Analysis of Protocols to Ethernet Automation Networks

Raimundo Viégas Jr[1], Ricardo A.M. Valentim[1],Daniel G. Texeira [1], Luiz Affonso Guedes [1]

[1]Laboratory of Computation and Automation Engineering, Federal University of Rio Grande do Norte, Natal -RN,Brazil
(Tel : +55-84-3215 3696; E-mail: rviegasjr,valentim,danielgt, affonso@dca.ufrn.br)

Abstract: The Ethernet technology dominates the market of computer networks. However, it was not been established as technology for industrial automation, where the requirements demand determinism and real-time performance. Many solutions have been proposed to solve the problem of non-determinism, which are based on TDMA (Time Division Multiple Access), Token Passing and Master-Slave. This article realizes measurements performance comparing implementations in the data communication with UDP and RAW Ethernet protocols, identifying the most viable alternative to support the development of real-time application to industrial automation networks.

Keywords: Real-time Ethernet, Ethernet determinism, Real-time communication

1. INTRODUCTION

The use of standard communications system is an indispensable concept in industrial automation. Currently, the tendency in the industrial automation is the Fieldbus approach to use as communication infrastructure between devices such as sensors, actuators and controllers [10]. But, it is very difficult to establish standards, because there are several models of connectivity with many kinds of protocols available in the technology market [8].

At the usual way, technologies based in Fieldbus get proprietary implementations and there are few suppliers of solutions to industrial networks that is still expensive, what makes it difficult to use in large scale. The most famous examples are: Profibus, Fieldbus Foundation, Interbus, CAN, ControlNet and DeviceNet [10].

The alternative to the traditional industrial networks is the use of the Ethernet based solutions [2]. The conception and design of the Ethernet technology for computer networks were developed at the Xerox's research laboratory in 1973 [7].

Nevertheless, the Ethernet technology was not projected to support real-time requirements of automation industrial networks and it was normalized later as IEEE 802.3 standard [5]. In this context, it is a very interesting technology, because of its simplicity, high speed (Ethernet transmission rate that ranges between 10Mbps to 10Gbps, depending of the utilized standard), low cost and high interoperability [3][4].

However, there are two inappropriate characteristics of Ethernet to support industrial control application: hostile environment and non-determinism problem. The environment related different problems to office and industry. The industrial environment submits the network to mechanics efforts, high temperatures and magnetic interferences. In contrast the hostile environment factors of the Ethernet has been solved, because there are today Ethernet components such as switches, hubs, cables and connectors projected to support the showed characteristics [1].

According to Carreiro [3], currently the main problem of the Ethernet technology is the non-determinism that is attributed to the CSMA-CD protocol (Carrier Sense Multiple Access - Collision Detection) [5]. The CSMA-CD protocol works when a station wants to transmit, it listens to the transmission medium. If the transmission medium is busy, the station waits until it goes idle; otherwise, it transmits immediately. If two or more stations simultaneously begin to transmit, the transmitted frames will collide, all the transmission stations will terminate their own transmission and send a jamming sequence to ensure that all transmitting station abort the transmission and counts in aleatory time (Backoff Mechanism). When there are collisions detection randomly initiated transmissions between packets of data destined to the same network. This non-determinism propriety of CSMA protocol becomes not possible to calculate the precise time to data packet delivery. What implicates on the in existence of constant lateness [9].

Otherwise, the switches based solution can mask this collision problem [12], because there is a contention risk in the device that usually happens because there is a big number of data packets receiving destined to the same port, causing unexpected retarding or the frames discard [11]. In Ethernet architecture, the minimum frame payload is 46 Bytes (dictated by the slot time). This implies a frame consisting of 72 Bytes (see table 1) with a 9.6 Microseconds inter-frame gap (corresponding to 12 Bytes at 10Mbps). The total used a period (measured in bits) corresponds to 84 Bytes [14].

Table 1 Number Of Bit Periods Occupied By Smallest Size Of Ethernet Frame.

Frame Part	Minimum Size Frame
Inter Frame Gap (9.6μ)	12 Bytes
MAC Preamble (+SFD)	08 Bytes
MAC Destination Address	06 Bytes
MAC Source Address	06 Bytes
MAC Type (or Length)	02 Bytes
Payload (Network PDU)	46 Bytes
Check Sequence (CRC)	04 Bytes
Total Frame Physical Size	84 Bytes

According to Kiszka [11] there has been actually an effort to support QoS (Quality of Service) [6] in switches to improve this situation, but there is still high cost to

industrial application.

In order to resolve the non-determinism problem in Ethernet networks, there are researches today working essentially over two distinct ways: The first proposes changing in the medium access control (MAC) and the other proposes bus access control implementations over the high level network protocol as UDP. Normally, TDMA (Time Division Multiple Access), Token Passing and Master-Slave are used to get bus access control [9].

According to the exposed context, the present paper examines performance comparison in a local area network with Ethernet technology, confronting implementations over the UDP protocol, that is a freely available and well-known standard with low protocol overhead [13], and RAW Ethernet protocol works on the link layer will process the data packets to be sent on the network in the appropriate form and pass it on to the hardware.

The main objective of this paper is to determinate which protocol is most available to support deterministic solution development to the bus access control. It seeks to find the efficient solution to the non-determinism problem in industrial networks based in Ethernet technologies.

This paper is organized as follows: section 2 describes the experimental project and employed methodology to work with UDP and Raw Ethernet protocols in the test environment; section 3 shows a discussion about experimental results; and the section 4 presents the conclusion and future works.

2. EXPERIMENTAL PROJECT

The experiments reported in this paper show to measure the Ethernet technology performance using two protocols (UDP and RAW Ethernet) in similar way for low processing power machines. The performance measurements of the protocols were guided to a methodology inserted on the data collect that is examined along of this section.

2.1 Experimental Environment

In order to execute the experiments, we used two kinds of structures to setup the experimental environment: Hardware and Software. The hardware is formed by two similar microcomputers connected to an Ethernet network using a hub or switch, as shown in figure 1.

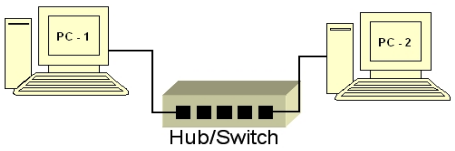

Fig. 1 Computer infrastructure used in the experiments

- Processor Intel Pentium III, 800Mhz;
- RAM: 128 MB SDRAM DIMM;
- Main Board: Soyo model 7VBA 133;
- Net Board: 10/100Mbps, Chipset:Realtek;
- Hub: 10Mbps, 3Com, Super Stack II hub 10;
- Hub: 100Mbps, 3Com, Super Stack II hub 100;
- Switch: 10/100/1000Mbps 3Com, Super Stack III;

The experimental setup consists on two similar microcomputers running software specially developed to the kinds of tests showed in this paper. The computers running a Linux standard were connected via hub or switch. The experimental environment is not connected to the Internet to avoid interference from another traffic. The used applications are based on client-server paradigm. The client software, that is able to send data by UDP and Raw Ethernet protocols was installed on the PC-1 and the server software, that is able to receive and resend UDP and RAW Ethernet payload to the application original client, was installed on PC-2. The software structure were totally developed in Linux Kernel 2.6.12 operational system, using the C program language. The developed applications to implement tests were compiled on GCC version 3.4.5.

On the next subsection the architectures from this implemented applications to the test was detailed.

2.2 Application Architecture

Four applications were developed to provide analysis of the Ethernet network performance:

- Two of them based in UDP and;
- The other two based in RAW Ethernet.

The figures 2 and 3 show the software architectures used in the test and their working mechanisms. Theses two approaches operate in different layers on the protocol stack, but their activities way are similar.

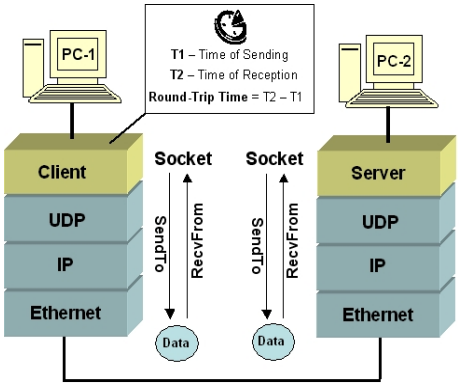

Fig. 2 Application Architecture Based on UDP

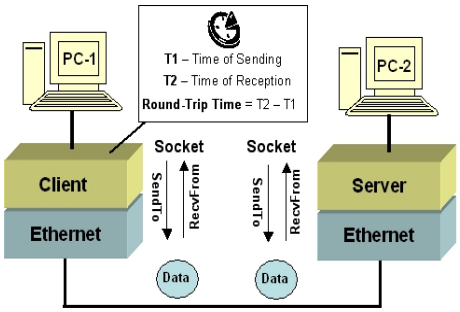

Fig. 3 Application Architecture Based on RAW Ethernet

The activity way of the applications is guided by the time measuring of the Round Trip Time (RTT). Therefore a time measurer with precision of microseconds was implemented on the client applications.

The time measurer works as long as the data sending function is called. Therefore the $T1$ time is measured (beginning of sending time) and this function is finished when the same data sent to server application is received and come back to the client application, then the $T2$ time is measured. The RTT time of data packet totalizes the difference between T2 and T1, that is, $RTT = T2 - T1$. As it can be seen in the figures 2 and 3, client applications send data to the server applications.

The server applications receive the data and immediately return the same data received to the client applications. This kind of procedure is adopted to guarantee that the round trip data is the same, in other words, the data packages come back to their respective applications and have exactly the same size. A substantial factor that distinct this application in terms of software architecture is the access points in the protocol stack. In the case of the application of the figure 2, a socket is opened on the transport layer (UDP). In the case of the RAW Ethernet application (see figure 3), a socket is opened on the link layer.

This strategy was adopted in order to measure the difference performance between UDP and RAW Ethernet. It is considering the overhead cost to applications that work in the level two (Link) and in the level four (Transport) from the OSI/ISO reference model [12].

The number of times that the data packets are sent and their respective sizes can be configured in the client application, as in the example: 1000 data packets of 512 Bytes are sent.

On the next subsection the cases of tests that compose the performance measuring strategy are described.

2.3 Experimental Setup

In the case of the software tests were developed in three distinct sceneries:

- PC's connected by 10Mbps hub
- PC's connected by 100Mbps hub
- PC's connected by 100Mbps switch

In all sceneries the data packets (payload) were created according to the sizes in bytes: 64, 128, 256, 512 e 1024. These data packets were sent from PC-1 to PC-2 and returned from PC-2 to PC-1. For every experiment it were measured the RTT times, using the test software to UDP and RAW Ethernet protocols, where we got the arithmetic media and standard deviation.

Based in Kiszka[11], the next standard measurements were determined, what is showed in the table 2, where "n"corresponds to the number of repetitions of the experiment. It was verified by the experiments that "n"values upper than 500 are enough to get a good and believable statistic. For all experiments n=1000 was used. In this case, $T1_{proc} = T2_{proc}$ was considered, because the two machines are similar. The time processing values in the protocol stack were measured using the loopback data packet sending. All the test sceneries proposed in this article provided the necessary samples to performance the analysis.

Table 2 RTT - Round Trip Time

VARIABLE	DESCRIPTION
$T1_{proc}$	Processing Time in the Protocol Stack added NIC Time of the PC-1
$T2_{proc}$	Processing Time in the Protocol Stack added NIC Time of the PC-2
T_{frame}	Frame Propagation Time on Ethernet (Round Trip Time)
RTT	$\frac{1}{n}\sum_{i=1}^{n} T1_{proc}+2T_{frame}+T2_{proc}$

3. EXPERIMENTAL RESULTS

The results showed in this section are guided by the graphics generated from the samples obtained with the experiments performed by each scenery.

3.1 Test Scenery

The tests based in the proposed sceneries were based in examples collected from the developed systems, what allowed the get of results and construction of the necessary graphics. On the figure 4 there is an example of $(RTT/2)$ for UDP protocol with several kinds of samples collected using a 100Mbps hub. These samples change according to the data packet size. For all samples the standard deviation was of approximately 5 microseconds. As seen on figure 4, there are points out of the arithmetic average, this happens when an operating system standard is not able to serve all requirements in real-time.

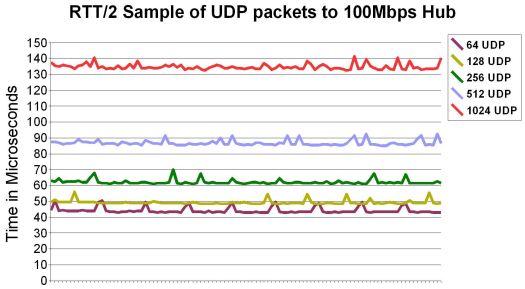

Fig. 4 Sample of data packets sent in microseconds

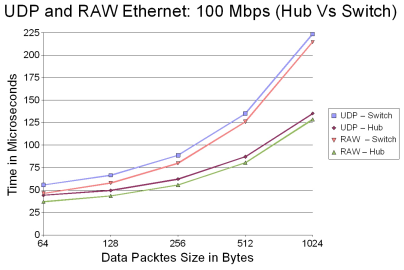

Fig. 5 UDP and RAW Ethernet (100 Mbps)

For the experiments, the processing time on the protocol stack was checked. Therefore, the loopback data packet sending was used, as seen in figure 6 and table 3. It was verified that RAW Ethernet times for all the sizes of sent packages are practically constant. In contrast, the behavior observed with UDP protocol varies according to the size of the data packet.

Table 3 Performance Measurements Of Protocol

10Mbps(HUB)	UDP(μs)	RAW(μs)
64 Bytes	136	107
128 Bytes	293	264
256 Bytes	503	474
512 Bytes	136	107
1024 Bytes	917	887
100Mbps(SWITCH)	UDP(μs)	RAW(μs)
64 Bytes	55	46
128 Bytes	66	58
256 Bytes	88	80
512 Bytes	135	126
1024 Bytes	223	214
100Mbps(HUB)	UDP(μs)	RAW(μs)
64 Bytes	44	37
128 Bytes	49	43
256 Bytes	62	55
512 Bytes	87	80
1024 Bytes	135	128
Loopback	UDP(μs)	RAW(μs)
64 Bytes	26	9
128 Bytes	27	10
256 Bytes	29	9
512 Bytes	132	11
1024 Bytes	40	12

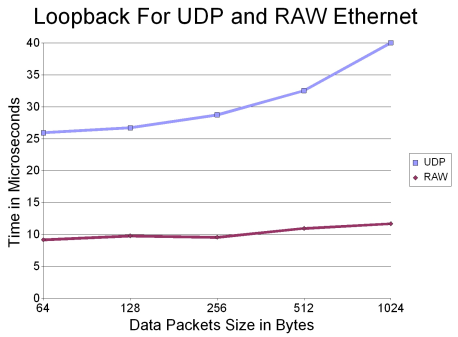

Fig. 6 Loopback data packet for UDP e RAW Ethernet

On figure 5 and table 3 the Ethernet performance for the two protocols is illustrated, using network equipments with different characteristics (bandwidth and medium access control). In this experiment, is showed that the performance of the 100Mbps hub is better than 10Mbps hub and 100Mbps switch.

A very important factor observed in this experiment is that both, network equipment with same transmission rates (100Mbps hub and switch) and the protocol performances was different. When the data packets sizes were 1024 Bytes the performance of hub was approximately 67% better than the switch.

When the data packets size was 64Bytes, the performance of the hub, compared to switch, was approximately 25% better, independently of the protocol. This factor depends on the processing time of the switch that rises according to the data packet size, what does not happen to hub. The results gotten with theses experiments show the cost with the UDP[1] protocol overhead is bigger than with the RAW Ethernet[2]. This cost reflects on the transmission of data packet and on the protocol stack processing time. Therefore, the UDP protocol rises the processing time and the communication costs too. According to results the best communication performance was of RAW Ethernet protocol using the 100Mbps hub.

3.2 Results

This Section contains the results from experiments tests, and one of the most important parts of this paper was to verify the costs during the data packets sending, considering the 100Mbps hub, because that one got the best performance. The methodology used was to divide the sending cost in:

- SUT - Stack Up Time
- SDT - Stack Down Time
- NUT - NIC Up Time
- NDT - NIC DownTime
- HTX - Header transmission Time
- DTX - Data Transmission Time.

The transmission times of the Header (HTX) and of the Data Transmission Time (DTX) were gotten as theoretical way. The others variables were gotten by the experimental values. The Stack Up Time (SUT) and Stack Down Time (SDT) along the stack were gotten by the loopback experiments, it includes the preambles of the frames Ethernet. The Network Interface Card processing times (T_{NIC}) was gotten according to the following equations.

Thus:

$$T_{frame} = HTX + DTX$$

$$T_{stack} = SUT + SDT$$

$$T_{NIC} = \frac{RTT}{2} - [T_{frame} + \frac{T_{stack}}{2}]$$

Then,

$$T_{NIC} = NUT + NDT$$

On figure 7, the communication and processing costs were illustrated, where we can see the cost with UDP protocol overhead (HTX) that is constant and 55.5% bigger than with RAW Ethernet. It was observed that the processing costs in stack and in network devices are respectively related to processor and NIC performances. This experiments show that one of the biggest communication costs is NIC (NUT and NDT). Another factor is that the minimum overhead in stack using the RAW Ethernet protocol. This aspect is true to RAW Ethernet, no matter the data packet size, but this behavior was not verified in UDP protocol. On figure 8, the performance of a 100Mbps hub is compared to a 100Mbps switch. As seen, the switch provides the biggest communication time (transmission delay), because the processing grows according to the data packet size.

[1] Total Overhead UDP is 46Bytes
[2] Total Overhead for RAW Ethernet is 18Bytes

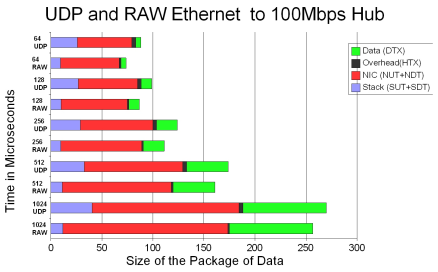

Fig. 7 Cost time Segmented Transmission

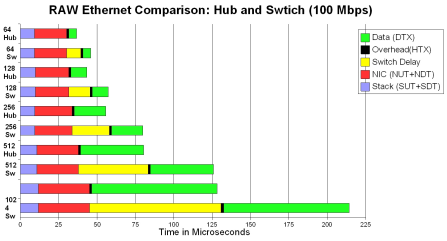

Fig. 8 Comparison between UDP and RAW Ethernet

4. CONCLUSIONS

This paper permitted to check the Ethernet technology the conduct across an experimental project of performances analyzes. The performances measures were examined across the UDP and RAW Ethernet Protocols. The tests were made in an environment where the microcomputer has a low processing power, what typically happens in embedded industrials applications. This factor brings a meaningful contribution, because the project can serve as base to the development of deterministic protocol to real-time applications. The obtained results were very important; because it could point that a big part of the communication cost is due to NIC, because of the use of a standard Linux that has a task scheduler processing joined to low hardware processing. Another substantial observation is that the RAW Ethernet protocol performance is 55.5% better, compared to UDP overhead protocol, however, it was possible to verify that the development of the applications for Raw Ethernet grows in difficulty, because its API (Application Program Interface) is from low standard, if compared to API socket UDP. An interesting results obtained to RAW Ethernet can be observed in the tests with 100Mbps hub where was obtained performances between 25% and 65% (changing according to size of packets) better than with the 100Mbps switch. The reason of this behavior can be explain by the delay time imposed by switch to the data packets. After all the obtained results, we have thought to make in futures work the same tests utilizing a real-time Linux. Consequently, a better performance is expected. The experiments realized in this paper, as the next work in perspective, must contribute to the developments of effective's techniques to the bus access control to deterministic Ethernet network. It making possible the development real-time applications.

REFERENCES

[1] Amphenol Socapex Field Buses, "Reinforced rj45 for industrial Ethernet network applications," *Available: www.amphenol-socapex.com/*, Mar, 2006.

[2] J. D. Decotignie, "A perspective on Ethernet-TCP/IP as a fieldbus," *in Proceedings of LORIA. 4th International Conference on Fieldbus Systems and their Applications*, 15-16 Nov. 2001, Nancy, France, 2002.

[3] F. Carreiro, J. Fonseca, V. Silva, and F. Vasques, "The Virtual Token Passin Ethernet: Implementation and Experimental Results,," *presented in Proceedings of the 3rd Workshop on Real-Time Networks*, Catania, Italy, 2004.

[4] O. Dolejs, P. Smolik and Z. Hanzalek . "On the Ethernet use for real-time publish-subscribe based applications". In 5th IEEE International Workshop on Factory Communication Systems, Vienna, Austria, Sep. 2004.

[5] IEEE 802.3/ISO 8802-3 - Information processing systems - Local area networks - Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications, 2nd edition, 21 September 1990.

[6] IEEE-802.1Q - IEEE Standards for Local and Metropolitan Area Networks: Draft Standard for Virtual Bridged Local Area Networks, P802.1Q, January 1998.

[7] R. Metcalfe and D. R. Boggs, "Ethernet: Distributed packet switching for local computer networks," *ACM Communications*, 395-404, 1976.

[8] P. Neumann, "Communication in Industrial Automation what is going on?," *In South African Institute for Measurement and Control (SAIMC)*, September 2004.

[9] P. Pedreiras, L. Almeida, P. Gaia e B. Giorgio, "FTT-Ethernet: A Flexible Real-Time communication Protocol That Supports Dynamic QoS Management on Ethernet-Based Systems," *IEEE Transactions on Industrial Informatics*, Vol. 1, N. 3, August 2005.

[10] J. P. Thomesse, "Fieldbus Technology in Industrial Automation," *Proceedings of the IEEE*, Vol. 93, No. 6, June, 2005.

[11]] J. Kiszka, B. Wagner, Y. Zhang and J.F. Broenink, "RTnet - A flexible Hard Real-Time Networking Framework", In 10th IEEE International Conference on Emerging Technologies and Factory Automation, 19-22 Sept., Catania, It, 8 pages, 2005

[12] K. Hansei, "Redundancy Ethernet in Industrial Automation", In Emerging Technologies and Factory Automation - ETFA, Catania, Italy, 2005.

[13] G. Prytz, G. and S. Johannessen . "Real-time Performance Measurements using UDP on Windows and Linux", In Emerging Technologies and Factory Automation - ETFA, Catania, Italy, 2005.

[14] C. E. Spurgeon, Ethernet: The Definitive Guide, 1rd. Ed., O´RELLY, 2000.