.4- Interference between priority levels

The use of priority levels is an adequate technique to differentiate the service given to different types of traffic. Within the same VLAN, it is however impossible to completely isolate message streams with different priorities because of the blocking caused by the non-preemptive nature of message transmission in the switch ports. Moreover, similarly to the previous case with VLANs, there is a potential for further interference among priority levels arising from resource sharing inside the switch such as contention in the memory access for message queuing purposes. The following experiments address this issue by proposing a load (table 6) divided between the two priority levels of the switch.

Table 6 – Load for experiments 3.4

VLAN	Mesg	Add	Prio	Data size	Period
1	1*	Brd	H	46B	1ms
1	2	Uni	L	100B	0.25ms
1	3	Uni	L	50-200B**	0.4-0.6ms**
1	4***	Uni	L	100B	0.2ms

* monitored stream ** uniformly distributed *** only in exp. B

The experiments were also carried out in two parts. Firstly, using a high but moderate offered load in the lower priority level with an overall traffic utilization of 60% of the 10Mbps bandwidth (messages 2 and 3). The results are shown in table 7 where the interference due to non-preemptive blocking is clear in the jitter figures.

Table 7 – Results for experiment 3.4A

Measured period (µs)				Relative jitter (µs)			
min	Max	avg	std	min	max	avg	std
897	1179	1000.1	86.4	0	279	121.1	70.3

In the second part, message 4 was added, raising the submitted load to 103.5% and leading to packet losses. However, since the message size of the low-priority messages was kept unchanged, which is the factor that determines the blocking incurred by the higher priority stream, we would expect the jitter figures to remain roughly unchanged. Instead, we have observed a misbehavior of the switch, queuing many consecutive instances of the high-priority message, and then transmitting them in a burst. This resulted in a high dispersion of the delay values shown in table 8 and depicted in figure 3. When receiving bursts, the measured period was around 68µs as expected. A few rare lower values (>51µs) were detected which seem to result from an interference of the monitoring tool.

Beyond this misbehavior, message losses also affected the high priority traffic, reaching 16.79% of the total number of packets submitted.

Table 8 – Results for experiment 3.4B

Measured period (µs)		Lost packets
min	max	
51	252020	16.79% (in 150000)

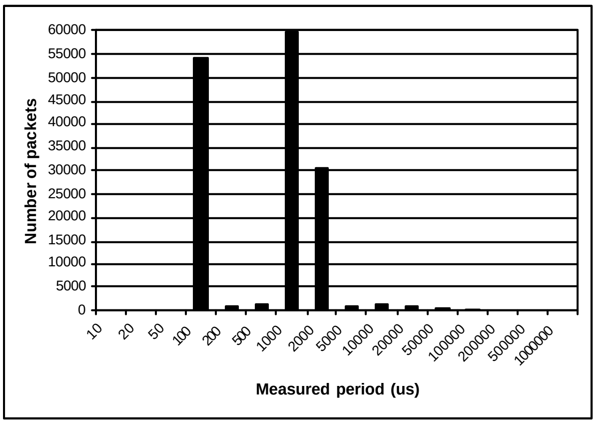

Figure 3 –Dispersion of measured periods with overload in the lower priority level (experiment 3.4B).

3.5- Summary of results

Summarizing, the experiments described above show three main results:

- The extra latency caused by using a switch with respect to a hub is relatively small for a cut-through switch (only the time to transmit 14bytes);
- Independent VLANs interfere with each other, either through the switch CPU load (small interference) as well as through the queues memory. This second factor is more important and occurs when the traffic in one VLAN causes a memory overflow, leading to packet losses in any VLAN that requires queuing;
- In the same VLAN, the interference caused by lower priority traffic can have a dramatic effect on higher priority one. In particular, an excess of lower priority traffic may cause high memory usage, leading to a deterioration of the switch temporal behavior, or, if memory overflows, to packet losses in the higher priority levels.

These misbehaviors occur when the switch is heavily loaded, either permanently or temporarily during long packet bursts. This suggests that further traffic control at the switch input, i.e. the stations that generate the traffic, is required in order to prevent such misbehaviors. This traffic control can be achieved, for example, using traffic smoothing [8] or centralized traffic scheduling [6].

Flow control, as specified in IEEE 802.3x, does not solve the problem because it seems to apply to the whole switch, affecting equally the higher priority traffic.

4. Conclusions

This paper contributes to a better understanding of the real-time behavior of Ethernet switches. We believe this is of great interest nowadays, as this technology is being strongly proposed for use in distributed real-time applications, e.g. in automation environments, and in which there is a growing need to support traffic of diverse characteristics, namely real-time and non-real-time, in an integrated way. The experiments carried out in this work have shown diverse sources of interference the RT traffic in a switch is subject to. Some of these sources, such as the effect of low number of priority levels and blocking from lower priority levels can be easily modeled so that they can be taken into account in the timeliness analysis of the RT traffic. However, the interference between VLANs as well as the loss of packets in higher priority levels, caused by memory over-usage, are factors that still lack an adequate model, if one is possible.

Several aspects that will be handled in future work include the improvement of the monitoring process, the analysis of using flow control, and the impact of internal switch architectures, running the same battery of tests on several commercially available switches.

References

[1] L. Lo Bello and O. Mirabella. Design issues for Ethernet in automation. In *Proceedings of ETFA'01 - 8th IEEE International Conference on Emerging Technologies and Factory Automation*, Antibes, France, October 2001.

[2] J-D. Decotignie. A perspective on Ethernet as a fieldbus. In *Proceedings of the 4th FeT'2001 - International Conference on Fieldbus Systems and their Applications*, pages 138–143, Nancy, France, November 2001.

[3] Paolo Gai, M. Giorgio, L. Abeni, and G. Buttazzo. A new kernel approach for modular real-time systems development. *Euromicro Conference on Real-Time Systems*, Delft, The Netherlands, June 2001.

[4] J. Jasperneit and P. Neumann. Switched ethernet for factory communication. In *Proceedings of ETFA2001 - 8th IEEE International Conference on Emerging Technologies and Factory Automation*, Antibes, France, October 2001.

[5] F. Monk. The future of Ethernet in the manufacturing environment. In *IEEE International Workshop on Factory Communication Systems WFCS'00 - Wip Session*, Porto, Portugal, September 2000.

[6] Paulo Pedreiras, Luis Almeida, and Paolo Gai. The FTT-Ethernet protocol: Merging flexibility, timeliness and efficiency. *Proceedings of the 14th IEEE Euromicro Conference in Real-Time Systems*, Vienna, Austria, June 2002.

[7] C. Venkatramani and T. Chiueh. Supporting real-time traffic on Ethernet. *Proceedings of IEEE Real-Time Systems Symposium*, San Juan, Puerto Rico, December 1994.

[8] Lo Bello, L., Mirabella, O., et al.. Fuzzy Traffic Smoothing: an Approach for Real-time Communication over Ethernet Networks. *Proc. of the 4th IEEE Workshop on Factory Communication Systems*, Västeras, Sweden, August 2002.