Technical Report

Engineering PROFIBUS Networks with Heterogeneous Transmission Media

Mário Alves

Eduardo Tovar

TR-061003

Version: 1.0

Date: October 2006

Engineering PROFIBUS Networks with Heterogeneous Transmission Media

Anis KOUBAA, Mário ALVES, Eduardo TOVAR

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail: {akoubaa@dei, mjf, emt@dei}.isep.ipp.pt

http://www.hurray.isep.ipp.pt

Abstract

A significant number of process control and factory automation systems use PROFIBUS as the underlying fieldbus communication network. The process of properly setting up a PROFIBUS network is not a straightforward task. In fact, a number of network parameters must be set for guaranteeing the required levels of timeliness and dependability. Engineering PROFIBUS networks is even more subtle when the network includes various physical segments exhibiting heterogeneous specifications, such as bus speed or frame formats, just to mention a few. In this paper we provide underlying theory and a methodology to guarantee the proper operation of such type of heterogeneous PROFIBUS networks. We additionally show how the methodology can be applied to the practical case of PROFIBUS networks containing simultaneously DP (Decentralised Periphery) and PA (Process Automation) segments, two of the most used commercial-off-the-shelf (COTS) PROFIBUS solutions. The importance of the findings is however not limited to this case. The proposed methodology can be generalised to cover other heterogeneous infrastructures. Hybrid wired/wireless solutions are just an example for which an enormous eagerness exists.

Engineering PROFIBUS Networks with Heterogeneous Transmission Media

Mário ALVES*, Eduardo TOVAR

IPP-HURRAY! Research Group
Polytechnic Institute of Porto, School of Engineering
Rua Dr. António Bernardino de Almeida, 431, 4200-072 Porto, Portugal
{mjf, emt}@isep.ipp.pt
Phone: +351.22.8340502 Fax: +351.22.8340509
* corresponding author

Abstract

A significant number of process control and factory automation systems use PROFIBUS as the underlying fieldbus communication network. The process of properly setting up a PROFIBUS network is not a straightforward task. In fact, a number of network parameters must be set for guaranteeing the required levels of timeliness and dependability. Engineering PROFIBUS networks is even more subtle when the network includes various physical segments exhibiting heterogeneous specifications, such as bus speed or frame formats, just to mention a few. In this paper we provide underlying theory and a methodology to guarantee the proper operation of such type of heterogeneous PROFIBUS networks. We additionally show how the methodology can be applied to the practical case of PROFIBUS networks containing simultaneously DP (Decentralised Periphery) and PA (Process Automation) segments, two of the most used commercial-off-the-shelf (COTS) PROFIBUS solutions. The importance of the findings is however not limited to this case. The proposed methodology can be generalised to cover other heterogeneous infrastructures. Hybrid wired/wireless solutions are just an example for which an enormous eagerness exists.

Keywords: fieldbus networks; PROFIBUS; interoperability in heterogeneous networks; real-time communications.

1. Introduction

1.1. Context and structure of the paper

Industrial communication systems have suffered significant changes over the last 20 years or so. Local Area Networks (LANs) have substituted point-to-point communications, initially triggered by big savings in wiring and maintenance costs. The increasing decentralisation of measurement and control tasks, as well as the increasing use of intelligent microprocessor-controlled devices in industrial computer-controlled systems triggered the proliferation of fieldbus networks. A fieldbus network is a specific type of LAN aimed at the interconnection of sensors, actuators and controllers in applications ranging from discrete manufacturing, process control, building automation and in-vehicle control.

Current fieldbus technologies provide real-time, reliable and cost-effective solutions for industrial automation systems. Standard and commercial-off-the-shelf (COTS) fieldbus networks such as PROFIBUS [1], P-NET [1], WorldFIP [1], Foundation Fieldbus [1] or Ethernet/IP [2] offer a panoply of application software packages, functionalities, devices and networking interoperability solutions that make these technologies important building blocks for e-Manufacturing approaches [3].

Typically, industrial automation applications undergo process reengineering, and the underlying communication systems must be adapted and extended accordingly, rather than totally replaced. Industrial communication systems must therefore cope with the need for

interoperability between heterogeneous technologies. Although modern industrial information technologies may play an important role in facilitating the integration and interoperability of applications, this is only profitable if industrial communication infrastructures are still able to provide crucial characteristics, such as timeliness or reliability.

It is in this context that we consider the problem of supporting distributed real-time applications with heterogeneous fieldbus networks. Specifically, we consider the case of fieldbus networks being composed of profiles exhibiting heterogeneous physical layer specifications. We exercise this problem for the most widely used fieldbus - PROFIBUS, with over 14 million nodes installed worldwide [4], namely considering a scenario involving a heterogeneous PROFIBUS-DP/PA network

Setting up a single segment PROFIBUS network for supporting real-time distributed applications is, by itself, a non trivial task. There is the need to compute and set a number of relevant network parameters in order to guarantee bounded message response times, among other system requirements (e.g. [5,6,7,8,9]).

Engineering PROFIBUS networks is even more subtle when the network includes various physical segments exhibiting heterogeneous characteristics, such as bus speed or frame formats. An intuitive solution for the interconnection of the heterogeneous physical segments is using intermediate systems operating at the physical layer level. For simplification, these intermediate systems are labelled as repeaters, and the overall system would then result in a "broadcast" network, where every node listens to every transmitted message (Figure 1).

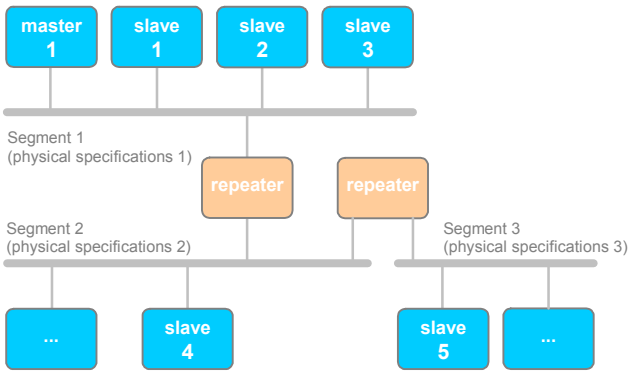

Figure 1: "Broadcast" network with heterogeneous physical media

This approach triggers an important media adaptation problem to be solved. Since the network segments may exhibit different bit rates and different physical layer frame formats, messages may experience unbounded and unpredictable delays (introduced by repeaters' operation). This may be unacceptable for real-time distributed applications.

This paper presents an innovative solution for this media adaptation problem, which relies on the insertion of additional inactivity (idle) periods before the transmission of every request frame (by a master node) in order to guarantee bounded and predictable message response times. PROFIBUS nodes can be masters or slaves, but only master nodes have initiative to start a message transaction. A message transaction usually comprises a request from a master node and an immediate response/acknowledgement from the addressed slave node.

The minimum values for the above mentioned inactivity periods must be computed according to a number of network (e.g. bit rates) and node (e.g. message length) parameters. Then, the standard PROFIBUS Idle Time parameters must be set in every master, prior to run-time. While this may seem a trivial approach, the optimal solution requires a thorough timing analysis, which will be reasoned out throughout this paper.

The remainder of the paper is structured as follows. Section 1.2 overviews related scientific works and COTS technologies. Then, the characteristics of the PROFIBUS Data Link and Physical Layers that are relevant to the context of this paper are presented in Section 2. Section 3 states the problem and outlines the solution. In Sections 5 and 6, a methodology to properly set the PROFIBUS Idle Time and Slot Time parameters, which is a solution to the problem, is discussed and proposed. For this purpose, we use the analytical models (for the repeaters and physical media) early proposed in Section 4. Section 7 instantiates the application of the proposed methodologies to an example scenario involving a heterogeneous PROFIBUS-DP/PA network. Finally, Section 8 draws some conclusions about this work.

1.2. Related work

The heterogeneity of current and future industrial communication systems brings up interoperability problems. Therefore, there is the need to provide the appropriate mechanisms to achieve full interoperability between nodes belonging to different types of networks, such as Fieldbus, Industrial Ethernet and Wireless Local Area Networks (WLANs). We consider the heterogeneity of industrial communication networks due to the coexistence of fieldbus and higher level networks, dissimilar fieldbus networks and separated domains of the same fieldbus network. In all these cases, interoperability is mandatory and must be achieved through the use of appropriate interconnecting devices acting as repeaters, bridges, routers or gateways. This section outlines some related research efforts and commercially available products.

Nowadays, many companies supply solutions for interconnecting field-level networks and higher level networks, mainly motivated by the enormous trend towards Internet access to the factory floor. The “I can access anything from anywhere” concept is definitely driving new strategies to tackle the communication requirements of the modern factory. Most fieldbus manufacturers provide gateways to Ethernet TCP/IP, permitting the access to process data over the Internet (e.g. Siemens, Hilscher, Deutschmann Automation, AEG/Schneider, HMS and Bihl&Wiedemann provide Internet (TCP/IP on top of Ethernet) gateways to several types of fieldbus (e.g. ControlNet, PROFIBUS DP, Interbus, CANopen, AS-I, ModBus). Most of the times, the gateway behaves as a master station in the fieldbus network, maintaining an updated image of process data to be accessed from the Ethernet network. Also, some research works proposed solutions for Internet monitoring, control and maintenance of fieldbus networks (e.g. [10],[11]).

Although fieldbus systems are in widespread use in industry for more than one decade, there is still a significant number of devices that only communicate via a serial data interface (e.g., RS232), usually using Modbus/Modnet higher layer protocols. Several companies provide serial/fieldbus gateways (e.g. Hilscher’s PKV, Deutschmann Automation’s UNIGATE, HMS’s AnyBus) to integrate these legacy systems into several fieldbus networks. These gateways can operate in two different ways: either they maintain an internal image of the (serial) device to which they are connected (proxy-like behaviour), or each individual frame is converted directly between the two protocols.

Several companies provide products for the interoperability between different fieldbus networks (e.g. Bihl&Wiedemann, Siemens, Anybus, Deutschmann Automation). The interconnecting devices usually act as gateway, providing interoperability between fieldbus systems such as AS-I, CAN, PROFIBUS, DeviceNet, LonWorks, ModBus). There is a very limited number of relevant scientific papers addressing this topic. In [12], the authors proposed several gateway architectures to interconnect different fieldbus networks.

There are also some commercially available products for providing wireless extensions to traditional (wired) fieldbus networks, usually based on interconnecting devices operating as

simple repeaters. For example, ALSTOM provides a radio extension to WorldFIP networks and KVASER provides a wireless extension to CAN (WAVEcan). Elprotech, Satel, HMS, RadioLinx, Siemens, Prosoft Technology and Phoenix Contact are companies that provide wireless (radio or infra-red based) extensions to PROFIBUS. Concerning research efforts, it is worthwhile to mention the proposals in [13,14] and [15], since they provide complete architectures on PROFIBUS, where multiple wired segments and multiple wireless cells are interconnected by repeaters (the former) and by bridges (the latter), both supporting inter-cell mobility of nodes and guaranteeing real-time communications. [16] summarizes some architectural approaches for hybrid wired/wireless fieldbus networks.

In this paper, we address the interconnection between different domains of the same fieldbus network (PROFIBUS, in our case). The PROFIBUS standard defines an "extended addressing" scheme, but does not specify some fundamental aspects about how traffic is relayed between nodes belonging to different domains, namely the data transfer mechanisms and time-related issues. [17] analyses the PROFIBUS Standard's guidelines for segmentation and proposes a "bridge-like" behaviour.

In order to fulfil the industrial need to interconnect PROFIBUS-DP and PROFIBUS-PA networks, several companies (e.g. Siemens, Pepperl&Fuchs, Trebing&Himstedt) supply DP/PA Bus Coupler and DP/PA Link products. The PROFIBUS DP/PA Link operates as a "proxy-like" gateway, while the PROFIBUS DP/PA Bus Coupler operates as a repeater. In the former, two different logical rings exist (one on DP and the other on PA). The DP/PA Link device includes one PA master and one DP Slave. The PA master is responsible for maintaining an updated process data image of the PA network which, in turn, may be accessed by a DP master through the DP slave of the DP/PA Link device. On the other hand, the DP/PA Bus Coupler only adapts the asynchronous format and bit rate (93.75 or 45.45 kbit/s, depending on the implementation) of the DP messages and the synchronous format and bit rate (31.25 kbit/s) of the PA messages. Further details on this subject will be provided in Section 7. It is worthwhile to mention that P-NET and ModBus are examples of fieldbuses that provide native solutions for the interconnection (router and bridge-based, respectively) between different network domains.

2. Relevant aspects of PROFIBUS Data Link and Physical Layers

2.1. Overview

The PROFIBUS protocol [1] is based on the OSI (Open Systems Interconnection) reference model [18], although only the Physical Layer (PhL), the Data Link Layer (DLL) and the Application Layer (AL) are defined and implemented.

A maximum of 32 nodes, either masters or slaves, can be supported in a single segment. However, the network can be extended to a maximum of 126 nodes by using more segments in a linear or tree-like topology, provided that the segments use the same physical layer protocol and no more than 3 repeaters exist in the path between any pair of nodes. The maximum cable length for a single segment depends on the bit rate, ranging from 1200 m for lower bit rates (9.6-93.75 kbit/s) down to 100 m, if higher bit rates (3-12 Mbit/s) are used.

A master can send a message on its own initiative, once it receives the token, which circulates between masters in a logical ring fashion. Slaves do not have bus access initiative; therefore they only acknowledge or respond to requests from masters. A message cycle (or transaction) comprises the request frame sent by an initiator (always a master) and the associated acknowledgement or response frame from the responder (usually a slave).

The PROFIBUS Medium Access Control (MAC) protocol, being based on the measurement of the actual token rotation time, induces a well-defined timing behaviour for the transferred messages, since the token cycle duration can be estimated prior to run-time [6].

After a master issues a request frame, the corresponding acknowledgement or response frame must arrive before the expiration of the Slot Time (T_{SL}), otherwise the initiator repeats the request or aborts the transaction. Therefore, the message turnaround time (T_{rt} – the time span since a request frame is completely transmitted by the initiator, until it starts receiving the corresponding response frame), must always be smaller than T_{SL}. Figure 2 illustrates a scenario where a first message transaction has succeeded, followed by another message transaction where an error occurred (response did not arrive to the master before T_{SL} expired).

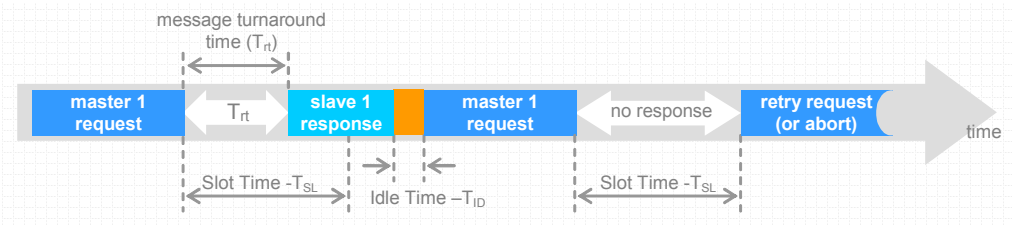

Figure 2: The PROFIBUS Slot Time (T_{SL}) and Idle Time (T_{ID}) parameters

Before issuing a request (or token) frame, the master must wait a time interval defined by the Idle Time (T_{ID}) parameter (also illustrated in Figure 2), in order to create an inter-frame synchronising period of idle bits (at least 33 idle bit periods) [1].

Both T_{SL} and T_{ID} are standard PROFIBUS parameters that must be properly set (in master nodes) prior to run-time. As it will be clear throughout the remainder of this paper, these parameters are of particular importance for engineering heterogeneous PROFIBUS networks, and therefore additional reasoning on these two parameters is provided in Sections 2.3 and 2.4 for T_{ID} and T_{SL}, respectively.

2.2. Frame formats

PROFIBUS defines 4 types of Data Link Layer (DLL) frames, each of them characterised by a different Start Delimiter (SD) identifier. Frame formats and contents for these 4 types are depicted in Figure 3a) to d). Two Physical Layer (PhL) frame formats are also outlined – the asynchronous (RS-485) and the synchronous (MBP – Manchester coding Bus Powered) specifications.

In the RS-485 physical layer, usually used in PROFIBUS-DP implementations, each frame is coded using UART (Universal Asynchronous Receiver/Transmitter) characters, each comprising 11 bits: 1 start bit, 8 data bits, 1 (even) parity bit and 1 stop bit (Figure 3e)). The MBP version is normally used in PROFIBUS-PA intrinsic safety applications. The main differences to the RS-485 version is that every character of the DLL is coded in 8 bits (Figure 3f)), and the frame contains a synchronisation preamble and increased error checking, since a 2 bytes CRC (Cyclic Redundancy Check) substitutes the 1 byte FCS (Frame Check Sequence) field that appears in the RS-485 version.

In the MBP specification, each frame starts with a preamble of at least 1 octet, to synchronise the receiver, followed by a special start delimiter (PA SD, in Figure 3), and by the PROFIBUS DLL frame. The MBP PhL has a specific end delimiter (PA ED, in Figure 3).

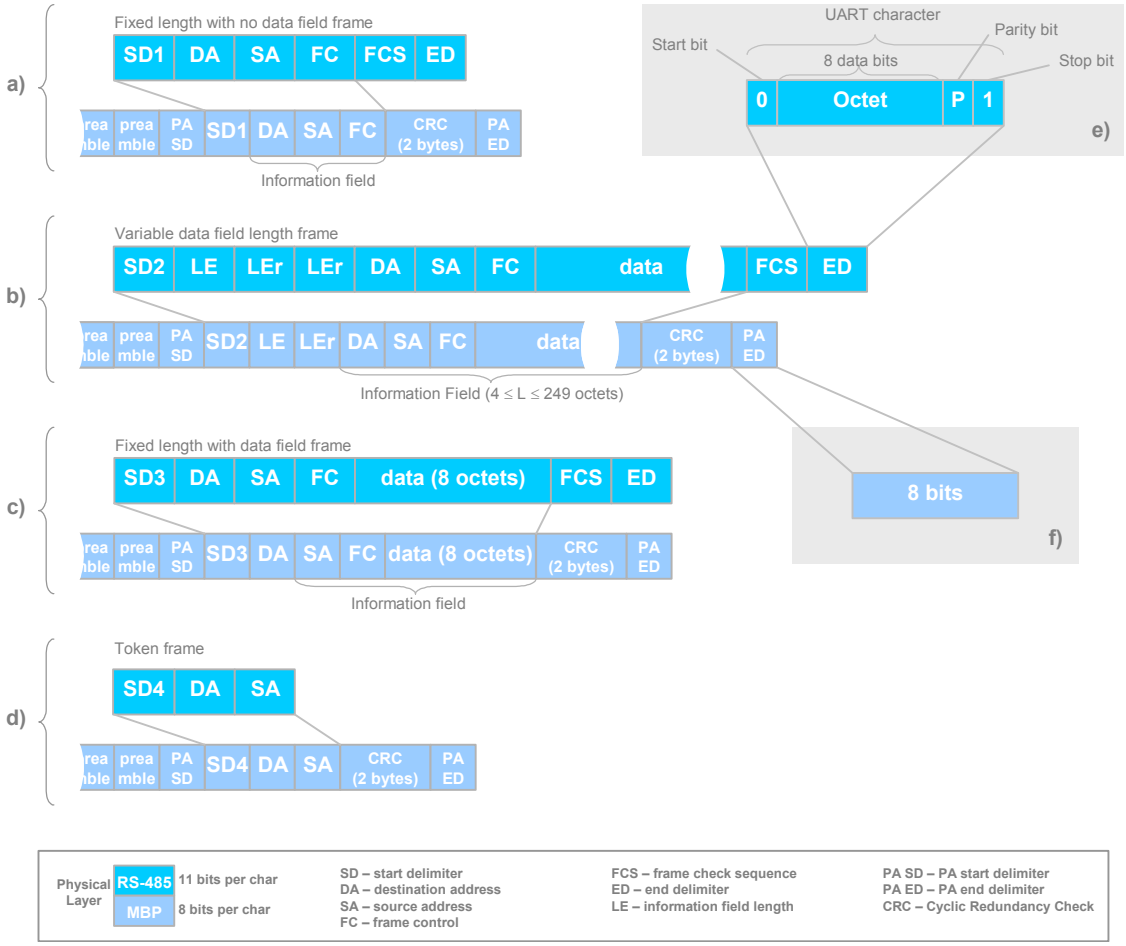

Figure 3: PROFIBUS frame formats (RS-485 and MBP physical media)

2.3. Further details on the Idle Time (T_{ID}) parameters

The Idle Time is a period of physical medium inactivity that is inserted by master stations between consecutive message transactions. After an acknowledgement, response or token frame, a master station inserts an idle time with a value given by:

$$T_{ID1} = \max\left\{T_{SYN} + T_{SM}, \min\left\{T^{i}_{SDR}\right\}, T_{SDI}\right\}, \ \forall i \in \text{RespondersSet} \qquad \textbf{(1)}$$

T_{SYN} (synchronisation time) is the minimum time interval during which each station must receive idle state from the physical medium (33 bits); T_{SM} is a safety margin; T^{i}_{SDR} is the station delay of responder i; T_{SDI} is the station delay of the initiator. Refer to the PROFIBUS standard [1] for further details on these parameters.

Conversely, after an *unacknowledged request frame*, a master station must insert an idle time given by:

$$T_{ID2} = \max\left\{T_{SYN} + T_{SM}, \max\left\{T^{i}_{SDR}\right\}\right\}, \ \forall i \in \text{Responders Set} \qquad \textbf{(2)}$$

Figure 4 illustrates the use of T_{ID1} (Figure 4a) and T_{ID2} (Figure 4b).

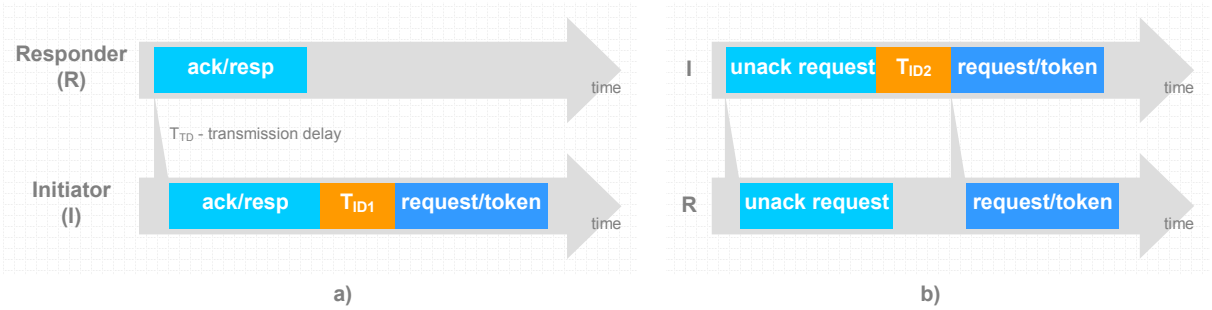

Figure 4: Illustration of the Idle Time parameters – a)T_{ID1} and b)T_{ID2}

Throughout the paper, parameters denoted as 'T' represent bits while parameters denoted as 't' represent time. Additionally, the station delay of the responder t_{SDR} (T_{SDR} in time units) will be referred as *responder's turnaround time* – t_{rt}. This is the time span since a request frame is completely received by the responder until it starts transmitting the correspondent response frame.

The idle time parameters can be set in a per-station basis; that is, each master can hold a different value for the (T_{ID1}, T_{ID2}) pair. Eq. (1) and (2) are valid for a single segment network. As it will be seen in Section 5, in a multiple segment network composed of physical media with different bit rates and (Physical Layer) frame formats, the idle time parameters must be derived differently.

2.4. Further details on the Slot Time (T_{SL}) parameter

The Slot Time is a parameter used by a master node to detect communication or node errors that lead to abnormal medium inactivity. A master node always checks if the time elapsed between the transmission of the last bit of a request (or token) frame and the reception of the first character of the following frame (transmitted by another node) is smaller than T_{SL}. If this does not happen, the master retransmits the frame (request or token) or aborts the transmission.

To set the T_{SL} parameter, it is necessary to compute two different components: T_{SL1} and T_{SL2}. T_{SL1} is the maximum time the initiator waits for the complete reception of the first character of the acknowledgement/response frame from the responder (*R*), after transmitting the last bit of the request frame (Figure 5a).

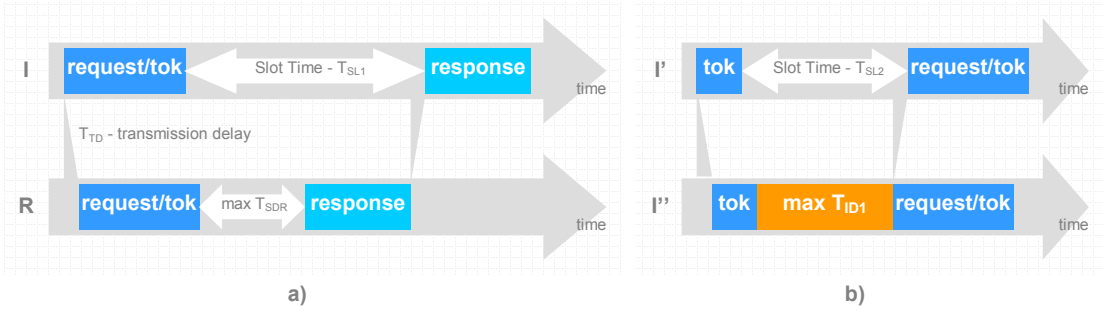

Figure 5: Illustration of the Slot Time components – a)T_{SL1} and b)T_{SL2}

T_{SL1} can be computed as follows:

$$T_{SL1} = 2 \cdot T_{TD} + \max\{T^{i}_{SDR}\} + 11 + T_{SM}, \ \forall i \in \text{RespondersSet} \qquad \textbf{(3)}$$

T_{TD} is the transmission (propagation) delay; T^{i}_{SDR} is the station delay of responder *i*; T_{SM} is a safety margin.

T_{SL2} is the maximum time a master node (*I'* in Figure 5b) waits after having transmitted the last bit of the token frame until it completely receives the first character of a frame (either a request or the token) transmitted by the master node that received the token (*I''* in Figure 5b). T_{SL2} can be computed as follows:

$$T_{SL2} = 2 \cdot T_{TD} + \max\left\{T_{ID1}^{i}\right\} + 11 + T_{SM}, \ \forall i \in \text{InitiatorsSet} \quad \textbf{(4)}$$

Contrarily to the Idle Time parameters, the Slot Time parameter must be set with the same value in every master in the network (this is imposed by the token passing mechanism), which is the maximum between T_{SL1} and T_{SL2}:

$$T_{SL} = \max\{T_{SL1}, T_{SL2}\} \quad \textbf{(5)}$$

While Eqs. (4) and (5) are valid for a single segment network, for a network with multiple heterogeneous segments interconnected by repeaters, the appropriate T_{SL} value must be determined using a more elaborated reasoning. This will be addressed in Section 6.

3. The media adaptation problem

3.1. The problem

As mentioned before, a master must receive the response to a request within the Slot Time (T_{SL}). If a timeout occurs, the master retries the request or aborts the transmission. In a network composed of several heterogeneous segments interconnected by repeaters, message turnaround times will increase, due to relaying latencies in the repeaters. These latencies result from the fact that the repeaters must relay frames between segments with different physical layer frame formats and different bit rates, as it will be clarified in Section 4.

Consider the network scenario previously outlined in Figure 1. As depicted in the timing diagram of Figure 6a), the turnaround time for a message transaction between an initiator and a responder belonging to the same network segment (e.g. master 1 and slave 3) is the traditional responder's turnaround time for a PROFIBUS responder node (t_{rt}). However, when initiator and responder belong to different segments (e.g., master 1 and slave 4), the turnaround time will increase, as a consequence of the relaying action performed by the repeater (Figure 6b)). In this paper, this end-to-end turnaround time is denoted as system turnaround time (t_{st}), which includes a start relaying delay (t_{sr}) introduced by repeaters.

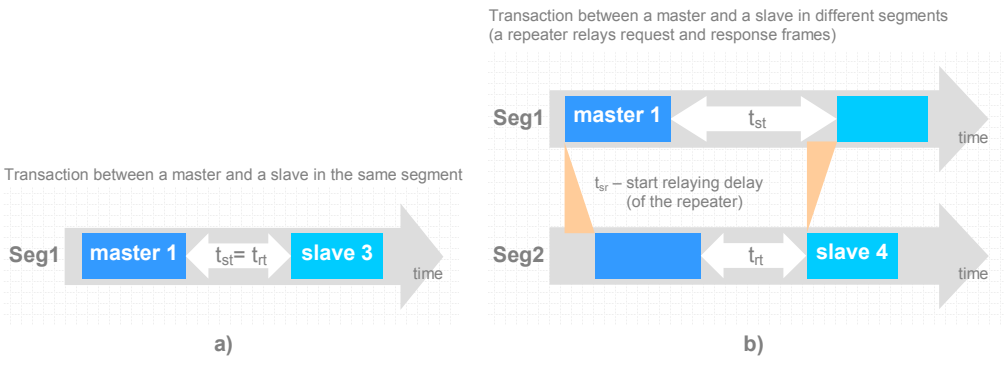

Figure 6: Turnaround times with single (a) and multiple (b) segments

An obvious problem is that frames may be affected by unbounded queuing delays in the repeaters. This is exemplified in Figure 7, where media heterogeneity is assumed to result from different bit rates and/or different PhL frame formats (and therefore different frame durations) for the two segments. The bit rate is lower in segment 2 (Seg2).

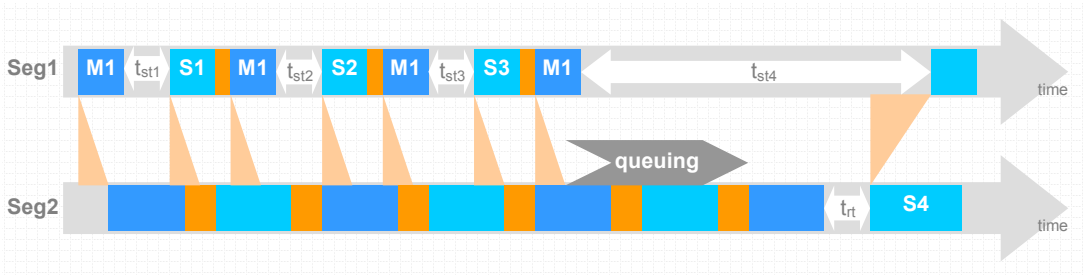

Figure 7: The media adaptation problem – unbounded queuing delay

As can be seen, the fourth message transaction (between master M1 and slave S4) is affected by queuing delays that were originated by a sequence of 3 transactions between M1 and slaves (S1-3) in Seg1, imposing a system turnaround time t_{st4} that is much longer than the responder's turnaround time (t_{rt}).

In fact, the queuing delay can be unbounded. Consider the following elucidative example. Assuming M1 as the only master in the network, several token frames could be transmitted consecutively during a certain time interval, due to the master having no messages to transmit. If after that sequence of self-passing the token M1 initiates a message transaction with a slave in another segment (e.g. S4), that request frame will experience a significant queuing delay in the repeater, since the repeater must first relay all pending token frames to the "slower" network segment (Seg2). This results from the "broadcast" nature of the system.

Since, generically, these queuing delays cannot be bounded [14], it would not be possible to compute an upper bound for the system turnaround time of message transactions between initiator and responder in different segments. Note that system turnaround times are crucial to find a minimum value for setting the T_{SL} parameter in the master nodes. Additionally, high values for t_{st} (and therefore for T_{SL}) may result in an inadmissibly low responsiveness to failures.

3.2. The solution

An intuitive solution to this problem relies on delaying request frames by inserting additional idle time between every transmitted frame, in master nodes [14]. This is depicted in the timing diagram of Figure 8, where t_{st4} is significantly reduced when compared to the scenario of Figure 7, if M1 inserts additional idle periods (t_{ID1+}) before issuing request frames.

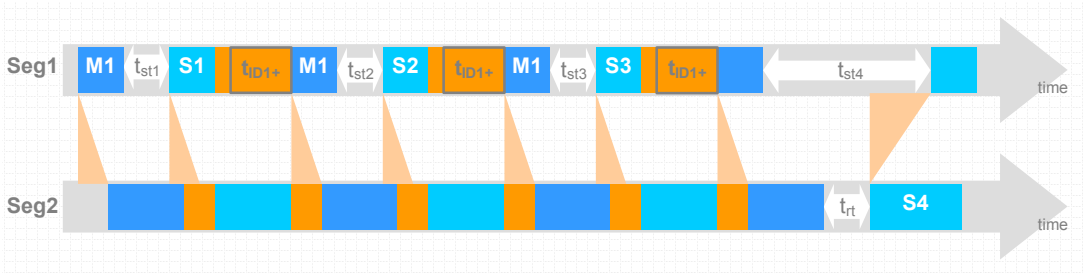

Figure 8: The media adaptation solution – inserting extra idle times

The only drawback resulting from the insertion of additional idle times is a potential reduction of network throughput when the responder is in the same segment as the initiator. However, eliminating unpredictable delays is mandatory for the proper operation of the system. Additionally, a better responsiveness to failures is attained, since T_{SL} will be potentially smaller (if errors occur, retransmissions are undertaken sooner). This will become clearer in Section 6.

Importantly, this mechanism relies on standard features of the PROFIBUS protocol – the Idle Time parameters. Section 5 outlines the methodologies to compute the minimum values for T_{ID1} and T_{ID2} according to a number of network (e.g. bit rates) and node (e.g. message length) parameters.

4. Outline of the analytical models of physical media and repeaters

In this section, we outline the analytical models of the network components that most affect the timing behaviour of the network – repeaters and physical media. The model for the Physical Media mainly defines the bit rate and the Physical Layer frame format, while the model for the repeaters characterises its relaying behaviour.

4.1. About the *model for the physical media*

A physical medium can be modelled with the following parameters: r - bit rate; l_H - overhead of the head per PhL frame; l_T - overhead of the tail per PhL frame; k - overhead per char for the PhL protocol; o - offset defining the total number of bits until knowing the length of the data field.

The generic format of a PhL frame is as depicted in Figure 9. We assume that the DLL frame is embedded in the data field of the PhL frame. It should be noted that the offset o is a relevant parameter for the definition of the timing behaviour of the repeaters (which will be only briefly outlined in this paper).

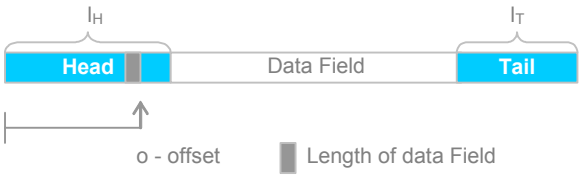

Figure 9: Generic format of a PhL frame

In order to compute the duration of a PhL frame, two Data Link Layer parameters must be considered: L - length of the DLL frame; d - number of bits per DLL char. The duration (C) of a PhL frame in segment D^i is then given by:

$$C^i = \frac{l_H^i + L \cdot \left(d + k^i\right) + l_T^i}{r^i} \qquad \textbf{(6)}$$

Further details on the physical media model can be found in [14].

4.2. About the *model for the repeaters*

Both cut-through and store&forward relaying behaviours are considered in the model for the repeaters. A minimised latency repeater (cut-through behaviour) is a repeater that starts relaying PhL frames as early as possible. A store&forward behaviour is a particular case of the generic cut-through behaviour, where a PhL frame must be completely received by the input port of the repeater before being retransmitted to the output port.

Since the repeaters may interconnect different physical media, it is assumed that they must support some sort of encapsulation/decapsulation mechanism (due to different PhL frame formats) and that they are able to receive/transmit at different bit rates.

In order to define the timing behaviour of the repeater, a start-relaying instant function – $t^{i \to j}_{sr}$ – is defined. It enables the computation of the earliest time instant for start relaying a specific PhL frame from segment Seg^i to segment Seg^j, measured from the beginning of the

transmission of the PhL frame in segment Seg^i. The start-relaying instant for a specific repeater depends on its behaviour – either store&forward or cut-through. For a cut-through repeater, the following was assumed:

1. relaying a frame from Seg^i to Seg^j cannot start before the first char of the DLL frame of Seg^i is completely received by the repeater;
2. the PhL frame cannot start being relayed before the length of the DLL frame is known (by the repeater);
3. when relaying a frame from Seg^i to Seg^j, the instant for start relaying the PhL frame must take into account that the repeater cannot run out of bits to relay from Seg^i to Seg^j, i.e. the transmission of a PhL frame in Seg^j must be continuous, without time gaps.

Taking these assumptions into account, the start-relaying instant for a cut-through repeater is given by:

$$t_{sr}^{i\to j} = \max\left\{t_{dr}^{i}, t_{lk}^{i}, t_{ng}^{i\to j}\right\} \quad \forall i,j \in \text{Segment Set} \tag{7}$$

Concerning Eq. (7), t^i_{dr} , the *data ready* instant, is the time instant at which a predefined amount of DLL data has been received from Seg^i (ready to be relayed). For the cut-through behaviour, it is considered that it is the instant at which the first DLL char is completely received. t^i_{lk} , the *length known* instant, is the instant at which the length of the DLL frame in Seg^i is known. In this case, the offset value for the correspondent Physical Medium is used.

$t^{i\to j}_{ng}$, the *no gaps* instant, is the earliest instant to start relaying the PhL frame from Seg^i to Seg^j in a way that guarantees that the transmission in Seg^j is continuous. It may be computed by subtracting the duration of the PhL frames (neglecting the tail) in Seg^i and Seg^j, and subtracting the duration $((d+k^j)/r^j)$ of the last DLL frame char in Seg^j.

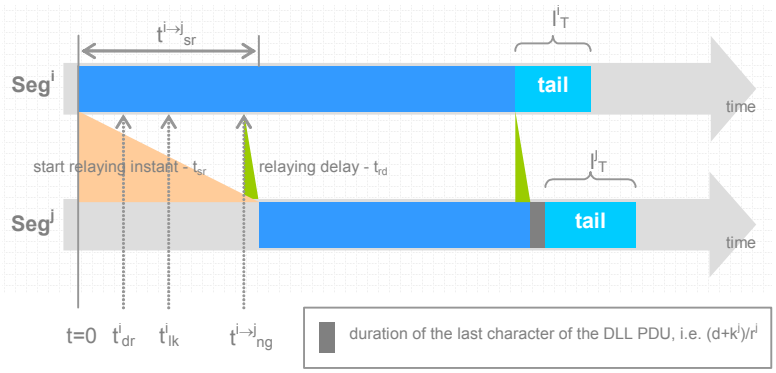

Figure 10: Relaying behaviour of a (cut-through) repeater

Consider the example depicted in Figure 10. The first time instant is data ready (t^i_{dr}), followed by the time instant when the length of the frame is known (t^i_{lk}). The last instant (thus the highest of the three) is the time instant that guarantees a continuous retransmission of the PhL frame ($t^{i\to j}_{ng}$). This situation usually happens when the duration of the PhL frame in Seg^j is smaller than in Seg^i. Nevertheless, and for the general case, any of these time instants can be the highest value between them.

Further details on the repeaters model can be found in [14].

5. Adapting heterogeneous physical media through the insertion of extra idle time

5.1. Outline of the methodology for setting the Idle Time parameters

As it was previously mentioned, our solution for media adaptation relies on master nodes introducing additional inactivity times between consecutive frames in the network.

For a single segment PROFIBUS network, the Idle Time parameters of every master must be set to the minimum default values (T_{ID1m} , T_{ID2m}), which is usually adequate to cope with bit synchronisation requirements. For a PROFIBUS network with heterogeneous physical media, the Idle Time parameters must be set considering three different situations, which are illustrated in Figure 11a), b) and c).

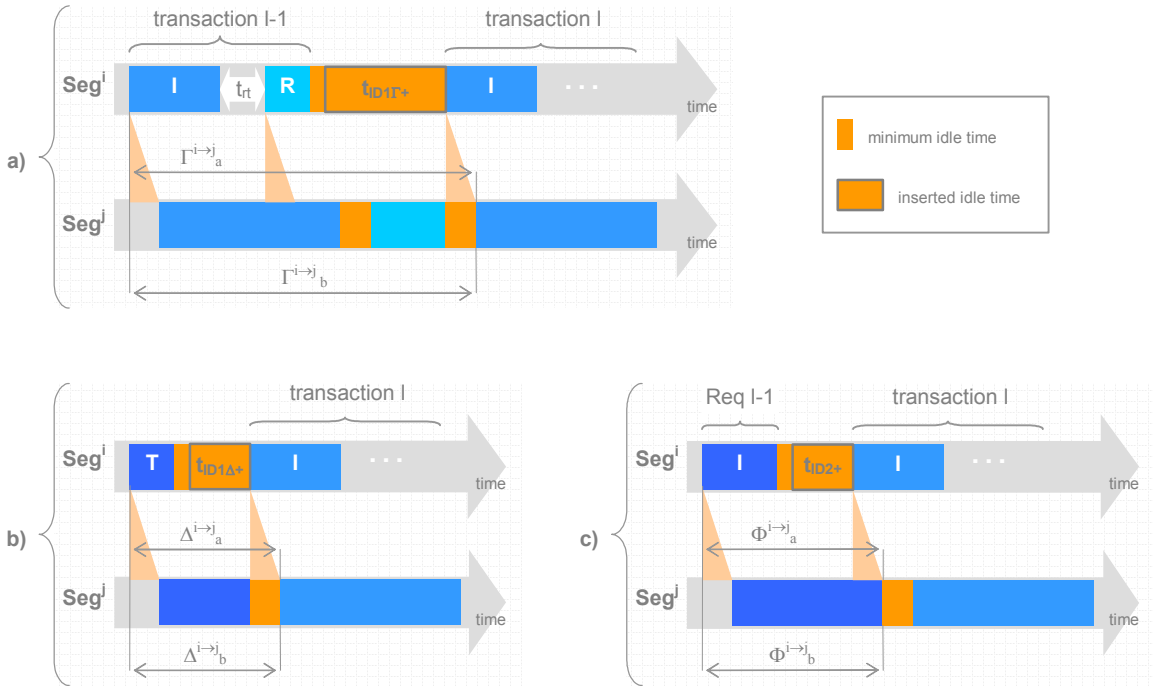

Figure 11: Inserting additional idle times after
a) acknowledged request; b) receiving the token; c) an unacknowledged request

Considering *Situation a)*, after a master receives a response or acknowledgement to a request and before transmitting another request or the token, it must insert additional idle time to guarantee that $\Gamma^{i\to j}{}_a \geq \Gamma^{i\to j}{}_b$ (for every physical media $j \neq i$) to avoid queuing delay in the repeater. This inequality (details not presented here) leads to the computation of $t^{i\to j}{}_{ID1\Gamma+}$ – the first component of $t^{i\to j}{}_{ID1}$.

In *Situation b)*, after a master receives the token and before transmitting another request or the token, it must insert additional idle time to guarantee that $\Delta^{i\to j}{}_a \geq \Delta^{i\to j}{}_b$ (for every physical media $j \neq i$) to avoid queuing delay in the repeater. This inequality (details not presented here) leads to the computation of $t^{i\to j}{}_{ID1\Delta+}$ – the other component of $t^{i\to j}{}_{ID1}$.

The PROFIBUS T_{ID1} parameter is the idle time a master must insert after receiving a response PDU or after receiving the token PDU. Taking this into account, the inserted idle time $t^{i\to j}{}_{ID1+}$ is defined as the maximum between $t^{i\to j}{}_{ID1\Gamma+}$ and $t^{i\to j}{}_{ID1\Delta+}$, i.e.:

$$t_{ID1+}^{i\to j} = \max\left\{t_{ID1\Gamma+}^{i\to j}, t_{ID1\Delta+}^{i\to j}\right\},$$
$$\begin{cases}\text{for every physical media } i, j \\ \text{for every DLL PDUs length for the message streams of the master}\end{cases} \quad (8)$$

Noting that, for a given master, the idle time must be set prior to run time and the same value will be used for all acknowledged transactions, $t^{i\to j}_{ID1\Gamma+}$ must be set to the worst-case (maximum) scenario imposed by all the message streams of the master under consideration. Therefore, t^{i}_{ID1+} is defined as:

$$t_{ID1+}^{i} = \max\left\{t_{ID1+}^{i\to j}\right\}, \text{ for every physical media } j \quad (9)$$

Finally, in *Situation c)*, after a master node transmits an unacknowledged request and before transmitting another request or the token, it must insert additional idle time to guarantee that $\Phi^{j\to y}{}_a \geq \Phi^{j\to y}{}_b$ (for every physical media $j \neq i$) to avoid queuing delay in the repeater. This inequality (details not presented here) leads to the computation of $t^{i\to j}_{ID2+}$.

Again, the idle time must be set prior to run time, which implies finding a worst-case value for $t^{i\to j}_{ID2+}$, for every physical media, i.e.:

$$t_{ID2+}^{i} = \max\left\{t_{ID2+}^{i\to j}\right\}, \text{ for every physical media } j \quad (10)$$

Taking into account that the PROFIBUS protocol supports only one register for T_{ID1} and another register for T_{ID2}, there is the need to aggregate both the "minimum" idle times T_{IDXm} with the inserted idle times T_{IDX+} in one variable (in bit times), for each master station, i.e.:

$$T_{ID1}^{i} = T_{ID1m} + \left\lceil t_{ID1+}^{i} \cdot r^{i} \right\rceil \qquad T_{ID2}^{i} = T_{ID2m} + \left\lceil t_{ID2+}^{i} \cdot r^{i} \right\rceil \quad (11)$$

Where t^{i}_{ID1+} and t^{i}_{ID2+} represent the additional inserted idle times and r^{i} denotes the bit rate of the physical medium (i) the master belongs to.

The detailed methodology and analytical formulation to compute the optimal Idle Time parameter values can be found in [14].

5.2. Simplified algorithm for the computation of the Idle Time parameters

The methodology outlined above permits to set both idle time parameters individually for each master in the network, taking into account all possible transactions (message streams) for that master. In this sense, each master in the network would have a unique pair (T_{ID1}, T_{ID2}) of idle time parameter values. For the sake of simplicity, we consider a simplified algorithm that returns the same idle time parameter values for all masters in a given physical medium.

Therefore, instead of considering the particular set of message streams for each master station, a worst-case scenario where maximum and minimum PDU lengths for the (overall) network is considered. This requires the definition of the following additional network-specific parameters: $L^{max}{}_{req}$ – maximum length of DLL request frame; $L^{max}{}_{resp}$ – maximum length of DLL response frame; $L^{min}{}_{req}$ – minimum length of DLL request frame; $L^{min}{}_{resp}$ – minimum length of DLL response frame. Moreover, acknowledged and unacknowledged DLL request PDUs are considered to have the same maximum and minimum lengths.

A pseudo-code algorithm that computes the values for T_{ID1} and T_{ID2} is outlined in Figure 12. Considering the context of this paper, and for the sake of simplicity, several computation details are omitted (refer to [14]).

```
1:  Begin
2:    /* Input network-specific parameters */
3:    input(L_token , L^max_req , L^max_resp , L^min_req , L^min_resp , t^min_rt , T_IDm , d, t_rd , nm)
4:    /* Set minimum idle time parameters to the same value */
5:    T_ID1m = T_ID2m = T_IDm
6:    For i=1 to n /* For every physical medium */
7:       /* Read Physical Medium-specific parameters *)
8:       input(r^i, l^i_H, l^i_T, k^i, o^i)
9:       For j=1 to n  /* For every physical medium */
10:           if j <> i then   /* other than i (j≠i) */
11:              Compute t^{i→j}_{ID1Γ+} (details not presented)
12:              Compute t^{i→j}_{ID1Δ+} (details not presented)
13:              t^{i→j}_{ID1+} = max{t^{i→j}_{ID1Γ+}, t^{i→j}_{ID1Δ+}}  /* Eq. (8) */
14:              Compute t^{i→j}_{ID2+} (details not presented)
15:           endif /* if j <> i */
16:      endfor /* For j */
17:      t^i_{ID1+} = max{t^{i→j}_{ID1+}}   /* Eq. (9) */
18:      t^i_{ID2+} = max{t^{i→j}_{ID2+}}   /* Eq. (10) */
19:      T^i_{ID1} = T_{ID1m} + ⌈t^i_{ID1+} · r^i⌉  /* Eq. (11) */
20:      T^i_{ID2} = T_{ID2m} + ⌈t^i_{ID2+} · r^i⌉  /* Eq. (11) */
21:   endfor /* For i */
22: end.
```

Figure 12: Simplified algorithm for the computation of T_{ID1} and T_{ID2}

6. Computation of the worst-case duration of message transactions and of the Slot Time parameter

6.1. *The impact of increased latencies on the duration of message transactions and on T_{SL}*

The message's response time in a heterogeneous PROFIBUS network such as the one considered is dependent on the medium access delay (contention due to other messages in the queue and due to other nodes holding the token) and on the duration of the message transaction. Such duration includes both the duration of the request/response frames and the system turnaround time associated with that transaction, that is, the time interval between the end of the request transmission and the beginning of the response reception.

When considering the case of PROFIBUS networks with multiple segments, there may exist several repeaters between initiator and responder. Therefore, system turnaround times can be several orders of magnitude higher than the duration of the request/response frames themselves. This is illustrated in the timing diagram of Figure 13, considering the network scenario illustrated in Figure 1. A transaction between M1 (in segment 1) and S5 (in segment 4) must be relayed through two repeaters. C_{ack} is the duration of the message transaction and t_{st} is the system turnaround time for that transaction.

In this context, the PROFIBUS Slot Time parameter (T_{SL}) assumes a particular importance. On one hand, T_{SL} must be set large enough to cope with the extra latencies introduced by the repeaters. On the other hand, T_{SL} must be set as small as possible, such as the system responsiveness to failures does not decrease dramatically; that is, a master must detect a message/token loss or a node failure within an acceptable time interval. Moreover, and in the context of a pre-run-time schedulability analysis of PROFIBUS messages [e.g. 5,9], it becomes obvious that as T_{SL} is a time component of the worst-case duration of a message transaction, its value will impact the evaluation of the worst-case message response time.

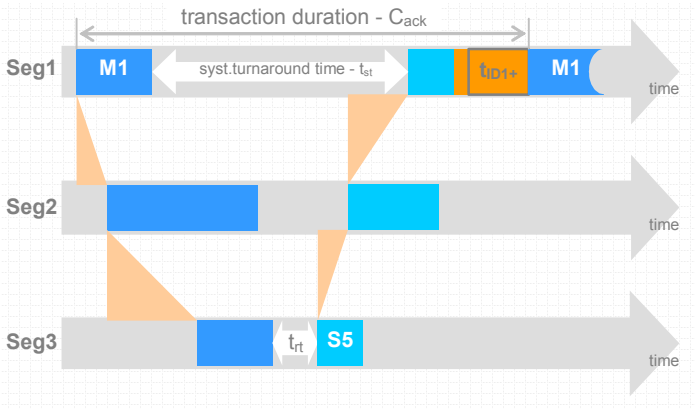

Figure 13: Duration and system turnaround time of a message transaction

The computation of the worst-case system turnaround time (t_{st}) for every message transaction in the network permits to compute one of the components of the PROFIBUS Slot Time parameter – T_{SL1} – in all master nodes. This parameter defines the timeout before which a response/acknowledgement must arrive (for every message transaction), and it is also used for the token recovery mechanism. The same reasoning, applied to the case where a master node passes the token and waits for the next master node to transmit, permits to compute T_{SL2}. The Slot Time – T_{SL} – must be set to the maximum between T_{SL1} and T_{SL2}, prior to run-time.

The remainder of this section outlines the methodologies to compute the worst-case system turnaround time for every message transaction in the network (t_{st}), the duration of acknowledged (C_{ack}) and unacknowledged (C_{unk}) message transactions and of the PROFIBUS Slot Time parameter (T_{SL}).

6.2. *An outlook of the methodology to compute worst-case transactions duration and the Slot Time parameter*

In order to guarantee the real-time behaviour of a multiple segment PROFIBUS network, there is the need to compute the worst-case duration of every message transaction (stream) and also to determine the value for the Slot Time parameter that will be (equally) set in all master nodes in the network. For this purpose, it is necessary to follow a rigorous methodology [14], based on the (ordered) computation of the following parameters:

1. t_{st} – the worst-case system turnaround time for every message transaction in the network;
2. T_{SL1} – one of the components of the PROFIBUS Slot Time parameter, based on the worst-case system turnaround time of all message transactions;
3. C_{ack} and C_{unk} – the duration of acknowledged and unacknowledged message transactions, respectively;
4. T_{SL2} – the second component of the PROFIBUS Slot Time parameter, based on the worst-case system turnaround time after token passing;
5. T_{SL} – the PROFIBUS slot time parameter, based on the maximum value between T_{SL1} and T_{SL2}.

Figure 14 shows an example of the time variables involved in the computation of the (worst-case) duration (C_{ack}) of a request/response message transaction (transaction l), for the network scenario presented in Figure 1.

We can easily derive that C_{ack} it is the sum of the duration of the request frame (C_{Lreq}), the worst-case system turnaround time (t_{st}), the duration of the response frame (C_{Lresp}), and finally the idle time that must be inserted before transmitting the following frame (t_{ID1}).

$$C_{ack} = C^1_{Lreq} + t_{st} + C^1_{Lresp} + t^1_{ID1m} + t^1_{ID1+} \quad \textbf{(12)}$$

Nevertheless, the worst-case system turnaround time (t_{st}) involves an important component - Q – that deserves some attention, as briefly explained next.

It has been proved [14] that the inserted idle time guarantees that there is no increasing queuing delays in the repeaters. Nevertheless, there may occur (bounded) queuing delays in some repeaters (except the first) between initiator and responder of a transaction (or between a master and its successor, when passing the token). Consequently, the insertion of additional idle time enables the computation of the worst-case queuing delay – Q – affecting any request frame. Such worst-case queuing delay will be a component of the worst-case system turnaround time for any message transaction ($t_{st} = Q + t_{stn}$), where t_{stn} is used to denote the system turnaround time assuming no queuing delay.

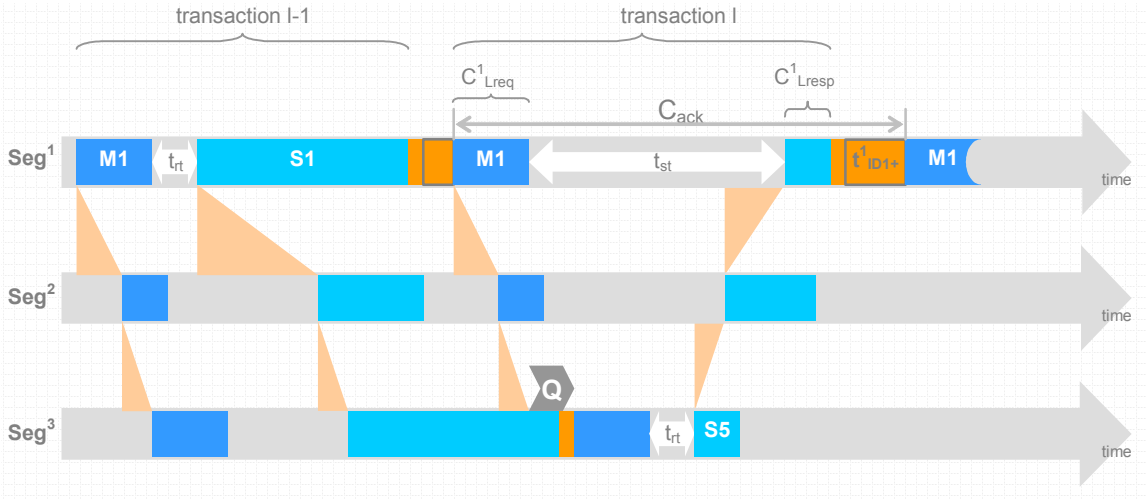

Figure 14: Request affected by a queuing delay – *Q*

Figure 14 also depicts an example case where the request frame of transaction *l* is affected by a queuing delay (Q) in the second repeater between initiator and responder. This additional latency is caused by the fact that the repeater connecting Seg2 and Seg3 is still relaying the response frame of the previous transaction (*l*-1), when the request of transaction *l* arrives. Obviously, Q will be a component of the system turnaround time of that transaction, and thus it must be considered in the computation of the worst-case system turnaround time.

Due to size restrictions, the detailed methodology and analytical formulation to compute the the worst-case system turnaround times, the worst-case duration of message transactions and the Slot Time parameter values can be found in [14].

7. Case Study – PROFIBUS DP/PA interoperability

The methodologies outlined in this paper can be generically applied to a network with multiple segments and different physical media. This section analyses the particular case of a DP/PA network interconnected by a repeater (usually known as PROFIBUS coupler).

7.1. *Context*

Some PROFIBUS installations require the interconnection of different physical media. For instance, when interoperability between factory automation and process automation field devices must be provided, there is the need to link PROFIBUS-DP (Decentralised Periphery) and PROFIBUS-PA (Process Automation) segments. Since process automation devices are often located in areas subject to the danger of explosions, intrinsically safe network devices must be used. Therefore, PROFIBUS provides two physical layer technologies that fulfil the appropriate requirements for these situations (limited voltage/current, power over the transmission medium): MBP-IS (EEx ia/ib), synchronous transmission running at a fixed data rate of 31.25 kbit/s, and more recently RS485-IS (EEx ib), with the traditional asynchronous transmission with data rates ranging from 9.6 kbit/s to 1.5 Mbit/s.

Commercial products for the interconnection between PROFIBUS-DP and PROFIBUS-PA networks usually fit into one of two types: PROFIBUS (segment) *coupler* or PROFIBUS *link*. A PROFIBUS coupler is a repeater with some data link layer functionality, since it not only adapts different (DP/PA) bit rates, but also converts between synchronous/asynchronous physical/data link layer frame formats (refer to Figure 3 for details on PROFIBUS frame format). With this device, only one PROFIBUS-PA segment can be connected and only one bit rate is admissible on the DP side, usually below 100 kbit/s (e.g. 45.45 kbit/s in Siemens [19] and 93.75 kbit/s in Pepperl+Fuchs [20] and solutions) and as close as possible to the PA bit rate (31.25 kbit/s). The fact that a fixed and low bit rate must be respected by the DP segment is most probably due to bit rate adaptation problems which are not reported in the respective PROFIBUS coupler manuals (e.g. [19, 20]).

PROFIBUS *link* devices open up the possibility of having several PROFIBUS-PA segments connected to a PROFIBUS-DP segment. Also, the bit rate of the PROFIBUS-DP segment can be freely set in the range from 9.6 kbit/s to 12 Mbit/s. This is true since the PROFIBUS link acts as a proxy gateway, where all field devices (PROFIBUS slaves) belonging to a MBP segment are mapped as a single slave in the RS-485 segment. In this case, even if the bit rates of the RS485 and MBP sides are highly unbalanced, there is no speed adaptation problem. However, the functionality of an asynchronous gateway is limited, due to the complexity to maintain a dynamic image of process data, namely for networks with a significant number of nodes. Therefore, this type of proxy-like gateways is only adequate for networks with just two segments.

7.2. *Characteristics of the example network*

In this case study, we will assume the network topology depicted in Figure 15.

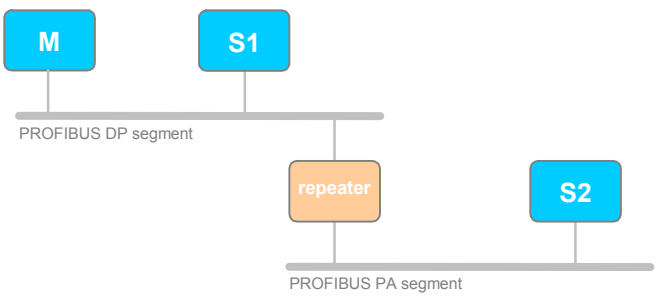

Figure 15: Example of a PROFIBUS DP/PA network

According to the physical media model that was defined in Section 4.1, the DP (RS-485) and PA (MBP) physical media are defined by the parameters (refer to Figure 3, for further details) presented in Table 1.

Table 1: Physical layer parameters for the example network

	DP (RS-485)	PA (MBP)
bit rate – r (kbit/s)	$r^{DP} = 93.75$ [a]	$r^{PA} = 31.25$
head - l_H (bits)	$l_H^{DP} = 0$	$l_H^{PA} = 16$ [b]
tail – l_T (bits)	$l_T^{DP} = 22$ [c]	$l_T^{PA} = 24$ [d]
overhead per char – k (bits/char)	$k^{DP} = 3$ [e]	$k^{PA} = 0$ [f]
offset – o (bits)	$o^{DP} = 33$ [g]	$o^{PA} = 40$ [h]

For Table 1, consider the following:

a) We assume the DP bit rate adopted by some manufacturers (e.g. Pepper+Fuchs, ABB);

b) We assume a two octets preamble for the MBP frame;

c) Taking into consideration that the FCS and ED field of the DP frame are removed, when relaying to the PA side, we assume 2 chars (2x11 = 22 bits) of *tail* on the DP side. The token is an exception, since it has neither FCS nor ED, therefore a null tail is assumed for the token;

d) 2 octets *CRC* (16 bits) plus *PA ED* (8 bits);

e) RS-485 physical layer adds 3 bits (start, parity and stop bits) to the 8 bits of the DLL char;

f) MBP synchronous physical layer uses just 8 bits per DLL char;

g) The offset (that represents the total number of bits since the beginning of the PhL PDU until the length of the data field is known (o)), is different for the two physical mediums. In the case of PROFIBUS DP (RS-485), it must be taken into account that the "length of data" information is found inside the DLL PDU either in an implicit or explicit way, depending on the type of DLL PDU. In all types of PDUs but the "variable length" PDU type, the length is implicit to the Start Delimiter (SD) field, since there is a unique identifier (SD1-4) for each PDU type (e.g. SD3 corresponds to a "Fixed length frame with data field" – refer to Figure 3). Since the Start Delimiter field is always the first field of the DLL PDU, the offset (o) for this type of PDU is always equal to 11 bits (1 UART char). Nevertheless, for the case of the variable data field type of PDU (with Start Delimiter SD2), the length of the DLL PDU is explicitly present at the beginning of the DLL PDU (LE,LEr). Therefore, the offset (o) would depend on the type of PROFIBUS DLL PDU being considered. For the sake of simplicity, it has been considered that the offset for DP RS-485 media is always equal to 33 bits ($o^{DP} = 33$), i.e. the length of the DLL PDU is only known at the end of the third character (after the LEr field).

h) For the PA MBP physical medium, the same reasoning as in f) (length known after the third character) is assumed, but the 2 octets preamble must also be considered. Therefore, 2 octets (16 bits) preamble plus 3 chars (8 bits/char = 24 bits), results in a total of 40 bits.

For the example network, let us assume the message streams outlined in Table 2. A message stream is a temporal sequence of message transactions concerning, for instance, the remote reading of a process variable.

Table 2: Message Streams parameters (periodicity omitted)

Message Stream	Initiator	Responder	L_{req} (chars)*	L_{resp} (chars)*
MS^1	M	S1	8	8
MS^2	M	S1	57	57
MS^3	M	S1	107	107
MS^4	M	S1	253	253
MS^5	M	S2	8	8
MS^6	M	S2	57	57
MS^7	M	S2	107	107
MS^8	M	S2	253	253

* To be suitable for both DP RS-485 and PA MBP physical layers, the length of the request (L_{req}) and response (L_{resp}) frames does not include the 2 chars of the *FCS* and *ED* fields.

7.3. *Duration of DP RS-485 and PA MBP physical layer frames*

The duration of the DP and PA physical layer frames can be computed using Eq. (6), as described next. The duration of DP RS-485 PhL frames can be computed as follows:

$$C^{DP} = \frac{0 + L \times (8+3) + 22}{93.75 \times 10^3} = \frac{11 \times L + 22}{93.75} \quad (ms)$$

and for the particular case of the token frame:

$$C^{DP}_{TOKEN} = \frac{3 \times 11}{93.75 \times 10^3} \approx 0.352\, ms$$

For PA MBP physical media, the duration can be computed as:

$$C^{PA} = \frac{16 + L \times 8 + 24}{31.25 \times 10^3} = \frac{8 \times L + 40}{31.25} \quad (ms)$$

Table 3 presents the PhL PDU duration for several PROFIBUS PDU lengths.

Table 3: Physical layer frame duration for DP RS-485 and PA MBP

Frame Type	L (chars)*	C^{DP} (ms)	C^{PA} (ms)
Token	3	0.35	2.05
Fixed length no data	4	0.70	2.30
1 DLL data octet	8	1.17	3.33
50 DLL data octets	57	6.92	15.87
100 DLL data octets	107	12.79	28.67
150 DLL data octets	157	18.66	41.47
246 DLL data octets	253	29.92	66.05

* Again, note that the frame length (L) does not include the 2 chars corresponding to the DP RS-485 *FCS* and *ED* fields (the token frame does not include those two fields).

From the table, the fact that the bit rate in PA MBP physical media is 3 times smaller than in DP RS-485, results in that frames have a longer duration in the PA segment. For instance the

token frame is roughly 6 times longer in PA (2 ms against 0.35 ms), and a maximum length frame (L = 253) takes more than two times to transmit in PA (66 ms against 30 ms).

As it can easily be figured out from these results, and at the light of the reasoning presented in the previous sections, this fact has a strong impact in the timing behaviour of the network. Namely, in order to have real-time guarantees, there is the need to compute and set the appropriate values for the Idle Time and Slot Time parameters, which is presented next.

7.4. Idle Time parameters

The idle time parameters were computed using a software tool implementing the algorithm presented in [14]. For this purpose, it was assumed that the repeater (coupler) has an internal relaying delay (t_{rd}) of 25 μs and that the minimum idle time (T_{IDm}) is equal to 100 bit times. The turnaround (reaction) time (PROFIBUS T_{SDR} parameter - station delay of the responders) of the responders (S1 and S2) is assumed to be in the range of 10-50 μs (t^{min}_{rt} = 10 μs (min T_{SDR}); t^{max}_{rt} = 50 μs (max T_{SDR})).

Table 4 summarises the idle time values for this case study.

Table 4: Idle Time parameter values – DP at 93.75 kbit/s

Node	t_{ID1+} (ms)	T_{ID1} (bits)	t_{ID2+} (ms)	T_{ID2} (bits)
DP Master	77.58	7374	38.26	3687
(PA Master)	0	100	0	100
Repeater	-	100	-	100

From the table, DP master (M1) must introduce an additional idle time of around 80 ms after receiving a response/token frame, and of around 40 ms after transmitting an unacknowledged request frame. In the hypothetical case of the network including a PA master, this would not have to introduce additional idle times (only the default 100 bit times), since the bit rate in the PA side is significantly lower (there would be no increased queuing latencies in the repeater). It is assumed that the repeater introduces the default (minimum) idle time value (100 bit times) between any consecutive (relayed) frames.

7.5. Transactions duration and Slot Time parameter

Table 5 summarises the worst-case system turnaround time (t_{st}) and duration of message transactions (C_{ack}) obtained by applying the methodologies outlined in the previous sections to the case study. All parameters were computed using a software tool implementing the algorithm described in Annex B of [7].

The *Path* column describes the Physical Media in the path from initiator to responder. For instance, the *path* for message stream 1 (MS^1) is $\{DP\}$, since the request PDU is issued from M1 in a DP segment and arrives to S1, in the same segment. The *path* for message stream 5 (MS^5) is $\{DP,PA\}$, since the request PDU is issued from M1 in a DP segment, is relayed by the repeater and arrives to S2, in the PA segment.

In the case where initiator and responder belong to the same segment (without any repeater in the path between them), the worst-case system turnaround time is equal to the maximum responders' turnaround time, i.e. (t_{st} = max T_{SDR} = t^{max}_{rt} = 50 μs. This is the case of Message Streams 1-4.

Table 5: System turnaround times and duration of transactions

Msg. Stream	Init.	Resp.	L_{req} (chars)*	L_{resp} (chars)*	Path	t_{st} (ms)	C_{ack} (ms)	t_{st} (ms)	C_{ack} (ms)
MS[1]	M	S1	8	8	DP	0.05	81.05	0.05	19.96
MS[2]	M	S1	57	57	DP	0.05	92.55	0.05	43.68
MS[3]	M	S1	107	107	DP	0.05	104.29	0.05	67.88
MS[4]	M	S1	253	253	DP	0.05	138.55	0.05	138.55
MS[5]	M	S2	8	8	DP,PA	41.11	85.11	3.01	22.93
MS[6]	M	S2	57	57	DP,PA	17.70	110.20	3.70	47.33
MS[7]	M	S2	107	107	DP,PA	31.57	135.80	4.88	72.72
MS[8]	M	S2	253	253	DP,PA	72.06	210.55	8.96	147.47
						DP at 93.75 kbit/s		DP at 45.45 kbit/s	

Concerning the duration of these message transactions (C), they may be computed using Eq. (12). As an example, message transactions corresponding to message stream 1 have the following worst-case duration:

$$C_{ack} = 1173.(3) + 50 + 1173.(3) + \frac{7374}{93.75} \times 10^3 \approx 81053\,\mu s \approx 81.05\,ms$$

The first component of the Slot Time parameter – t_{SL1} – should be greater than the maximum between the worst-case system turnaround time of all message transactions in the network, i.e. t_{SL1} = 72.06 ms (underlined value in Table 5). However, since the worst-case turnaround time after transmitting the token is t_{SL2} = 78.66 ms (computation details not presented here), the Slot Time parameter should be set to the maximum between these two parameters, i.e. t_{SL} = 78.66 ms. Considering M1 in DP running at 93.75 kbit/s, the Slot Time parameter should be set to T_{SL} = 78.66 x 93.75 = 7374 bit times.

If the bit rate on the DP side is reduced in order to be closer to the bit rate on the PA side, the network will be more balanced. Therefore, the additional idle times inserted by the DP master will be smaller (over 10 times):

Table 6: Idle Time parameter values – DP at 45.45 kbit/s

Node	t_{ID1+} (ms)	T_{ID1} (bits)	t_{ID2+} (ms)	T_{ID2} (bits)
DP Master	12.85	685	5.33	343
(PA Master)	0	100	0	100
repeater	0	100	0	100

In spite of the frame duration on the DP side increasing, most worst-case system turnaround times and all worst-case transaction durations are reduced, as may be also seen in Table 5.

Worst-case system turnaround times for transactions between M and S2 (repeater in the path) are reduced, since as the request frame takes longer to transmit (lower bit rate) on the DP side, the time elapsed until starting receiving the response (t_{st}) is smaller. This fact, together with the reduction in Idle Time (T_{ID1}), results in smaller worst-case transactions duration, leading to higher network throughput.

If the DP bit rate is further reduced to 19.2 kbit/s, then the DP master would not have to introduce any additional idle times (only a hypothetical PA master would) and the system turnaround times would be much smaller (and consequently also the Slot Time parameter). Nevertheless, transactions duration would increase significantly.

As a conclusion, from the set of standard bit rates defined by the PROFIBUS standard for DP RS-485 networks, we suggest to adopt the 45.45 kbit/s bit rate (used in Siemens DP/PA coupler systems [19]), provided that the network parameters are set as proposed in this paper. In this way, the Idle Times, the worst-case system turnaround times and the Slot Time parameter are reduced, increasing the responsiveness of the network to communication failures (when an error occurs, retransmissions are issued sooner).

8. Conclusion

An increasing number of industrial automation systems require interoperability between different communication networks. Many factors foster this heterogeneity, such as the interoperability between lower and higher level networks (e.g. Fieldbus to Ethernet/Internet) or between different fieldbus networks.

With over 14 million nodes worldwide [4], PROFIBUS is a leading fieldbus technology, covering a wide range of factory and process automation applications. Therefore, there is a trend towards total interoperability in hybrid PROFIBUS networks, such as PROFIBUS-DP/PA and wired/wireless networks. In this context, going beyond the standard single segment PROFIBUS network brings up complex issues concerning the interoperability between the different segments of the network.

Assuming that these segments only differ at the physical layer level (bit rate, frame format), one possible solution is to achieve interconnectivity through the use of repeaters. This results in a broadcast network, where every node listens to every transmitted message. However, queuing delay problems in the repeaters (due to different bit rates and frame formats) introduce additional and unpredictable queuing latencies in message transactions. This fact turn message response times unpredictable, which is not admissible in a real-time system.

In this paper, we outlined a mechanism for media adaptation based on the insertion of additional idle time between consecutive messages. We have also presented the major guidelines of a methodology for engineering this type of heterogeneous PROFIBUS networks, by a pre-run-time computation and setting of a number of standard PROFIBUS parameters. It should be highlighted that this methodology can be applied to any heterogeneous PROFIBUS network where repeaters are used to interconnect different physical layers. Namely, this methodology has already been successfully applied to the case of hybrid wired/wireless PROFIBUS networks [13,14], namely in the context of the RFieldbus European Project (IST) [21,22], where two pilot field-tests were carried out to validate and demonstrate the proposed architecture and technologies (prototype cut-through wired/wireless repeaters were used) [23,24].

Finally, we illustrate the application of the presented methodologies with an example scenario involving a heterogeneous PROFIBUS-DP/PA network. For this case, we have computed the most relevant parameters (e.g. Idle Times, Slot Time, worst-case message duration) for commissioning such a network, in a way that bounded and predictable message response times are guaranteed. As a general practical conclusion, the bit rates of the PROFIBUS DP and PA segments should be as much similar as possible, since the additional idle times to be inserted and the system turnaround times would be much smaller (and consequently also the Slot Time parameter), increasing the responsiveness of the network to communication failures.

References

[1] EN50170, “General Purpose Field Communication System”, Volume 1 – P-NET, Volume 2 – PROFIBUS, Volume 3 – WorldFIP, Volume 4 – Foundation Fieldbus, European Norm, 1996-2002.

[2] ODVA, “Ethernet/IP Specification”, Release 1.0, ControlNet International and Open DeviceNet Vendor Association, June 2005. Available online at http://www.odva.org

[3] Rockwell Automation, "Making Sense of e-Manufacturing: a Roadmap for Manufacturers", Rockwell Automation Inc., Cleveland, Ohio, Industry White Paper, 2000.

[4] PROFIBUS official WWW site: http://www.profibus.com

[5] Tovar, E. and Vasques, F., “Real-Time Fieldbus Communications Using Profibus Networks”, IEEE Transactions on Industrial Electronics, Vol. 46, No. 6, pp. 1241-1251, December 1999.

[6] Tovar, E. and Vasques, F., “Cycle Time Properties of the PROFIBUS Timed Token Protocol”, Computer Communications, Vol. 22, No. 13, pp. 1206-1216, Elsevier Science, August 1999.

[7] Tovar E., “Supporting Real-Time Communications with Standard Factory-Floor Networks”, PhD Thesis, University of Porto, 1999.

[8] Monforte, S., Alves, M., Vasques, F., Tovar, E., "Designing Real-Time Systems Based on Mono Master Profibus-DP Networks", in Proc. of the 16th IFAC Workshop on Distributed Computer Control Systems (DCCS'2000), Sydney, Australia, November 29-December 1, 2000, pp. 36-43.

[9] Cavalieri S., Monforte S., Tovar E., Vasques F.. “Multi-Master Profibus-DP Modelling and Worst-Case Analysis Based Evaluation”, in Proc. of the 15th IFAC World Congress on Automatic Control, Barcelona, Spain, 2002.

[10] Pacheco F., Tovar E., Kalogeras A., Pereira N. (2001). Supporting Internet Protocols in Master-Slave Fieldbus Networks. Proceedings of the 4th IFAC International Conference on Fieldbus Systems and Their Applications (FET'2001), Nancy, France, pp. 260-266.

[11] Pratl G., Lobachov M., Sauter T., “Highly Modular Gateway Architecture for Fieldbus/Internet Connections”, Proc of the 4th IFAC International Conference on Fieldbus Systems and their Applications (FeT’2001), Nancy, France, pp.267-273.

[12] Sveda M., Zezulka F. (1997). Interconnecting Low-Level Fieldbuses. Proc of the 23rd Euromicro Conference, Budapest, Hungary, pp. 614-620.

[13] Alves, M., Tovar, E., Vasques, F., Hammer, G., Roether, K., "Real-Time Communications over Hybrid Wired/Wireless PROFIBUS-based Networks", in Proc. of the 14th Euromicro Conference on Real-Time Systems - ECRTS'02, Vienna, Austria, pp. 142-150. June 2002.

[14] Alves, M., “Real-Time Communications over Hybrid Wired/Wireless PROFIBUS-Based Networks”, PhD Thesis, University of Porto, 2003.

[15] L. Ferreira, "A Multiple Logical Ring Approach to Real-time Wireless enabled PROFIBUS Networks", PhD. Porto, Portugal: University of Porto, 2005.

[16] Decotignie J.-D., “Interconnection of Wireline and Wireless Fieldbusses”, in “The Industrial Information Technology Handbook”, CRC Press, Industrial Electronics Series, 2005, ISBN 0-8493-1985-4.

[17] Monforte S., Alves M., Vasques F., Tovar E., “Segmentation Aspects in Profibus”, IPP-HURRAY Technical Report HURRAY-TR-0019, May 2000.

[18] ISO 7498, “Information Processing Systems – Open Systems Interconnection – Basic Reference Model”, 1984.

[19] Siemens AG, “SIMATIC DP/PA Bus Coupler Manual”, Edition 4, 2000.

[20] Pepperl+Fuchs, “Segment Coupler SK1 and SK2 – Instruction Manual”, 2002..

[21] Rauchhaupt, L., “System and Device Architecture of a Radio Based Fieldbus – The RFieldbus System”, in Proc. of the 4th International Workshop on Factory Communication Systems (WFCS’02), Mälardalen University, Västerås, Sweden, pp. 185-192, 2002.

[22] http://www.hurray.isep.ipp.pt/rfieldbus

[23] http://www.hurray.isep.ipp.pt/rfpilot

[24] Behaeghel, S., Nieuwenhuyse, K., Marques, L., Alves, M., Tovar, E., "Engineering Hybrid Wired/Wireless Fieldbus Networks - a case study", in Proc. of the 2nd International Workshop on Real-Time LANs in the Internet Age (RTLIA’03), Porto, Portugal, July 2003, pp. 111-114, 2003.

Short Bio

Mário Alves was born in 1968 and has a Degree (1991), a MSc (1995) and a PhD (2003) in Electrical and Computer Engineering at the University of Porto. Since 1995, he is with the School of Engineering of the Polytechnic Institute of Porto (ISEP/IPP). He has over 12 years experience on distributed computer controlled systems (DCCS) and industrial communication systems, namely on Fieldbus, Industrial Ethernet and Wireless Sensor networks. He has authored and co-authored more than 15 scientific and technical papers, participated in the revision of papers and in the program committee of some of the most reputed conferences in the above mentioned areas. His PhD work addressed the analysis and design of a real-time hybrid (wired/wireless) PROFIBUS-based network, in the framework of the RFieldbus IST Project. Current research efforts focus on the design of real-time communications architectures for Wireless Sensor Networks.

Eduardo Tovar was born in 1967 and received the Licentiate, MSc and PhD degrees in Electrical and Computer Engineering from the University of Porto, Porto, Portugal, in 1990, 1995 and 1999, respectively. Currently he his Professor of Industrial Computer Engineering in the Computer Engineering Department at the Polytechnic Institute of Porto (ISEP-IPP), where he is also engaged in research on real-time distributed systems and factory communications. He leads the CISTER Research Unit (UI 608), a top rated (“Excellent”) unit of the FCT Portuguese network of research units. Since 1991 he authored or co-authored more than 60 scientific and technical papers in the area of real-time systems and industrial computer engineering. He served as General Chair and Program Chair in reputed scientific events such as the IEEE Workshop on Factory Communication Systems (WFCS) or the Euromicro Conference on Real-Time Systems (ECRTS).