INT J COMPUT COMMUN, ISSN 1841-9836
Vol.7 (2012), No. 4 (November), pp. 744-758

Minimum Cycle Time Analysis of Ethernet-Based Real-Time Protocols

J. Robert, J.-P. Georges, E. Rondeau, T. Divoux

Jérémy Robert, Jean-Philippe Georges,
Eric Rondeau, Thierry Divoux
Centre de Recherche en Automatique de Nancy,
Université de Lorraine, CNRS UMR 7039
Campus Sciences, BP 70 239, F-54506 Vandœuvre-lès-Nancy, France
E-mail: jeremy.robert@univ-lorraine.fr, eric.rondeau@univ-lorraine.fr,
jean-philippe.georges@univ-lorraine.fr, thierry.divoux@univ-lorraine.fr

Abstract:
The Ethernet standard is a standard solution for interconnecting industrial devices despite its intrinsic drawbacks, particularly its nondeterministic medium access method. Many Ethernet-based commercial solutions available (COTS - Components Off the Shelves) on the market guarantee time performance. This means that user selection of one particular solution is a critical decision, but the choice often depends more on political strategizing with an industrial device manufacturer than on the intrinsic performance of Ethernet-based interfaces. The objective of this paper is to provide a formal behavioural analysis of each Ethernet-based solution, in order to facilitate comparison.
Keywords: Real-time systems, performance analysis, embedded systems.

1 Introduction

Fieldbuses interconnecting industrial equipment are typically designed by Programmable Logic Controller (PLC) manufacturers. This is mainly attributable to specific constraints on industrial communication, which require a high degree of expertise. Fieldbuses must be robust in the noisy environments produced by the plant (physical layer), deterministic in guaranteeing data refresh during controller cycle periods (link layer), and capable of exchanging information between all types of industrial devices (application layer). Manufacturers provide different solutions to satisfy the first two constraints, without necessarily considering the final constraint to be relevant in business terms. This has resulted in the specification of a large number of fieldbus standards as described in [1].

A new trend endorsed particularly by the IAONA (Industrial Automation Open Networking Alliance) consortium was to promote the Ethernet network as a standard for industrial communications. The expected benefits are less costly network installations, because equipment is available off-the-shelf, and the avoidance of interoperability problems, because Ethernet technology is broadly used. Other advantages are that Ethernet is a well-known protocol, which is widely implemented, and its performance improves continuously with technological evolution (especially bandwidth).

However, access to the Ethernet medium relies on the nondeterministic CSMA/CD algorithm, which applies a stochastic method to resolving collisions and cannot guarantee that message transmissions will be received in bounded time. Consequently, the native Ethernet protocol cannot be directly implemented in a plant with severe time constraints, and many Ethernet-based solutions have been proposed to overcome this issue, as mentioned in [2–4]. However, if these Ethernet-based solutions are adapted to industry requirements, they lead to two types of problem. The first is that different solutions are not interoperable because different fieldbuses are

developed by each manufacturer. The second problem is that time performances are insufficiently evaluated or compared.

According to [5], different Ethernet products can be summarily classified into three main categories: the native Ethernet standard (Ethernet/IP, Modbus/TCP), Ethernet solutions using the priorities defined in IEEE802.1D/Q, and Ethernet-based solutions that incorporate new scheduling features in ASIC/FPGA (EtherCAT, Profinet IRT).

The last approach enables the elimination of all collisions and simplifies transmission time estimation, as described in [5, 6]. The authors compared EtherCAT with Profinet IRT in a simplified context using analytic models. This analysis was improved and refined in section 2 and extended to two other well-known industrial Ethernet products: the Modbus/TCP solution and Ethernet/IP. In section 3, we present several scenarios that use the analytic models to facilitate assessment of different Ethernet products.

2 Estimation of minimum cycle time

2.1 Introduction

The general objective of this study was to compare the time performance of the major industrial Ethernet products available on the market. This comparison could be achieved only in a common application context. Thus, the specification of the communication scenario was based on one controller (for example, a PLC), interconnecting sensors, and actuators in an Ethernet network. The controller was treated as the communication master that initiated all dialogues with slave nodes (sensors and actuators). The controller was characterized by its controller cycle time period, which was divided into three steps, as shown in Figure 1: - the sensor data refresh time, in the controller memory, - the processing time, and - the actuator update time.

Steps (1) and (3) represent communication periods, which should be less than the difference between the controller cycle time period and the processing time period. Thus, the time performance of each Ethernet product was compared according to a constraint named *minimum cycle time*, which was defined as:

The minimum cycle time was the communication time required by the controller to both collect and update the data memories of all sensors and actuators.

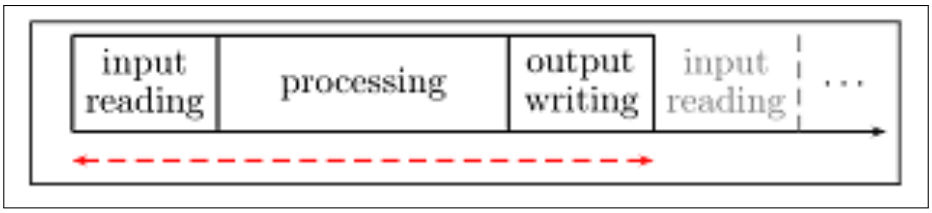

Figure 1: PLC cycle time

In the following section, the analytic models of minimum cycle time are elaborated for EtherCAT, Profinet IRT, Modbus/TCP, and Ethernet/IP. These models all used the following parameters: the transmission delay, the network device latency, the propagation delay, the link capacity, the payload, and the number of slaves. The notations of these are given in table 1. It was assumed that there were no transmission errors and that the network was dedicated to the PLC application and was not shared with other applications.

Table 1: Notations

Terms	Notation	Units
Minimum cycle time	Γ	s
Delivery time (from frame i)	d_i	s
Transmission delay	τ	s
Network device latency	ℓ	s
Propagation delay	δ	s
Link capacity	C	$bits/s$
Payload	x	$bytes$
Number of network devices (slaves)	n	–

2.2 EtherCAT

The EtherCAT network was developed by the Beckhoff company (type 12 in standard IEC 61158, [7, 8]). In theory, EtherCAT cards are standard Ethernet interfaces. In practice, specific hardware (FPGA, Field-Programmable Gate Array, or ASIC, Application-Specific Integrated Circuit) is used to mitigate the frame forwarding delay. The EtherCAT network adds a master/slave protocol over the Ethernet. A frame is sent by the master and slaves can read and write data *on the fly*. The duration of reading or writing operations corresponds only to the network device latency ℓ, which is independent of frame size and the same for all slaves. A logical ring is defined between the slaves such that when a frame reaches the last slave in the ring it is returned to the master via all the slaves. The space–time diagram shown in Figure 2 illustrates the behaviour of EtherCAT communications.

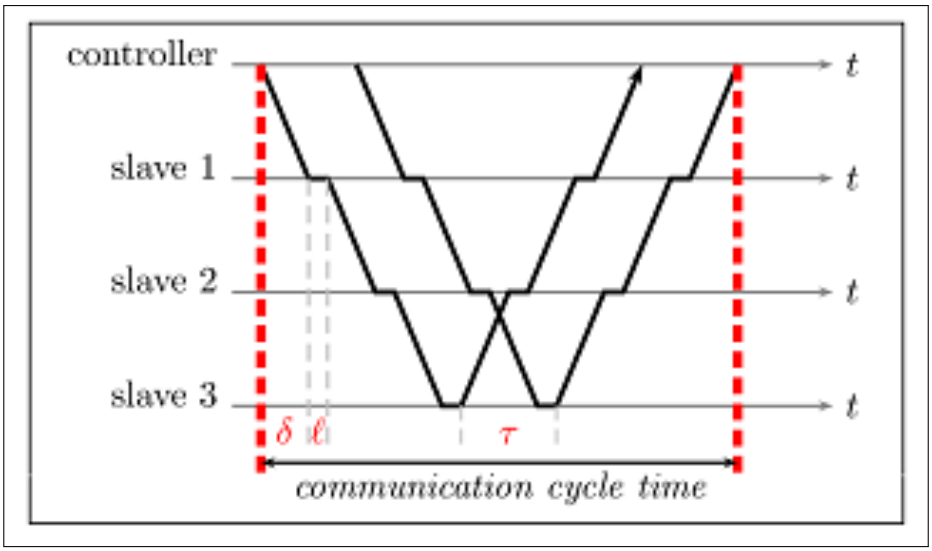

Figure 2: EtherCAT space-time diagram

The EtherCAT protocol can support both line and ring topologies. Because the line topology is used mainly in the industrial framework, this topology was investigated in our study (Figure 3).

The EtherCAT datagram is directly encapsulated inside the basic Ethernet frame as shown in Figure 4. An EtherCAT frame is composed of a header specifying the length of the frame and a list of datagrams. The number of datagrams depends on the number of slaves. A datagram is

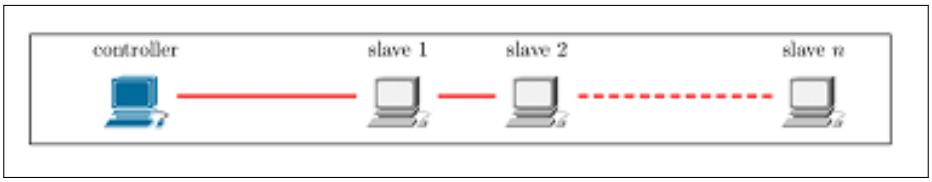

Figure 3: EtherCAT line topology

defined for each slave, and it contains the command type and associated data.

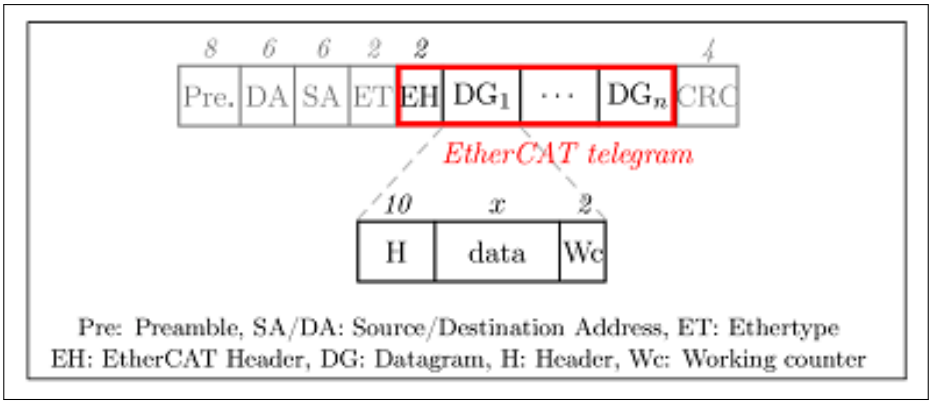

Figure 4: EtherCAT frame *(field lengths are given in bytes)*

In this study, our analysis of EtherCAT performance considered the following hypothetical test scenario: - the topology was a line, - the initialization step was ignored and only cyclic communication was studied, - the master sent only one frame per cycle, and - the payload x was the same for each slave.

The minimum cycle time shown in Figure 2 was determined for this scenario.

The link transmission delay is the ratio between frame size and link capacity C. The total frame size can be divided into two parts: - a constant value that equals the sum of the Ethernet protocol (26 *bytes*), the interframe gap (corresponding to the time of 12 *bytes*), and the EtherCAT header (2 *bytes*) and - a variable value that depends on the slave number n, the amount of data to transport x, and the header (12 *bytes*).

The link transmission delay is:

$$\tau = \frac{8\,(40 + \max\,(44, n\,(12 + x)))}{C}. \tag{1}$$

The term 44 in the equation (1) was added to ensure the minimum data size defined by the Ethernet protocol. If the EtherCAT telegram length was less than 46 *bytes*, an equivalent amount of padding was inserted in the Ethernet frame. The EtherCAT telegram already included a 2 *bytes* header, which meant that there was no padding requirement when the length of the datagram sequence was larger than 44 *bytes*.

As shown in Figure 2, the cycle time was estimated using the expression:

$$\begin{aligned} \Gamma &= (2n-1)\,\ell + 2n\delta + \tau \\ &= (2n-1)\,\ell + 2n\delta + \frac{8\,(40 + \max\,(44, n\,(12 + x)))}{C}. \end{aligned} \tag{2}$$

It should be noted that (2) considers only one frame. Because the Ethernet payload size depends directly on the number of slaves and the Ethernet frame size cannot exceed 1 526 *bytes*

(and therefore the data field 1 500 *bytes*), (2) is only valid if the number of devices interconnected to the network is less than:

$$n \leq n_{max} = \left\lfloor \frac{1500 - EH}{12 + x} \right\rfloor,$$

where EH is the EtherCAT header size (2 *bytes*) and n_{max} is the maximum number of datagrams, of length x, which can be included in a single frame (in the following, we have assumed that $x \leq 1486$ *bytes*).

In general, the number of slave devices on the network can be greater than the frame size capacity. This means that the controller has to send more than one frame in a cycle time. In fact, the number of Ethernet frames required to support n devices with a constant payload x is given by:

$$k = \left\lceil \frac{n}{n_{max}} \right\rceil.$$

Consequently, (2) now integrates a different transmission time for each frame and finally gives:

$$\Gamma = (2n - 1)\,\ell + 2n\delta + \frac{8}{C}\Big(40k + (k-1)\,n_{max}\,(12 + x)\Big) + \frac{8}{C}\max\Big(44, (n - (k-1)\,n_{max})\,(12 + x)\Big) \tag{3}$$

The final term of expression (3) was used to differentiate cases where the last frame generated padding.

Similar results are given in [5] but, in contrast to this earlier work, expression (3):

- takes into account the *on the fly* minimum cycle time mechanism proposed by EtherCAT; the main advantage of this is that a device can begin frame forwarding before complete reception of a frame (in contrast to, e.g., store-and-forward mode), which significantly reduces the forwarding time as shown in Figure 2,
- considers the use of padding, as defined by Ethernet,
- integrates the time required to forward the information sent from devices to the controller, as shown in Figure 2, and
- considers cases where the number of slaves and their payload requires the utilization of several frames.

The accuracy of the EtherCAT synchronization mechanism was reported in [9], which shows that this issue need not be considered because it was estimated as equal to a few nanoseconds.

2.3 Profinet IRT

Introduction

The Profinet protocol was developed by the Siemens company (type 10 in standard IEC 61158, [7, 8]). Profinet IRT manages real-time communications. However, standard Ethernet cards cannot be used because Profinet IRT requires the operation of specific hardware on slaves (ASIC type – 2 or 4 ports inbuilt switch). Profinet IRT is based on the time-slice mechanism, which specifies two modes, the asynchronous mode and the isochronous mode.

The asynchronous and isochronous modes are used for unconstrained traffic and real-time traffic, respectively. Our study dealt only with real-time traffic. Thus, only the isochronous mode was analysed. For more information on these two modes, the reader can refer to [10]. In

the isochronous mode, the master sends one data frame to each device and each device replies to the master. Profinet can support line, star, and ring topologies. Only the line topology (see Figure 5), with full-duplex links, was analysed in this study (as for EtherCAT).

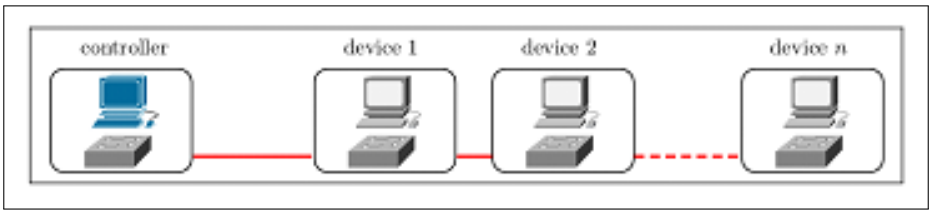

Figure 5: Profinet line topology

In this study, the Profinet IRT used the *slipstreaming effect*, where the controller began by sending frames to the most remote slave in the line topology, and then to the second remote slave slave and so on, until it reached the nearest slave. This mechanism enabled a reduction in the cycle time by minimizing the transmission time.

The *slipstreaming effect* was also applied to exchanges from slaves to the controller. The links were set up in full-duplex mode. The global communication scheme is given in Figure 6.

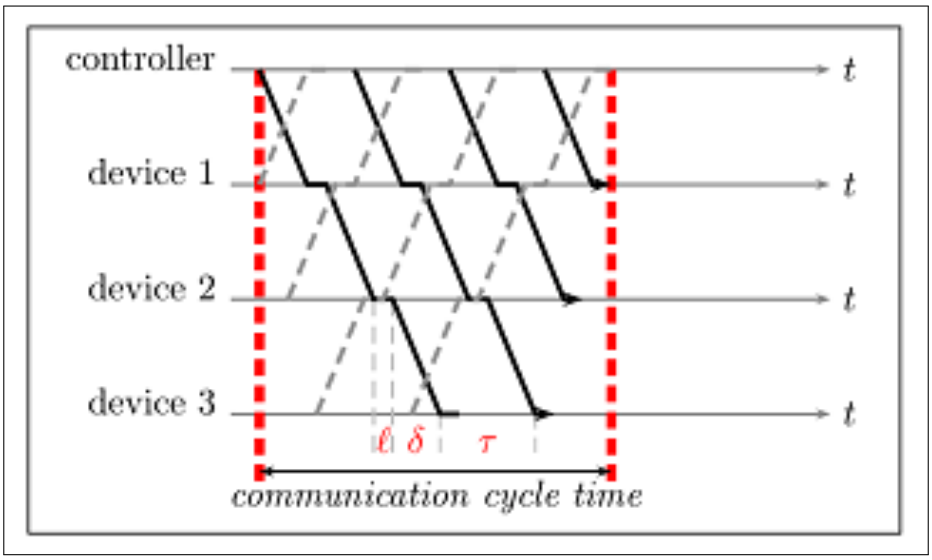

Figure 6: Profinet IRT space-time diagram with *slipstreaming effect*

The space–time diagram shown in Figure 6 illustrates the optimum use of the Profinet IRT protocol. Devices were assumed to be synchronized using a clock synchronization protocol, such as the IEEE 1588 standard. The IEEE 1588 standard generates synchronization frames (PTP) between devices, but these frames were not relevant to this study.

Minimum cycle time estimation

The slave devices periodically sent their messages to the controller at the same times as messages were sent by the controller to slaves.

Only the positive characteristics of the *slipstream effect* were considered. This required that $\tau \geq \delta + \ell$, as noted by [5].

In such cases, the minimum cycle time was given by [5,6] as the sum of:

- the latency ℓ that crosses all devices plus the propagation delay δ for each link and

- and the link transmission time τ for each frame sent by the controller.

Hence, the minimum cycle time is written as:

$$\Gamma = \delta + \ell + n\tau. \tag{4}$$

Equation (4) was developed by analysing the transmission delay τ. The Profinet datagram is encapsulated in the Ethernet frame as shown in Figure 7.

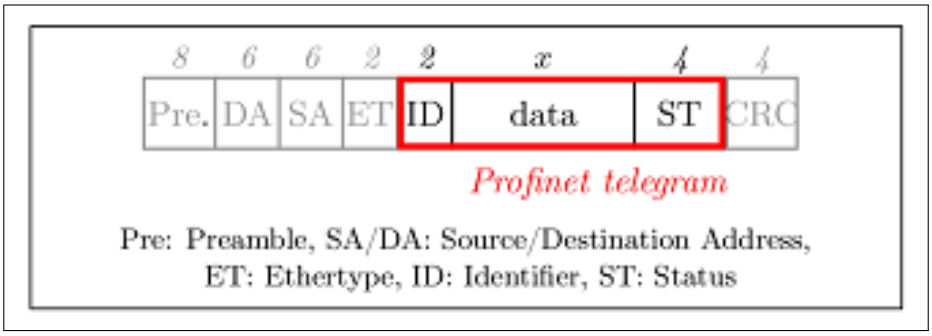

Figure 7: Profinet IRT frame *(field lengths are given in bytes)*

Three fields are added to the Ethernet frame: a data identifier (2 *bytes*), the data value (with the assumption that $x \leq 1494$ *bytes*), and an information status (4 *bytes*). When considering the constraint of the minimal Ethernet frame size, the transmission delay of a Profinet frame was given by:

$$\tau = 8\frac{38 + \max(46, 6 + x)}{C}. \tag{5}$$

The term 38 in the equation (5) corresponds to the size of the Ethernet layer (26 *bytes*) plus the interframe gap (12 *bytes*). As with EtherCAT, the term 46 was added in order to ensure the minimum data size as defined by the Ethernet protocol. If the EtherCAT telegram length was less than 46 *bytes*, an equivalent amount of padding was inserted in the Ethernet frame.

The final form of equation (4) was expressed as:

$$\Gamma = \delta + \ell + n\frac{8}{C}\Big(38 + \max(46, 6 + x)\Big)$$

Comments

The EtherCAT minimum cycle (3) was less than the Profinet IRT one (4), because the Profinet IRT transmission delay was multiplied by the number of devices. The Profinet IRT analysis assumed that all devices had the same clock reference. Figure 6 shows that all devices were synchronized because of the IEEE 1588 protocol. They shared the same clock and periodically sent their messages at the same time. Hence, it may be expected that clock synchronization errors will increase the minimum cycle time. This paper aims only at comparing optimal performances of COTS Ethernet-based protocols, i.e. Profinet IRT nodes sharing the same clock reference. Next studies will hence aim at reporting this synchronization issue.

2.4 Modbus/TCP

Modbus is a serial communication protocol developed by Modicon in 1979. Modbus/TCP is a variant of the Modbus protocol (type 15 in standard IEC 61158, [7,8]), which uses the Ethernet physical and link layers [12]. Modbus/TCP encapsulates a Modbus frame into a TCP frame as

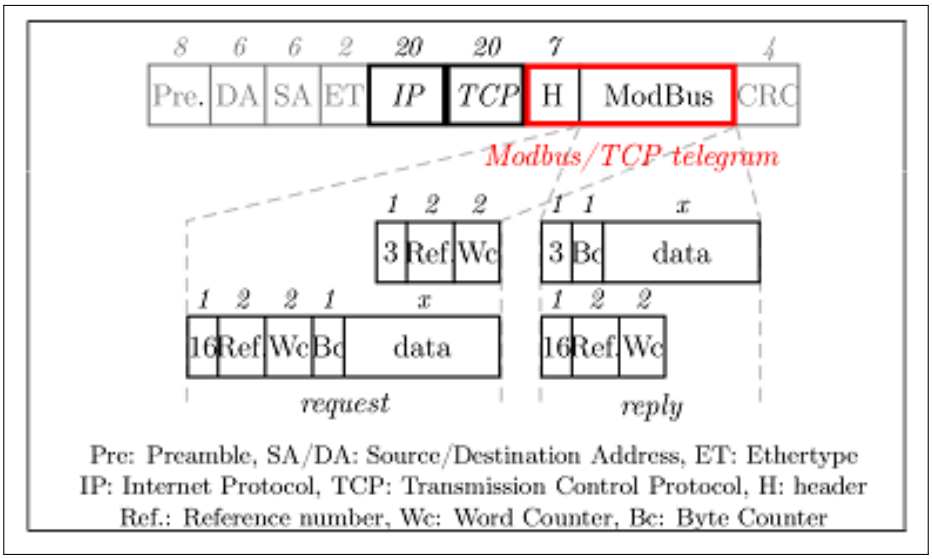

Figure 8: General Modbus/TCP frame and function specific Modbus application frame *(field lengths are given in bytes)*

shown in Figure 8. The Modbus datagram is composed of the ModBus Application Protocol Header (MBAP), the function (read/write), and the data value.

Modbus/TCP is a pragmatic approach that works on several types of configurations that impact on its performance. Parameters, including topology, exchange management in the application layer, and the processing capacity of devices, change the Modbus/TCP time behaviour. Modbus/TCP is based on connection-oriented transactions and it can use different exchange models, such as master/slaves, producer/consumer, or client/server. The objective of this study was to compare several Ethernet products, so it was necessary to use similar contexts for all Ethernet products. Because EtherCat and Profinet IRT were previously analysed using a master/slaves model, this model was also used for Modbus/TCP evaluation. The master/slaves model also simplified the analysis because all the devices were synchronized by network events (none clock synchronization protocol is required).

The communication scheme defined in this study followed these steps. The master sent a frame to one slave and when the slave received this frame, it sent a reply to the master. When the master received the reply, it repeated the same procedure with another slave. All the slaves were processed by the master using a round-robin method. This communication scheme was implemented in the application layer of the OSI model. The request/reply protocol is shown in Figure 9.

Modbus/TCP not only supports application data but also TCP PDU, and it comprises opening and closing TCP connections when acknowledging segment reception. The acknowledgment can be achieved either immediately a segment is received, after the reception of several segments, or inside the next data transmission (piggybacking). In practice, TCP behaviour changes according to the operating system, the TCP configuration, and whether or not the Nagle algorithm is used. In this study, we assumed that the timeout to send the acknowledgement was 0.5 s, meaning that only piggybacking was analysed, as shown in Figure 9. The transient states of the TCP opening and closing steps were not considered in the modelling.

Modbus/TCP supports line, star, and ring topologies. Complex architectures based on switches can be used, especially when the network is shared by several applications, although switches induce additional costs. However, because only the master/slaves communication

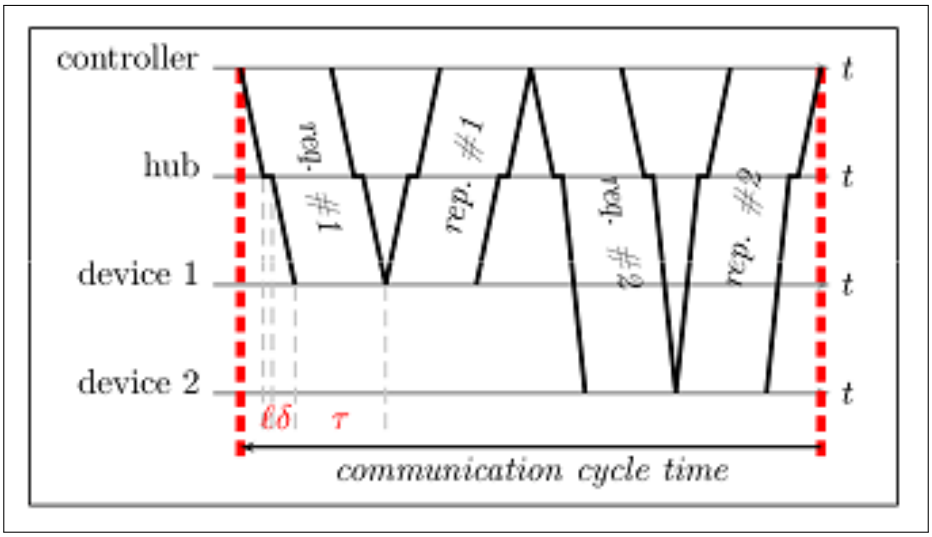

Figure 9: Modbus/TCP space-time diagram

scheme was considered in this study, a bus infrastructure was used.

It was assumed that each device was interconnected through a common hub, and the propagation delay was equal to δ. Figure 9 shows that the cycle time was equal to the number of devices n multiplied by the time required to poll one device.

Given previous assumptions, this can be written as:

$$\Gamma = n\left(\tau_{req} + \tau_{rep} + 2\left(2\delta + \ell\right)\right). \tag{6}$$

In the worst case, where a TCP acknowledgment is sent for each segment, the minimal cycle time would be equal to:$Gamma = n\left(\tau_{req} + \tau_{rep} + 3 \times \left(2\delta + \ell\right) + 2\tau_{ack}\right)$ where $\tau_{ack} = 672/C$ and 672 is the minimal size in bits for an Ethernet frame (interframe gap included). When the architecture was composed of several hubs, the propagation delay increased and (6) was slightly different.

The frame transmission time was composed of:

- a constant part related to the sum of the Ethernet protocol (38 *bytes* with the interframe gap), the IP header (20 *bytes* without options), the TCP header (20 *bytes* without options), and the Modbus/TCP header (7 *bytes* for the ModBus Application Protocol Header),
- a variable part related to the type of Modbus message, with the size changing according to the type of data (function code) and the transaction model state (request or reply).

  Consequently, the analysis of Modbus/TCP was only valid for an application framework. In this study, we considered only write requests.
- a variable part proportional to the payload (because the byte count field was stored as a single byte, this indicates that the date field length was limited to $x \leq 255$ *bytes*).

The sum of IP, TCP, and Modbus header sizes was larger than the minimal Ethernet data length. Thus, no added padding was required. The delay is directly given by:

$$\tau_{req_{10h}} + \tau_{rep_{10h}} = 8\frac{91 + x}{C} + 8\frac{90}{C}$$

such that (6) corresponds to:

$$\Gamma_{10h} = n\left(8\frac{181 + x}{C} + 2\left(2\delta + \ell\right)\right). \tag{7}$$

2.5 EtherNet/IP

Introduction

EtherNet/IP (IP, Industrial Protocol) is a network developed by Rockwell Automation in 2001 and supported by ODVA (Open DeviceNet Vendor Association) [13, 14]. EtherNet/IP (type 2 in standard IEC 61158, [7, 8]) uses the Common Industrial Protocol (CIP) for off-the-shelf Ethernet products and TCP-UDP/IP stack. Ethernet/IP is a connection-based network. A CIP connection defines the type of packet sent to the network. There are two types of connections: the Explicit Messaging connection and the I/O (or Implicit) connection. Explicit Messaging provides generic and multi-purpose communication paths between two nodes, whereas I/O messaging is specific to application I/O data and provides serial purpose communication paths. When the application is time-constrained, I/O Messaging is the preferred mode because it employs UDP rather than TCP sockets. CIP uses the producer/consumer model and requires broadcast exchanges encapsulated in UDP. This study evaluated only I/O connections.

Because EtherNet/IP relies on COTS, no particular topology was specified. Bus- or switch-based architectures are both possible. Switches are interesting because they break the collision domain, allowing the support of VLAN and the classification of service mechanisms [15]. Switched architectures are recommended for exchange management with time-critical (implicit) messaging. However, several switched Ethernet architectures are possible. In this study, a linear switched topology was selected, as shown in Figure 10. This can be viewed as an extension of the experimental set-up considered in [16], where the number of switches varied according to the number of ports per switch.

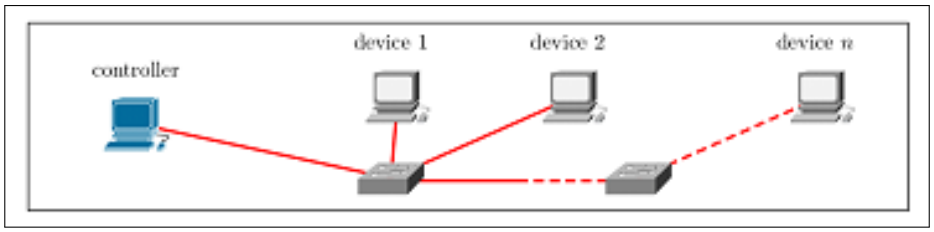

Figure 10: EtherNet/IP switched linear topology

In contrast to Modbus/TCP, utilization of a *slipstreaming effect* on Ethernet/IP does not facilitate the use of a bus topology. Indeed, it is possible that two messages can be on the network at the same time (as shown in Figure 11).

In order to compare the other protocols with Ethernet/IP, a similar exchange scenario was proposed. The controller sent a frame to each slave device and each slave device produced data that were sent to the controller. Figure 11 shows this behaviour.

Initially, EtherNet/IP did not support medium access synchronization points, as found in the master/slaves technique used in Modbus/TCP. This meant that any device could access the network at any time. However, the time synchronization mechanism can be used because of the support of EtherNet/IP by the IEEE 1588 [17] protocol (implemented by the CIPSync profile). Consequently, devices can send messages using the same clock reference. We used the *slipstreaming effect* for the controller and a common departure time for the devices, as shown in Figure 11. Obviously, this profile corresponded to an ideal case related to the minimal cycle time for this architecture, because lock synchronization errors would lead to another profile inducing a longer cycle time and the use of the *slipstreaming effect* requires (see Profinet IRT) that $\tau \geq \delta + \ell$.

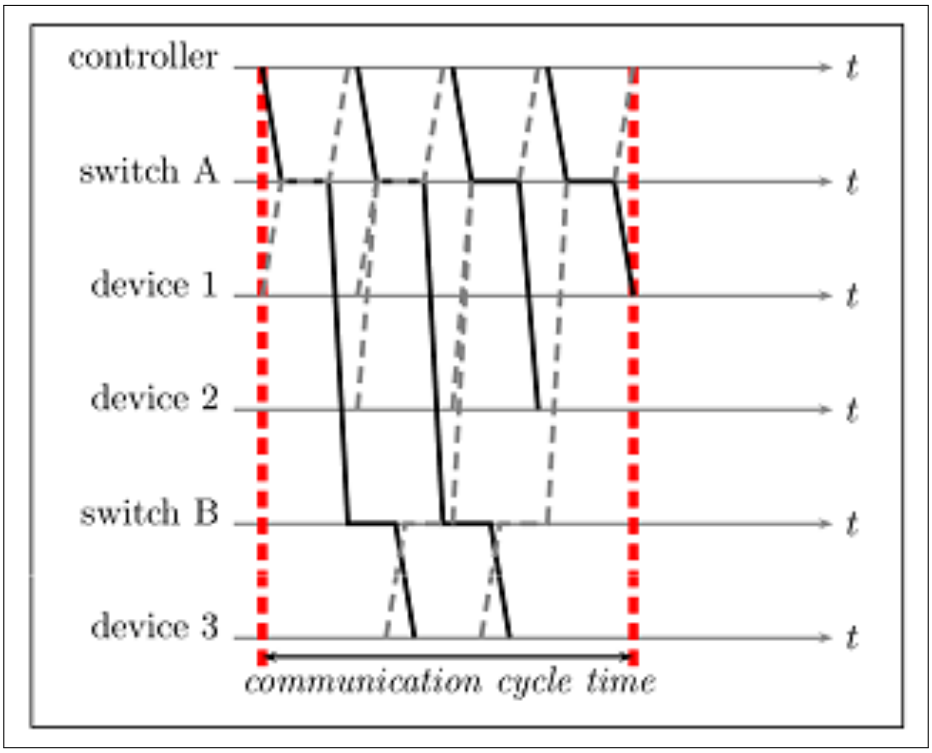

Figure 11: EtherNet/IP space-time diagram for 4-ports switches

Minimum cycle time

Considering the profile given in Figure 11, the minimum cycle time corresponded to the sum of: - the latency ℓ for crossing only one switch plus twice the propagation delay δ, between a controller/device and a switch and - the link transmission time τ for each frame sent by the controller.

Thus, the minimum cycle time can be written as:

$$\Gamma = 2\delta + \ell + n\tau. \tag{8}$$

The term given in (8) was valid only if $\tau \geq \delta + \ell$, meaning that switches did not work in the store-and-forward mode. The link transmission time was computed according to the encapsulation format of I/O messages.

Figure 12 shows that 18 *bytes* were added by the CIP protocol, 8 *bytes* by UDP, 20 *bytes* by IP, and finally 38 *bytes* by Ethernet. In this case, there was no padding. Thus, under the assumption that $x \leq 1454$ *bytes*, the link transmission time was given by:

$$\tau = 8\frac{38 + 20 + 8 + 18 + x}{C}.$$

(8) can be modified as:

$$\Gamma = 2\delta + \ell + 8n\frac{84 + x}{C}. \tag{9}$$

The profile given in Figure 11 can be optimized if several "items" (data) are encapsulated inside one frame, as specified in the CIP protocol. It induces a mitigation of the minimum cycle time.

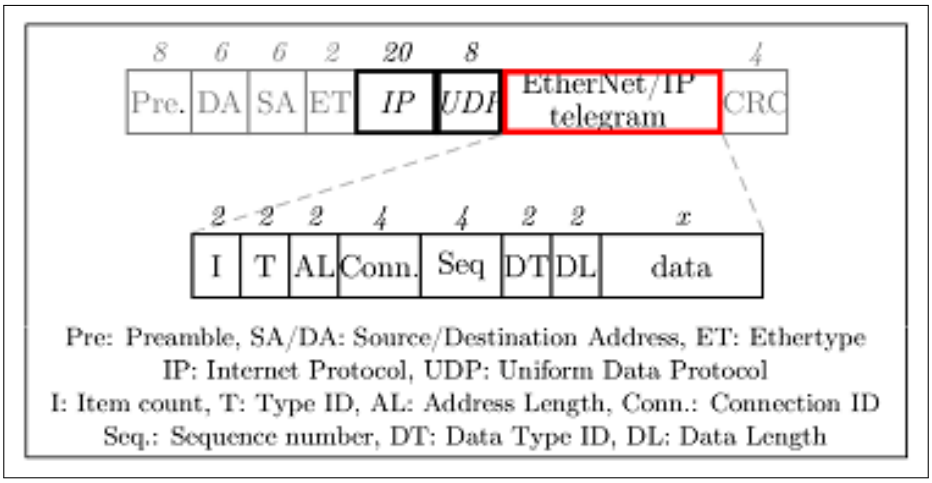

Figure 12: EtherNet/IP I/O messaging frame *(field lengths are given in bytes)*

Since we employed the same approach that for Profinet IRT (in particular the *slipstreaming effect*), next studies will hence aim at reporting the synchronization effect on the minimum cycle time.

In the next section, the results obtained for EtherCAT, Profinet IRT, Modbus/TCP, and EtherNet/IP are compared in different application contexts.

3 Comparisons

The objective was to analyse and compare the behaviour of different Ethernet-based solutions. The topologies used in each assessment were linear. Two bandwidths were studied: 100 Mb/s and 1 Gb/s. We analysed two payload sizes introduced by [6] as being representative of an industrial context: 16 *bytes* and 100 *bytes*. The network device latencies ℓ were defined in [5, 6] and those used in our study are described in Table 2. It was assumed the performances of switches used in EtherNet/IP were the same as those used by Profinet IRT. Clearly, alternative assumed values would yield different results. Thus, the results for the study using EtherNet/IP are only valid when $\tau \geq \delta + \ell$. The link propagation delay was 50 ns and corresponded to a distance of 10 m between two devices. The comparison is related to the minimum cycle time without synchronization errors.

Table 2: Latencies

Protocol	*FastEthernet* (100 *Mb/s*)	*GigaEthernet* (1 *Gb/s*)
EtherCat	1.35 μs	0.85 μs
Profinet IRT	3 μs	0.6 μs
Modbus/TCP	1 μs (*hub*)	
EtherNet/IP	3 μs	0.6 μs

Figure 13 shows the minimum cycle times (in ms), according to the number of slave devices with a payload equal to 100 *bytes* The first observation was that Modbus/TCP provided the worst results, whatever the payload. This was because of its medium access mechanism, which is based on polling at the application level.

In the case of a small payload, EtherCat provided the best results in the FastEthernet mode. The impact of bandwidth on Profinet IRT and Ethernet/IP was very significant, because it

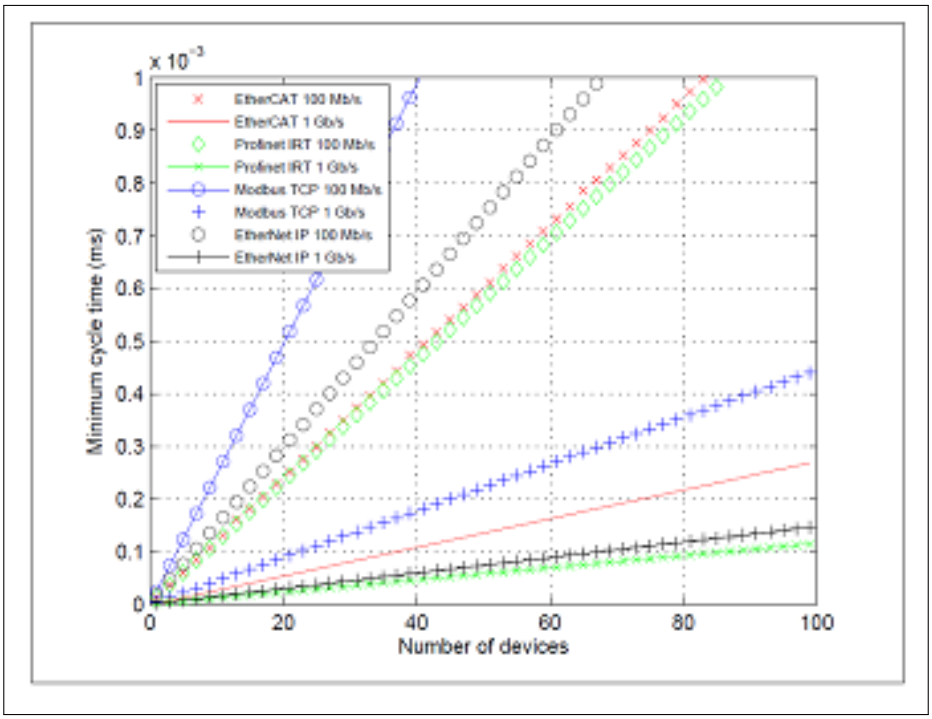

Figure 13: Minimum cycle time as a function of the number of devices with a constant payload of 100 bytes per device and two bit rates of 100 Mb/s and 1 Gb/s

enabled a reduction in the minimum cycle time. This benefit was lower with EtherCat. The explanation is simple: EtherCat sends only one frame for communicating with all its slaves and so EtherCat provides only one link per transmission. In contrast, Profinet IRT and Ethernet/IP sent n frames to dialogue with n slaves, which provides n times the link transmission. The bandwidth was the most crucial parameter in reducing the time cycle. Moreover, $2n$ latencies must be considered with EtherCat because Profinet and EtherNet/IP cycle times were composed of only one switch latency ℓ, because of the *slipstreaming effect*. Profinet IRT provided the best results at 1 Gb/s.

Figure 13 shows that all the Ethernet solutions, with the exception of FastEthernet Modbus/TCP, could interconnect with more than 60 slaves in cycle times of less than 1 ms. This cycle time constraint, and the number of slaves and the frame size, enables adequate coverage for most industrial applications. Performances were similar when the payload was increased. The only difference was that the EtherCat minimal cycle time was larger than Profinet when the number of devices was increased. This was because the EtherCat telegram had to be fragmented, which decreased protocol performance. An increase in frame size means that the choice of the FastEthernet solutions must be carefully considered because the cycle time grows quickly. The GigaEthernet solution eliminated this problem.

4 Conclusion

This study analysed the time performance of Industrial Ethernet protocols. The general conclusion was that all Ethernet protocols perform suitably with a bandwidth of 1 Gb/s for interconnected real-time systems. At 100 Mb/s, special attention is required by engineers in the selection of Ethernet-based protocols. Other considerations have to be taken into account in the selection of Ethernet-based products, including persistence in the market and interoperability

with other industrial equipment and network. Regarding interoperability concerns, Ethernet/IP is a standardized solution and provides acceptable performance for controlling industrial systems. Ethernet/IP can also implement priority mechanisms (IEEE 802.1p), which is important when a network is shared with other applications because this facility enables the differentiation of network services offered by real-time traffic and unconstrained traffic.

Moreover, multicast communications may be interesting in industrial applications. It is hence important to know what is the capacity of the analysed solutions to enhance nodes operation using multicasting techniques. As Ethernet/IP relies on COTS, multicast addresses and VLAN techniques may be used to create multicast communications. Because of its transport protocol (TCP), Modbus/TCP does not seem to be suitable to multicast communications (due to multiple acknowledgement issue). Profinet IRT considers only multicast communications in asynchronous mode. Finally, EtherCAT supports VLAN switches in its topology. As a conclusion, all solutions excepted Modbus/TCP make it possible to support multicast communications.

Future works are related to the clock synchronization issue, especially for solutions based on the *slipstreaming effect*. The objective is to deal with non optimal situations for Profinet IRT and Ethernet/IP where synchronization errors appear.

Bibliography

[1] M. Felser, T. Sauter, The fieldbus war: History or short break between battles?, *4th IEEE International Workshop on Factory Communication Systems*, pp.73-80, 2002.

[2] M. Alves, E. Tovar, and F. Vasques, Ethernet goes real-time: a survey on research and technological developments, *Technical Report HURRAY-TR-2K01, IPP-HURRAY, Polytechnic Institute of Porto (ISEP-IPP)*, 2000.

[3] J.-D. Decotignie, A perspective on ethernet-tcp/ip as a fieldbus, *4th IFAC International Conference on Fieldbus Systems and their Applications*, pp.138-143, 2001.

[4] J.-D. Decotignie, Ethernet-based real-time and industrial communications, *Proceedings of the IEEE*, 93(6):1102-1117, 2005.

[5] J. Jasperneite, M. Schumacher, K. Weber, Limits of increasing the performance of industrial ethernet protocols, *12th IEEE Conference on Emerging Technologies and Factory Automation*, pp.17-24, 2007.

[6] G. Prytz, A performance analysis of ethercat and profinet irt, *13th IEEE Conference on Emerging Technologies and Factory Automation*, pp.408-415, 2008.

[7] IEC, Digital data communications for measurement and control ? fieldbus for use in industrial control systems: Part 3: Data link service definition, *IEC Standard 61158, Part 3*, 2006.

[8] IEC, Digital data communications for measurement and control ? fieldbus for use in industrial control systems: Part 4: Data link protocol specification, *IEC Standard 61158, Part 4*, 2006.

[9] G. Cena, I C. Berlotti, S. Scanzio, On the accuracy of the distributed clock mechanism in ethercat, *4th IEEE International Workshop on Factory Communication Systems*, pp.43-52, 2010.

[10] J. Jasperneite , E. Elsayed, Investigations on a distributed time-triggered ethernet realtime protocol used by profinet, *3rd International Workshop on Real-Time Networks*, 2004.

[11] The Modbus Organization. Modbus.

[12] Modbus-IDA, Modbus application protocol specification v1.1b, 2006.

[13] P. Brooks, Ethernet/IP-industrial protocol, *8th IEEE International Conference on Emerging Technologies and Factory Automation*, Vol.2, pp.505-514, 2001.

[14] V. Schiffer, The common industrial protocol (CIP) and the family of CIP networks, *Technical Report PUB00123R0, Open DeviceNet Vendor Association, Inc. (ODVA)*, 2006.

[15] A. Modlovansky, Utilization of modern switching technology in ethernet/IP networks, *1st International Workshop on Real-Time LANs in the Internet Age*, pp.35-37, 2002.

[16] E. Alessandria, L. Seno, S. Vitturi, Performance analysis of ethernet/ip networks, *7th IFAC International Conference on Fieldbuses and Networks in Industrial and Embedded Systems*, Vol.7, 2007.

[17] IEEE Computer Society, IEEE standard for a precision clock synchronization protocol for networked measurement and control systems, *ANSI/IEEE standard 1588-2002*, 2002.