

Analysis of the Worst-Case Real Token Rotation Time in PROFIBUS Networks¹

Eduardo Tovar[‡], Francisco Vasques[†]

[‡] Polytechnic Institute of Porto (ISEP-IPP), Portugal, e-mail: emt@dei.isep.ipp.pt

[†] University of Porto (FEUP), Portugal, e-mail: vasques@fe.up.pt

Abstract. This paper provides a comprehensive study on how to support time critical distributed applications using PROFIBUS. We show that, despite the absence of synchronous bandwidth allocation, it is possible to guarantee real-time behaviour for the high-priority traffic in PROFIBUS networks. The main contribution of this paper is to give a methodology for the setting of the T_{TR} parameter, by proper analysis of the worst-case real token rotation time (T_{RR}).

1 Introduction

Communication networks aimed at the interconnection of sensors, actuators and controllers are commonly known as fieldbus networks. In the past, fieldbus scope was dominated by vendor specific solutions, which were mostly restricted to specific application areas. Moreover, the concepts behind each proposed network were highly dependent on the manufacturer of the automation system, each one with different technical implementations and also claiming to fulfil different application requirements, or the same requirements with different technical solutions [1].

In the recent years, more and more standardised fieldbuses are accepted supporting the open system communication concept and thus having a vendor independent communication. Controller Area Network (CAN) [2] and PROcess Field BUS (PROFIBUS) [3] are two of the most popular fieldbuses, which are defined as international standards, respectively by ISO and CENELEC. Though these two protocols were originally conceived for different purposes, they have significantly evolved so that they can be used in distributed computer controlled systems connecting remote sensors and actuators. CAN, in fact, was primarily conceived for automotive applications, to solve cabling problems found in some kind of vehicles. However, due to its interesting features, it is also being considered in the automated manufacturing and process control environments [4]. PROFIBUS and its communication stack, instead, were created as a communication support in automated factory environments. Its great flexibility however was achieved at the cost of a reduced network responsiveness, which means that PROFIBUS is not suited for high-speed real-time data exchanges. To overcome this problem, in the past few years a new upper layer protocol have been developed for PROFIBUS, called decentralised periphery (DP) [5]. Such development makes PROFIBUS particularly suited for use as a control network

¹ This work was partially supported by ISEP, FLAD, DEMEGI/FEUP and FCT.

in those applications where a number of small-sized data variables have to be exchanged at a high rate. Considering the general characteristics of CAN and PROFIBUS, several studies have shown that CAN is more suited for small-size event-driven distributed systems while PROFIBUS gives better performances when used in medium-to-large systems, whose interactions are mostly based on polling schemes [6].

The remainder of the paper is organised as follows: in section 2 we briefly describe the PROFIBUS MAC mechanisms. In section 3, we introduce some basic concepts concerning the real-time characteristics of the PROFIBUS protocol, which were previously presented in [7,8]. In section 4, we give an accurate bound for the worst-case token rotation time based on the evaluation maximum token lateness in each master station. Such worst-case token rotation time is the basis for the setting of the T_{TR} parameter, which is the main contribution of this paper. Finally, in section 5 we draw some conclusions.

2 The PROFIBUS Implementation of the Timed Token Protocol

The PROFIBUS MAC mechanism is based on a token passing procedure, used by master stations to grant the bus access to each one of them, and a master-slave procedure used by master stations to communicate with slave stations. The PROFIBUS token passing procedure uses a simplified version of the timed token protocol [9]. One of the PROFIBUS MAC main functions is the control of the token cycle time, which will now be briefly explained.

After receiving the token, the measurement of the token rotation time begins. This measurement expires at the next token arrival and results in the real token rotation time (T_{RR}). A target token rotation time (T_{TR}) must be defined in a PROFIBUS network. The value of this parameter is common to all masters, and is used as follows. When a station receives the token, the token holding time (T_{TH}) timer is given the value corresponding to the difference, if positive, between T_{TR} and T_{RR} . PROFIBUS defines two categories of messages: high priority and low priority. These two categories of messages use two independent outgoing queues. If at the arrival, the token is late, that is, the real token rotation time (T_{RR}) is greater than the target rotation time (T_{TR}), the master station may execute, at most, one high priority message cycle. Otherwise, the master station may execute high priority message cycles while $T_{TH} > 0$. T_{TH} is always tested at the beginning of the message cycle execution. This means that once a message cycle is started it is always completed, including any required retries, even if T_{TH} expires during the execution. We denote this occurrence as a T_{TH} overrun. The low priority message cycles are executed if there are no high priority messages pending, and while $T_{TH} > 0$ (also evaluated at the start of the message cycle execution, thus leading to a possible T_{TH} overrun).

3 Timing Requirements in PROFIBUS Networks

As previously described, and as far as there are high priority message transfers pending, a master station is guaranteed to transmit, at least, one high priority message

per token arrival (no matter if there is enough token holding time left). We define the upper bound between two consecutive token arrivals to a particular master station k as T_{cycle}^k . Based on the knowledge of the upper bound for the token cycle time, we will derive a methodology for setting the T_{TR} parameter in order to guarantee the real-time requirements of PROFIBUS messages.

3.1 Network and Message Models

We consider a bus topology containing n master stations. A special frame (the token) circulates around the logical ring formed by the masters. We denote the logical ring latency (token walk time, including node latency delay, media propagation delay, etc.) as τ . Message cycles generated at run-time in the system are either high priority or low priority messages. In each master k we consider nh^k high priority message streams. A message stream corresponds to a temporal sequence of message cycles related, for instance, with the reading of a process sensor. We denote the i^{th} ($i = 1, 2, \dots, nh^k$) high priority stream associated to a master k as Sh_i^k . A high priority message cycle stream Sh_i^k is characterised as:

$$Sh_i^k = (Ch_i^k, Dh_i^k) \quad (1)$$

Ch_i^k is the maximum amount of time required to perform a high priority message cycle of stream i belonging to a master k . Dh_i^k is the message cycle relative deadline, which is the maximum amount of time that may elapse between the transmission request and the reception of the related response, both at the application process level. We consider that, in the worst-case, the deadline can be seen as the minimum inter-arrival time (MIT) between two consecutive message requests in the same stream.

As we do not intend to guarantee deadlines for the low priority traffic, a low priority message stream Sl_i^k is merely characterised as follows:

$$Sl_i^k = (Cl_i^k) \quad (2)$$

where Cl_i^k is the maximum amount of time required to perform a message cycle of a low priority stream i belonging to a master k .

3.2 Queuing Delay for a PROFIBUS Message Cycle

Contrarily to other timed token based networks, in PROFIBUS we can not use analysis similar to those proposed in [10,11], since in PROFIBUS protocol there is no synchronous bandwidth² to be allocated in each master. However, it is possible to guarantee a real-time behaviour for the high priority traffic using the PROFIBUS protocol. In [7,8] the authors analyse and propose two different approaches. One of the approaches considers a worst-case scenario with only one high priority message cycle processed per token visit. As already mentioned, in PROFIBUS it is guaranteed that

² In PROFIBUS there is no synchronous bandwidth allocation (H_i). If a station receives a late token (T_{RR} greater than T_{TR}) only one high priority message may be transmitted. Contrarily, in the timed token protocol the station can transmit real-time (high priority) traffic during H_i , even if the token is late.

even if a master receives a late token, it will still be able to execute one high priority message cycle. Then, if there are m messages pending in the outgoing queue, in the worst-case it will take m token visits to execute all those high priority messages.

It is obvious that the queuing delay (Q^k) very much depends on how the outgoing high priority queue is implemented. In PROFIBUS the majority of the implementations use a First-Come-First-Served (FCFS) queue. In this paper we assume this type of queue implementation. For the queuing delay analysis, it is important to note that, in master k , the maximum number of pending messages will be nh^k , corresponding to one message per each Sh_i^k stream. This assumption results from the fact that if there are two messages of the same stream pending in the high-priority outgoing queue, then a deadline for that message stream was missed.

It is now clear that, if we assume that message deadlines are not missed (thus the maximum number of high priority pending messages is nh^k), the upper bound for the message queuing delay in a master k is:

$$Q^k = nh^k \times T_{cycle}^k \quad (3)$$

3.3 Pre-Run-Time Schedulability Condition

The worst-case end-to-end communication delay (E) can be expressed as follows:

$$E = g + Q + C + d \quad (4)$$

In (4), g represents the worst-case generation delay for the master application task to generate and queue that specific read request. The term Q corresponds to the worst-case delay for that request to gain access to the communications device after being queued. The term C corresponds not only to the worst-case for transmitting the request, but also to the time needed to receive the response from the slave (including processing at the slave side, slave turnaround time, propagation delay and other network latencies). Finally, d represents the delivery delay, that is, the time needed to process the response before finally delivering it to the destination task, which in the case of PROFIBUS is in the same host processor as the sending task.

It is now possible to re-write the end-to-end communication delay for a high priority PROFIBUS message stream as follows:

$$E_{Sh_i^k} = g_{Sh_i^k} + nh^k \times T_{cycle}^k + Ch_i^k + d_{Sh_i^k} \quad (5)$$

Thus, a set of PROFIBUS high priority message streams guaranteedly satisfies its deadlines if the following pre-run-time condition is verified:

$$Dh_i^k \geq E_{Sh_i^k}, \quad \forall_{\text{master } k, \text{ stream } Sh_i^k} \quad (6)$$

that is, if all high priority message stream deadlines are greater than or equal to their worst-case end-to-end communication delay (as defined in (5)). We need now to evaluate T_{cycle}^k . As the authors shown in [12], T_{cycle}^k is a function of T_{TR} . Thus, having

defined T_{cycle}^k , it will then be possible to set the T_{TR} parameter such as the pre-run-time schedulability condition (6) is verified, otherwise the high priority traffic is not schedulable, in the sense that deadlines can be missed.

4 Analysis of the Worst-Case Real Token Rotation Time

4.1 Evaluation of the Token Cycle Time

In PROFIBUS, the real token rotation time (T_{RR}^k) will always be smaller than T_{TR} , except when one or more masters in the logical ring induce the token to be late. Two reasons may justify a late token at a master k .

1. As once a message cycle is started, it is always completed, even if T_{TH}^k has expired during its execution, a late token may be delivered to the following stations. We define this occurrence as an *overrun* of the T_{TH}^k .
2. If a master receives a late token, it will still be able to send one high priority message. This may further increase the token lateness.

Based on these two conditions and on the following three theorems (the interested reader is referred to [12] for full treatment),

- *Theorem 1* - In PROFIBUS networks, if the master holding the token releases it before the T_{TH}^k expiration, then, the following master in the logical ring will receive an early token.
- *Theorem 2* - In a PROFIBUS network, in a specific token cycle, only one overrun will contribute to the token lateness.
- *Theorem 3* - If the token holding time in a PROFIBUS master k is greater than $T_{TR} - \tau$, all the following masters up to master $k - 1$ (modulo n) will receive a late token.

T_{cycle}^k is defined as follows:

$$T_{cycle}^k = T_{TR} + T_{del}^k \quad (7)$$

where T_{del}^k is the worst-case token lateness.

Such worst-case token lateness may result not only from an overrun in the master k but also from one occurring in one of the following masters ($k + 1$ until $k - 1$) (modulo n). Using fig. 1 as an illustrative example, assume that master 1 do not overruns its T_{TH}^1 (since the last message to be transmitted starts before the end of T_{TH}^1). Then, master 2 may use its available token holding time and produce an overrun.

If this overrun is longer than the sum of the maximum overrun in station 1 with the longest high priority message cycle of station 2, then this would led to a higher value for T_{del}^1 (fig. 2).

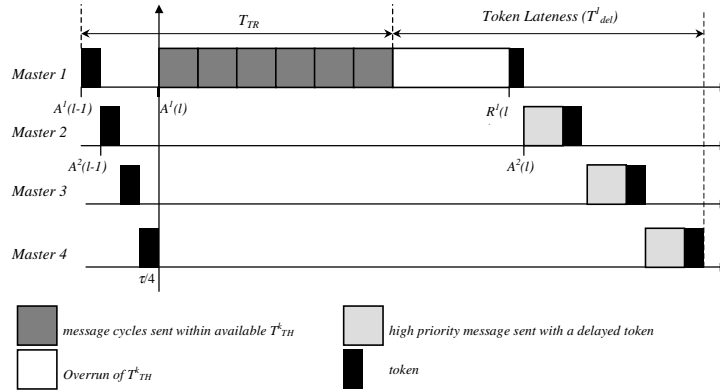


Fig. 1. Illustrative example for theorem 3

Similarly, if the maximum overrun in master 3 is longer than the sum of both an overrun in station 2 and the longest high priority message cycle of station 3, then this would lead to a higher value for T_{del}^l . Note that by theorems 2 and 3, for the token lateness evaluation in master k , we should only consider one overrun in station j (j ranging from k to $k - 1$ (modulo n)) and one high priority message cycle per each station whose address is between j and station $k - 1$ (modulo n).

Basically, we can conclude that the T_{cycle}^k will depend on which master produce the worst-case overrun and on its relative positioning in the logical ring.

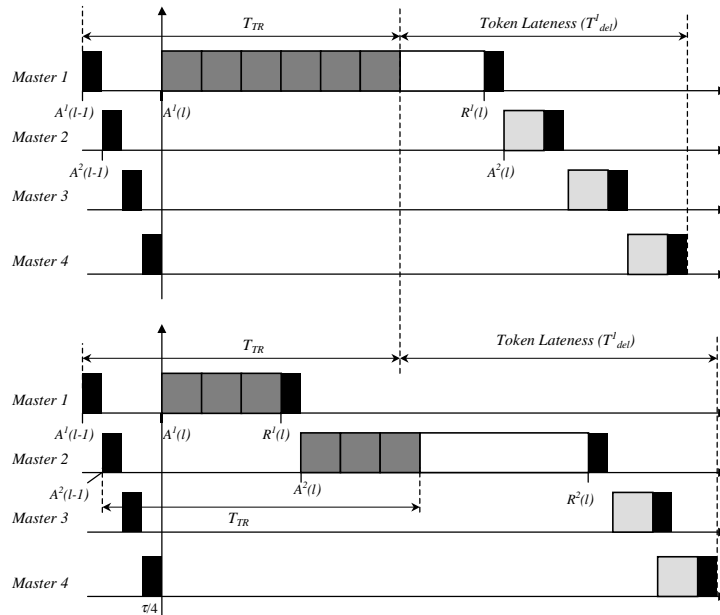


Fig. 2. Comparison between two overrun situations

For the token lateness (T_{del}^k) evaluation, we introduce the following master parameters: Ω^k , Φ^k and Ψ^k . Ω^k is the longest high priority message cycle transfer requested by a station k :

$$\Omega^k = \max_{i=1, \dots, nh^k} \{Ch_i^k\} \quad (8)$$

Φ^k is the longest low priority message cycle transfer requested by a station k :

$$\Phi^k = \max_{i=1, \dots, nh^k} \{Cl_i^k\} \quad (9)$$

Finally, Ψ^k is the longest message cycle transfer requested by a station k (including both types of message cycles):

$$\Psi^k = \max\{\Omega^k, \Phi^k\} \quad (10)$$

Using the outlined analysis, we can thus define the maximum token lateness in a PROFIBUS master station k (T_{del}^k) as:

$$T_{del}^k = \max_{j \in \phi_1} \left\{ \Psi^j + \sum_{i \in \phi_2} \Omega^i \right\} \quad (11)$$

where ϕ_j is defined as

$$\phi_1 = \begin{cases} k..n, & \text{if } k=1 \\ k..n, 1..k-1, & \text{if } k > 1 \end{cases} \quad (12)$$

and ϕ_2 as

$$\phi_2 = \begin{cases} j+1..n, & \text{if } k=1 \\ j+1..n, 1..k-1, & \text{if } k > 1 \end{cases} \quad (13)$$

4.2 Setting the T_{TR} Parameter

Using the Tcycle definition (7), the pre-run-time schedulability condition (6) can be rewritten as follows:

$$Dh_i^k \geq g_{Sh_i^k} + nh^k \times (T_{TR} + T_{del}^k) + Ch_i^k + d_{Sh_i^k}, \forall_{\text{master } k, \text{ stream } Sh_i^k} \quad (14)$$

Therefore, we may have an expression for setting the T_{TR} parameter:

$$0 \leq T_{TR} \leq \frac{Dh_i^k - Ch_i^k - \delta_{Sh_i^k}}{nh^k} - T_{del}^k, \forall_{\text{master } k, \text{ stream } Sh_i^k} \quad (15)$$

where δ aggregates both the generation and delivery delays of message stream.

5 Conclusions and Discussion

In this paper we have provided a comprehensive study on how to use PROFIBUS field bus networks to support real-time communication. The major contribution of this paper is to provide a methodology for the setting of the Target Token Rotation Time, in order to guarantee communication real-time behaviour using the PROFIBUS protocol.

One possibility to reduce the worst-case response time relies on the implementation of a priority-based queue at the application process level. The PROFIBUS protocol uses First-Come-First-Served (FCFS) outgoing queues at the communication stack. Thus, as future work, we are to evaluate the use of a priority-based queue for the high-priority message requests at the application process level, limiting the stack outgoing communication queue to one pending request.

References

1. Cardoso, A., Tovar, E.: "Industrial Communication Networks: Issues on Heterogeneity and Internetworking", Proceedings of the 6th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM'96, May 1996, Atlanta, USA, pp. 139-148.
2. ISO 11898, "Road Vehicle - Interchange of Digital Information - Controller Area Network (CAN) for High-Speed Communication", 1st Edition, ISO, 1993.
3. EN 50170, "General Purpose Field Communication System", European Standard, CENELEC, 1996, Vol. 2/3.
4. Zuberi, K., Shin, K.G.: "Real-Time Decentralised Control with CAN", Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation, November 1996, pp. 93-99.
5. DIN 19245, "PROFIBUS-DP - Process Field Bus Decentralised Periphery (DP) - Part 3", Draft Standard DIN 19245, issue 1994.
6. Cena, G., Demartini, C., Valenzano, A.: "On the Performances of two Popular Fieldbuses", Proceedings of the 2nd IEEE International Workshop on Factory Communication Systems, WFCS'97, October 1997, Barcelona, Spain, pp. 177-186.
7. Tovar, E., Vasques, F.: "Guaranteeing Real-Time Message Deadlines in Profibus Networks", Proceedings of the 10th Euromicro Workshop on Real-time Systems, Berlin, Germany, IEEE Press, 1998, pp. 79-86.
8. Tovar E., Vasques, F.: "Setting Target Rotation Time in Profibus Based Real-Time Applications", Proceedings of the 15th IFAC Workshop on Distributed Computer Control Systems (DDCS'98), Como, Italy, 1998, pp. 1-6.
9. Grow, R.: "A Timed Token Protocol for Local Area Networks", Proceedings of Electro'82, May 1982, Token Access Protocols, Paper 17/3..
10. Montuschi, P., Ciminiera, L., Valenzano, A.: "Time Characteristics of IEEE802.4 Token Bus Protocol", IEE Proceedings, January 1992, 139 (1), pp. 81-87.
11. Malcolm, N., Zhao, W.: "Guaranteeing Synchronous Messages with Arbitrary Deadline Constraints in a FDDI Network", Technical Report, Department of Computer Science, Texas A&M University, March 1993.
12. Tovar E., Vasques, F.: "Cycle Time Properties of the PROFIBUS Timed Token Protocol", September 1998, to appear in Computer Communications, Elsevier Science.