Guaranteeing Real-Time Message Deadlines in PROFIBUS Networks*

Eduardo TOVAR
Department of Computer Science
School of Engineering
Polytechnic Institute of Porto,
Rua de São Tomé, 4200 Porto, Portugal
e-mail: emt@dei.isep.ipp.pt

Francisco VASQUES
Department of Mechanical Engineering
School of Engineering
University of Porto,
Rua dos Bragas, 4099 Porto Codex, Portugal
e-mail: vasques@fe.up.pt

Abstract

This paper provides a comprehensive study on how to use Profibus networks to support real-time communications, that is, ensuring the transmission of the real-time messages before their deadlines.

Profibus is based on a simplified Timed Token (TT) protocol, which is a well-proved solution for real-time communication systems. However, Profibus differences to the TT protocol prevent the application of the usual TT analysis. The main reason is that, conversely to the TT protocol, in the worst case, only one high-priority message is processed per token visit. The major contribution of this paper is to prove that, despite this shortcoming, it is possible to guarantee communication real-time behaviour with the Profibus protocol.

1. INTRODUCTION

A recent trend in distributed process control systems is to interconnect the distributed elements by means of a multi-point broadcast network, instead of using the traditional point-to-point links. As the network bus is shared between a number of network nodes, there is an access contention, which must be solved by the Medium Access Control (MAC) protocol.

Usually, distributed process control systems impose real-time requirements. In essence, by real-time requirements we mean that traffic must be sent and received within a bounded interval, otherwise a timing fault is said to occur. This motivates the use of communication networks within which the MAC protocol is able to schedule messages streams according to its real-time requirements.

A reasonable number of commercial solutions, usually called field bus networks, have been proposed during the past decade. Some distinguished examples are WorldFIP [Cen96], Profibus [Cen96], CAN [Sae92] and P-Net [Cen96]. Concurrently, several international standardisation efforts have been and are still being carried out. One of the most relevant resulted into the European Standard EN 50170 [Cen96], which basically encompasses three well-proven field bus national standards: Profibus, WorldFIP and P-NET.

In this paper we address the Profibus MAC ability to schedule message streams according to its real-time requirements, in order to support real-time distributed applications.

The Profibus MAC is based both on a token passing procedure, which is used by the master stations to communicate with each other, and on a master-slave procedure, which is used by master stations to communicate with the slave stations.

The Profibus token passing procedure uses a simplified timed token protocol [Gro82]. One important parameter to consider in this kind of protocols is the Target Token Rotation Time (T_{TR}), which is set at the network initialisation time and stands for all network nodes. Whenever a station receives the token, it may transmit its high priority messages, for a time period no more than its allocated synchronous capacity (H_i). The low priority messages can then be transmitted, but only if the previous token rotation time was lower than T_{TR}. Therefore, the amount of time that a station may hold the token is dynamically adjusted to the speed of token rotation.

Comparing to the timed token protocol, the main difference of the Profibus token passing consists in the absence of synchronous bandwidth allocation. If a station receives a late token (the token rotation time was greater than the target token rotation time) only one high priority message will be transmitted. As a consequence, in Profibus, low priority traffic may drastically affect the high priority traffic capabilities. In fact, if a station receives an early token (the token rotation time was lower than the target token rotation time), and the low

* This work was partially supported by FLAD under the project SISTER 471/97 and by ISEP under the project REMETER.

priority traffic is not constrained in that station, the subsequent stations may be limited to one high priority message transmission when holding the token.

In this paper we propose a basis for setting the Profibus T_{TR} parameter, such as high priority (real-time) message's deadlines are met, even when low priority (non-real-time) traffic is unconstrained at the application level.

2. INTRODUCING PROFIBUS NETWORKS

2.1. General Characteristics

As previously said, the Profibus MAC includes a token passing procedure, which is used by the master stations to communicate with each other, and a master-slave procedure, which is used by master stations to communicate with the slave stations. Figure 1 illustrates this hybrid-operating mode.

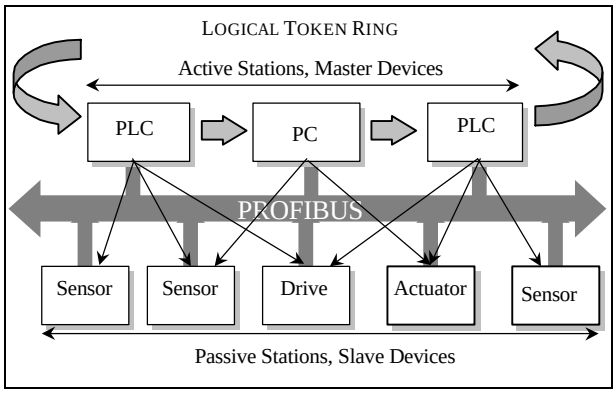

Figure 1

The MAC protocol is implemented at the layer 2 of the OSI reference model, which, in Profibus, is called Field bus Data Link (FDL). In addition to controlling the bus access and the token cycle time, the FDL is also responsible for the provision of data transmission services for the FDL user (e. g. the application layer).

Profibus supports four basic data transmission services: Send Data with No acknowledge (SDN); Send Data with Acknowledge (SDA); Request Data with Reply (RDR) and Send and Request Data (SRD).

The SDN is an unacknowledged service mainly used for broadcasts from an active station to all the other stations on the bus. Conversely, all the other services are based on a real dual relationship between the initiator (master station holding the token) and the responder (passive or active station not holding the token). Another important characteristic of these services is that they must be immediately answered, with either a response or an acknowledge.

In addition to these services, industrial applications often require the use of cyclical transmission methods. A centrally controlled polling method (cyclical polling) is a suitable transmission method, to scan simple field devices, such as sensors or actuators. Profibus enables a polling list to be created in the FDL layer, and can thus carry out a cyclical polling based on the RDR and SDR services.

An important Profibus concept is the *message cycle*. A message cycle consists on a master station's action frame (request or send/request frame) and the associated acknowledgement or response frame. User data may be transmitted in the action frame (send) as well as in the response frame (response).

All the stations, except the token holder (initiator) shall in general monitor all requests. The acknowledgement or response must arrive within a predefined time, the slot time, otherwise the initiator repeats the request. The initiator shall not issue a retry or a new request before the expiration of a waiting period (Idle Time).

All the real-time properties of the Profibus network presented in this paper are based on the knowledge of the message cycle time lengths. This time includes the time needed to issue the action frame and receive the response and also should include possible message retries.

2.2. Behaviour of the Access Control

After receiving the token, the measurement of the token rotation time begins. This measurement expires at the next token receipt and results in the real token rotation time (T_{RR}). T_{RR} is of significance for carrying out non-high priority message cycles. In order to keep the system reaction time, the target rotation time (T_{TR}) of the token must be defined in a Profibus network.

Independently of the real token rotation time, each master station may always execute one high priority message cycle per token receipt. In order to perform non-high priority message cycles, T_{RR} must be lower than T_{TR} at the instant of execution, otherwise the station shall retain the non-high priority message cycles and transmit them at the following token receipts. Once a message cycle is started it is always completed, including any required retries, even if T_{RR} reaches or exceeds the value of T_{TR} during the execution.

When holding the token will successively handle:

- high priority non-cyclical message cycles;
- poll list message cycles;
- low priority non-cyclic message cycles;
- GAP list management (logical ring maintenance).

The standard specifies that low priority, cyclic and GAP management message cycles underlain with specific rules. In our analysis, we only provide a real-time service for high priority messages, and thus all other messages will be considered as low priority traffic.

The following represents the considered Profibus token passing algorithm:

```
/* initialisation procedure */
At each station k, DO:
    T_TH ← 0 ;
    T_RR ← 0 ;
    release T_RR ;

/* run-time procedure */
At each station k, at the Token arrival, DO:
    T_TH ← T_TR - T_RR ;
    T_RR ← 0 ;
    Release T_RR ;              /* count-up timer */
    IF T_TH > 0 THEN
        Release T_TH          /* count-down timer */
    ENDIF;
    IF waiting High priority messages THEN:
        Execute one High priority message cycle
    ENDIF;
    WHILE T_TH > 0 AND pending High priority
      message cycles DO
        Execute High priority message cycles
    ENDWHILE;
    WHILE T_TH > 0 AND pending Low priority
      message cycles DO
        Execute Low priority message cycles
                /* this includes poll list and */
                    /* GAP list message cycles */
    ENDWHILE;
Pass the token to station (k + 1) (modulo k);
```

3. BASIC CONCEPTS OF THE PROFIBUS MAC TIMING ANALYSIS

3.1. Previous Relevant Work

The basic idea of the Timed Token protocol was presented by Grow [Gro82]. In this protocol, messages are distinguished in two types. One concerns the synchronous messages, which are periodic messages that come to the system at regular intervals and have delivery time constraints. The other concerns asynchronous messages, which are non-periodic messages that have no time constraint measured at least in units that are large relative to the token rotation time, as explained in [Sev87].

When a network is initialised, all the stations negotiate a common value for the Target Token Rotation Time (T_{TR}). This is an important protocol parameter, which gives the expected token rotation time. The T_{TR} should be chosen small enough to meet responsiveness requirements of all stations, i.e., fast enough to satisfy the most stringent timing requirements. Each station is assigned a fraction of T_{TR}, known as its synchronous capacity, which is the maximum time each station is allowed to transmit its synchronous messages, if any, every time it receives the token. The asynchronous messages can be transmitted, but only if the token has rotated fast enough, that it is "ahead of schedule" with respect to the target rotation time.

In the Timed Token Protocol, the time between two consecutive token arrivals at a specific station is bounded by 2 x T_{TR} and the average token rotation time is no more than T_{TR}. An intuitive explanation of these two timing properties can be found in [Gro82] and a formal proof in [Sev87].

In order to guarantee high priority message deadlines, the bounded token rotation time is a necessary but not sufficient condition. A node with inadequate synchronous capacity may be unable to guarantee message deadlines, and, on the other hand, allocating excess amount of synchronous capacities to the nodes increases T_{TR}, which may also cause message deadlines to be missed. Therefore, synchronous capacities must be properly allocated to individual nodes. As a consequence, synchronous capacities allocated to the nodes must satisfy two constraints [Agr92, Che92, Mal93]: a protocol constraint and a deadline constraint.

The *protocol constraint* states that the total sum of the allocated synchronous capacities should not be greater that the available portion of T_{TR}, i.e.,

$$\sum_{i=1}^{n} H_i \le TTRT - t \qquad (3.1)$$

Theoretically, the total available time to transmit real-time traffic, during a complete token rotation, can be as much as T_{TR}. However, factors such as ring latency and other protocol or network overheads reduce the total available time to transmit real-time traffic. We denote the portion of T_{TR} unavailable for transmitting synchronous messages by τ.

The *deadline constraint* states that the allocation of the synchronous capacities to the nodes should be such that the synchronous messages are always guaranteed to be transmitted before their deadlines.

A message set can be guaranteed by an allocation scheme once the protocol and the deadline constraints are satisfied. Several allocation schemes have been proposed in [Agr92, Che92, Mal93].

In [Vas94] we can find a first analysis on the message schedulability in Profibus networks. Based on the Timed Token Protocol, these results were later improved and presented in [Vas96]. The work here by described is a step forward in the analysis of message schedulability in Profibus networks.

3.2. Network Model and Message Model

We consider a bus topology containing n master stations. A special frame (the token) circulates around the logical ring formed by the masters (from node k to nodes $k + 1$, … until node n, then to nodes 1, 2, …). We denote the logical ring latency (token walk time, including node latency delay, media propagation delay, etc) as τ.

Message cycles generated in the system at run-time may be classified as either high priority or low priority messages. To each k master node we assume that there are $nh^{(k)}$ high priority message streams and $nl^{(k)}$ low priority message streams. A message stream corresponds to a message cycles sequence related with, for instance, the reading of the value of a process variable.

We denote the i^{th} (i = 1, 2, … $nh^{(k)}$) high priority stream associated to a master node k as $Sh_i^{(k)}$. Similarly low priority streams will be denoted as $Sl_j^{(k)}$ (j = 1, 2, …, $nl^{(k)}$).

A message stream S_i is characterised as $S_i = (C_i, D_i)$. C_i is the maximum amount of time required to transmit a message in a stream. In Profibus this time should also include possible message retries. D_i is the messages relative deadline, which is the maximum amount of time that may elapse between a message arrival and completion of its transmission. We consider that, in the worst case, the deadline can be seen as the minimum inter-arrival time between two consecutive messages in the same stream.

The following notation will then be used:

$$Sh_i^{(k)} = \left(Ch_i^{(k)}, Dh_i^{(k)}\right) \tag{3.2}$$

$$Sl_i^{(k)} = \left(Cl_i^{(k)}\right) \tag{3.3}$$

3.3. Approach Used in the Timing Analysis

Comparing to the timed token protocol, the main difference to Profibus protocol consists of the absence of synchronous bandwidth allocation. The synchronous bandwidth allocation consists of a minimum amount of time that a station holding the token has to transmit high priority messages. This time is set in each station according to their particular high priority messages requirements.

As a consequence, in Profibus, low priority traffic may drastically affect the high priority traffic capabilities. In fact, if a station receives an early token ($T_{TR} - T_{RR}^{(k)} > 0$) and the low priority traffic is not constrained in that station, subsequent stations may have an initial value for $T_{TH} < 0$. Figure 2 illustrates this situation.

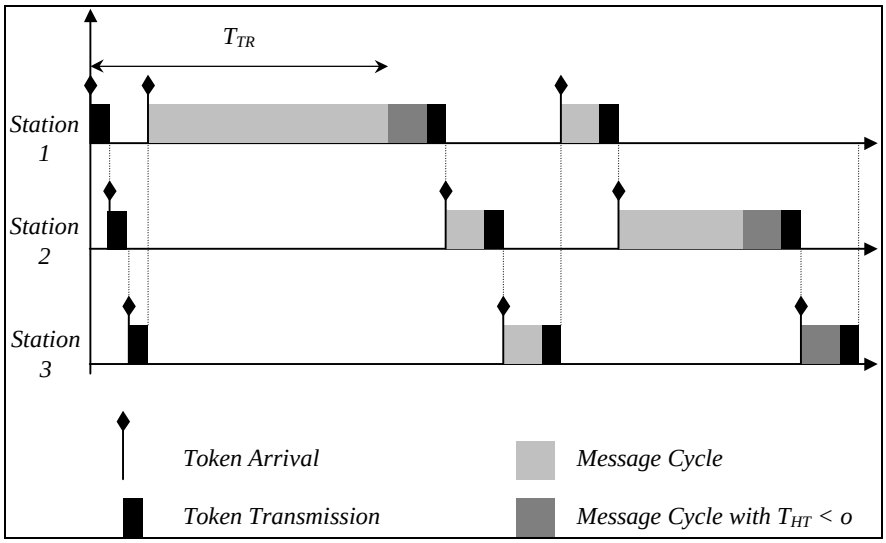

Figure 2

We will base our analysis in the following assumption: when a master station receives the token, $T_{RR}^{(k)}$ is always greater than T_{TR} thus only one high priority message will be transmitted.

4. A MAXIMUM BOUND FOR T_{TR}

4.1. Deadline Constraint: case of outgoing priority queues

Considering that we have only one high priority stream in a k master station, we must guarantee that after being produced, a message can be transmitted before the end of its deadline. To guarantee this, the token inter-arrival time to the station must be lower than the deadline.

If we denote the maximum time between consecutive token visits to a station k by $T_{cycle}^{(k)}$, this deadline constraint can be formulated as:

$$\frac{Dh_1^{(k)}}{Tcycle^{(k)}} \geq 1, \forall_k \tag{4.1}$$

If we now consider having $nh^{(k)}$ high priority streams in the k station, $T_{cycle}^{(k)}$ may be constrained as follows:

$$\sum_{i=1}^{nh^{(k)}} \frac{1}{Dh_i^{(k)} / Tcycle^{(k)}} \leq 1 \tag{4.2}$$

Expression (4.2) can be explained with a simple example. If we have only one high priority message stream in a k station, which deadline is 40ms, then the token should visit that station, at least, each 40ms. If the same station has 2 high priority message streams of 40ms deadlines then, the token should visit that station, at least, each 20ms. We should remember that we are supposing the worst case of only one high priority message transmitted per token visit, which is always guaranteed by Profibus, independently of the token delay.

As we will see, expression (4.2) will only stand for deadline constraint if the outgoing messages are put in a priority queue or, which is not reasonable, all the streams in the station k have equal deadlines.

This can also be shown with a simple example. If a station k has two streams of 40ms deadlines, from (4.2) $T_{cycle}^{(k)}$ should be equal to 20ms. Lets denote this as scenario A. If the same station has two streams, one with a 40ms deadline and the other with a 20ms deadline, then, from expression (4.2), we will have for $T_{cycle}^{(k)}$ the value of 13,3(3)ms. This will be denoted as scenario B.

We should note that messages related to each of the streams may be produced almost simultaneously. Lets suppose that in scenario B both messages were produced just after the station released the token. Additionally, suppose that the one with the 40ms deadline was put first in the FIFO outgoing buffer. Then, on a one message per token visit basis, the message with the 20ms deadline would miss it (13,3+13,3 = 26,6ms). This would not happen in scenario A (all streams have equal deadlines) nor if a different queuing mechanism was supported, for instance in a priority queue based on the messages deadline.

4.2. An Estimation of $T_{cycle}^{(k)}$

Assuming expression (4.2) for the deadline constraint, we need to estimate a value for $T_{cycle}^{(k)}$. Consider the following scenario (fig. 3), within which none of the three stations transmitted any message in the previous token cycle.

When station 1 receives the token it can send messages during a (T_{TR} - τ) time length. In fact it can hold the token by this amount plus the time corresponding to the transmission of the longest message issued by that station (including the non real-time messages). This may happens because once Profibus starts to send a message it will proceed till the end of the message cycle even if $T_{HT}^{(k)}$ time has elapsed.

For simplification we can derive the expression for $T_{cycle}^{(k)}$ using the maximum length for a message in the network (including high and low priority traffic):

$$Tcycle^{(k)} = T_{TR} + n \times C_{\max}, \forall_k \quad (4.3)$$

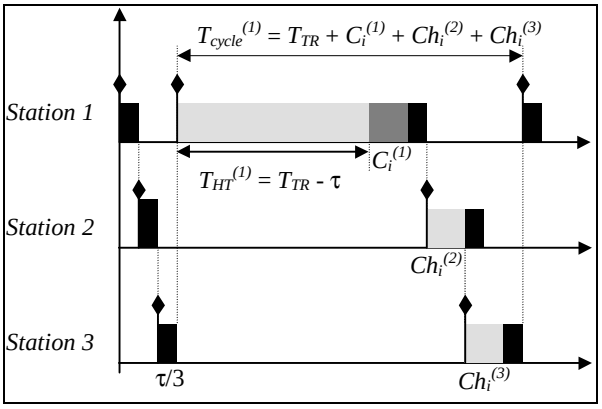

Figure 3

4.3. A First Approach to the Evaluation of T_{TR}

From (4.3), and considering that $T_{cycle}^{(k)}$ is identical for all stations (we now denote it as T_{cycle}) we can derive T_{TR} as being:

$$T_{TR} \le Tcycle \ - n \times C_{\max} \quad (4.4)$$

We can re-write expression (4.2) as follows:

$$Tcycle \le \frac{1}{\sum_{i=1}^{nh^{(k)}} \frac{1}{Dh_i^{(k)}}} \quad (4.5)$$

Thus, the station that imposes the shorter value for the Σ term in (4.5) will bound T_{TR}. Therefore, the value for T_{TR} is:

$$T_{TR} \le \min_k \left\{ \frac{1}{\sum_{i=1}^{nh^{(k)}} \frac{1}{Dh_i^{(k)}}} \right\} - n \times C_{\max} \quad (4.6)$$

This expression for T_{TR} uses a pessimistic evaluation of $T_{cycle}^{(k)}$ (similar message lengths).

4.4. Deadline Constraint: case of outgoing FIFOs

Considering a single outgoing FIFO in a *k* master station, between two consecutive token arrivals, multiple messages from different streams may be put in the outgoing FIFO. In a worst case scenario, assuming that only one message will be transmitted per token visit, then, *m* messages at outgoing FIFO will be dispatched only after *m* token arrivals. This is true if those messages are immediately produced after a station has passed the token to the subsequent one. The critical case is that of the message with the shortest value for $Dh_i^{(k)}$ is the last one in the FIFO. We will denote the Maximum Waiting Time for the most stringent high priority (lower deadline) message of a station *k* as $T_{MW}^{(K)}$.

Figure 4 illustrates this scenario. After a token rotation time of τ, station 1 receives the token. Just after token departure, messages from all the streams arrive to the outgoing buffer in the following order: $Mh_4^{(1)}$, $Mh_3^{(1)}$, $Mh_2^{(1)}$, $Mh_1^{(1)}$. Messages deadlines are reversed, considering the arrival order, i.e.: $Dh_4^{(1)} > Dh_3^{(1)} > Dh_2^{(1)} > Dh_1^{(1)}$.

Once more, in the worst case, the time between two consecutive token arrivals is T_{cycle} as given by (4.3). Assuming that, at each token arrival, only one high priority message is sent, the deadline constraint must be redefined.

The maximum waiting time for the stringent message in the outgoing FIFO of a *k* station with $nh^{(k)}$ high priority flows can be bounded to:

$$T_{MW}^{(k)} \le nh^{(k)} \times Tcycle \quad (4.7)$$

Then, the new deadline constraint would be as follows:

$$T_{MW}^{(k)} \le \min_{i=1..nh^{(k)}} \left\{ Dh_i^{(k)} \right\} \quad (4.8)$$

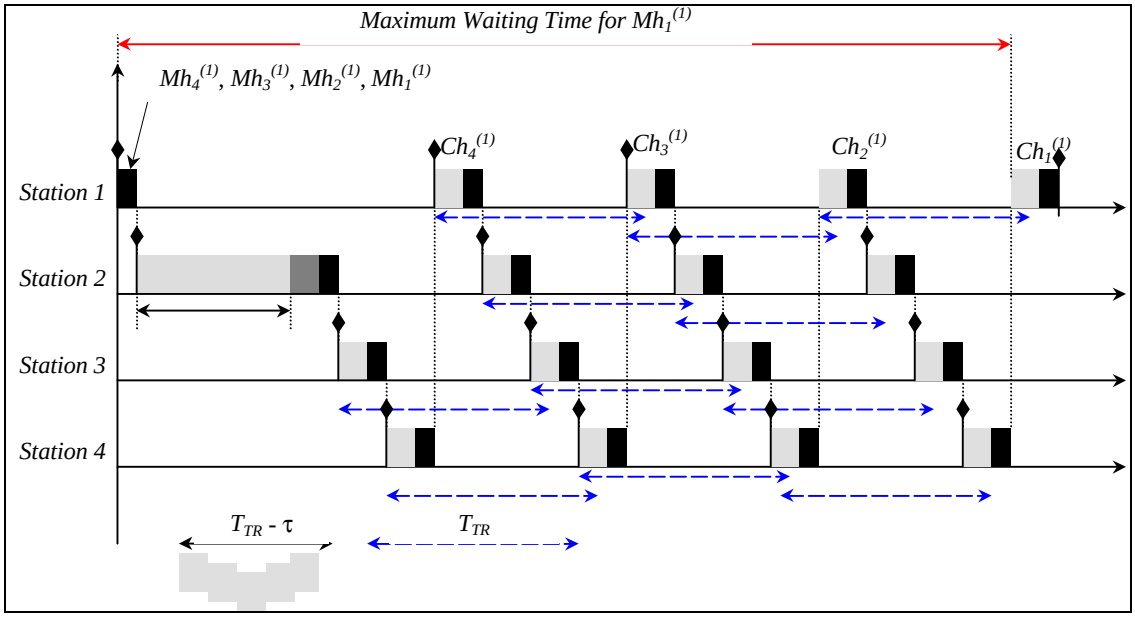

Figure 4

From expressions (4.7) and (4.8) we can write,

$$\min_{i=1..nh^{(k)}}\left\{Dh_i^{(k)}\right\} \geq nh^{(k)} \times \left(T_{TR} + n \times C_{\max}\right) \qquad (4.9)$$

which means that T_{TR} is bounded as follows,

$$T_{TR} \leq \min_{k}\left\{\frac{\min_{i=1..nh^{(k)}}\left\{Dh_i^{(k)}\right\}}{nh^{(k)}} - \left(n \times C_{\max}\right)\right\} \qquad (4.10)$$

This expression presents a maximum bound value for the T_{TR} parameter.

5. A More Accurate Bound for T_{TR}

The analysis that led to (4.10) expression was very pessimistic in two assumptions. One, and the most relevant one was considering that $T_{MW}^{(k)}$ would correspond to $nh^{(k)}$ consecutive $Tcycle^{(k)}$ (4.7). The other one relates to the use of a *Cmax* in (4.3), instead of specifying which are the messages accessing the bus at each station. In this section, we analyse the impact of these two considerations.

5.1. New Expression for $T_{cycle}^{(k)}$

Looking back to figure 4, and considering that with a late token only one high priority message can be transmitted, $T_{cycle}^{(k)}$ expression (4.3) can be re-write as follows (note that $T_{cycle}^{(k)}$ is now function of station *k*):

$$Tcycle^{(k)} = T_{TR} + \max_i\left\{C_i^{(k)}\right\} + \sum_{\substack{j \\ j \neq k}} \max_i\left\{Ch_i^{(j)}\right\} \qquad (5.1)$$

In (5.1) the second term corresponds to the situation where $T_{TH}^{(k)} \gtrsim 0$ before the beginning of a message cycle, and thus T_{TR} will be exceed in *k* station. This message cycle may correspond to either a high or a low priority one.

5.2. New Expression for $T_{MW}^{(k)}$

It is possible to show that, in the worst case, the time interval between *m* consecutive token arrivals is smaller than $m \times Tcycle^{(k)}$, because it is not possible to have *m* consecutive $Tcycle^{(k)}$ intervals.

Consider the following scenario (fig. 5): a 3 station bus with 4 messages arriving at the outgoing buffer in station 1 just after token departure. Considering that in the first and second token cycles, stations 2 and 3 use all the available bandwidth, $T_{MW}^{(K)}$ is bounded by:

$$T_{MW}^{(k)} = Tcycle^{(k)} + \left(nh^{(k)} - 1\right) \times T_{TR} \qquad (5.2)$$

since T_{TR} is always greater than the message lengths sum. Knowing that:

$$T_{MW}^{(k)} \leq \min_{i=1..nh^{(k)}}\left\{Dh_i^{(k)}\right\} \qquad (5.3)$$

then,

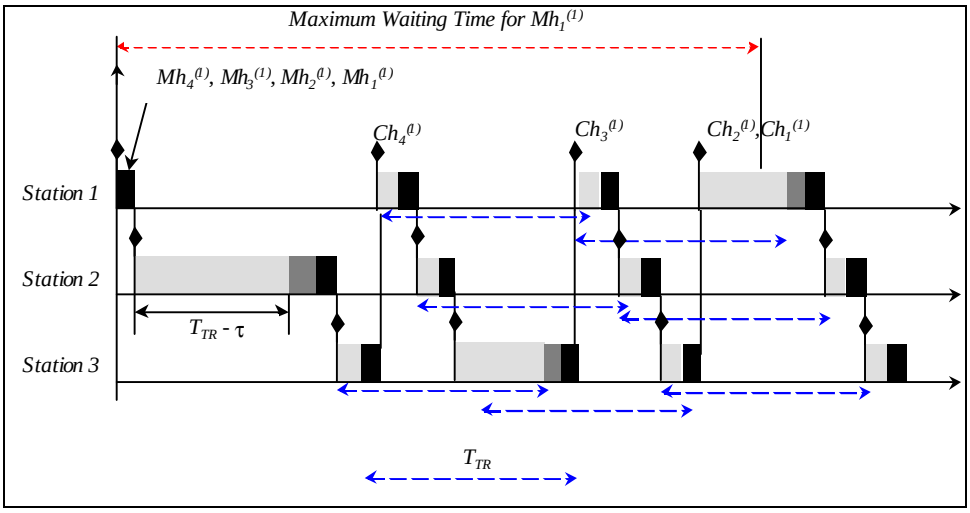

Figure 5

$$\min_i\left\{Dh_i^{(k)}\right\} \geq T_{TR} + \\ + \max_i\left\{C_i^{(k)}\right\} + \sum_{\substack{j \\ j\neq k}} \max_i\left\{Ch_i^{(j)}\right\} + \left(nh^{(k)} - 1\right)\times T_{TR} \qquad (5.4)$$

This expression can be simplified as follows:

$$\min_i\left\{Dh_i^{(k)}\right\} \geq nh^{(k)}\times T_{TR} + \max_i\left\{C_i^{(k)}\right\} + \sum_{\substack{j \\ j\neq k}} \max_i\left\{Ch_i^{(j)}\right\} \qquad (5.5)$$

Therefore, the deadline constraint can be relaxed (comparing with (4.10)) to:

$$T_{TR} \leq \min_k\left\{\frac{\min_i\left\{Dh_i^{(k)}\right\} - \max_i\left\{C_i^{(k)}\right\} - \sum_{\substack{j \\ j\neq k}} \max_i\left\{Ch_i^{(j)}\right\}}{nh^{(k)}}\right\} \qquad (5.6)$$

6. A NUMERICAL EXAMPLE

Consider a scenario with 3 master stations, each one with 3 high priority message streams, such as:

Master Station 1	Master Station 2	Master Station 3
$Sh_1^{(1)}$ = (10, 0.1)	$Sh_1^{(2)}$ = (14, 0.05)	$Sh_1^{(1)}$ = (12, 0.05)
$Sh_2^{(1)}$ = (10, 0.1)	$Sh_2^{(2)}$ = (20, 0.05)	$Sh_2^{(1)}$ = (12, 0.05)
$Sh_3^{(1)}$ = (10, 0.1)	$Sh_3^{(2)}$ = (30, 0.05)	$Sh_3^{(1)}$ = (20, 0.05)
C_{max} =0.2	C_{max} =0.2	C_{max} =0.2

From (4.10) we can compute the value of T_{TR}:

$$T_{TR} \leq \min_k\left\{\frac{\min_{i=1..3}\left\{Dh_i^{(k)}\right\}}{3} - (3\times 0.2)\right\}$$

$T_{TR}(k=1) = 2.73$; $T_{TR}(k=2) = 4.06$; $T_{TR}(k=3) = 3.4$

and so: $\boxed{T_{TR} \leq 2.73}$

Using (5.6), which gives a more accurate bound for T_{TR}, we have:

$$T_{TR} \leq \min_{k=1..3}\left\{\frac{\min_i\left\{Dh_i^{(k)}\right\} - \max_i\left\{C_i^{(k)}\right\} - \sum_{\substack{j=1,..,3 \\ j\neq k}} \max_i\left\{Ch_i^{(j)}\right\}}{3}\right\}$$

$T_{TR}(k=1) = 3.27$; $T_{TR}(k=2) = 4.56$; $T_{TR}(k=3) = 3.88$

and so: $\boxed{T_{TR} \leq 3.27}$, which provides more bandwidth to accommodate low priority traffic than in the previous case.

7. CONCLUSION

In this paper we have provided a comprehensive study on how to use Profibus networks to support real-time communication. The major contribution of this paper was to prove that, despite the limitations as regarding to the timed token protocol, it is possible to guarantee communication real-time behaviour with the Profibus protocol.

Fundamentally, we have derived a deadline constraint for both priority and FIFOs outgoing queues. This deadline constraint guarantees that high priority

messages are always transmitted before their deadlines.

Afterwards, we outlined the basis for tuning the T_{TR} network protocol parameter, which sets the expected time between token arrivals to master stations.

8. REFERENCES

[Agr92] Agrawal, G., B. Chen, W. Zhao and S. Davari, "Guaranteeing Synchronous Message Deadlines with the Timed Token Protocol", *Proceedings of the 12th IEEE International Conference on Distributed Computing Systems*, (June 1992).

[Cen96] EN 50170, General Purpose Field Communication System, European Standard, CENELEC, July 1996.

[Che92] Chen, B., G. Agrawal and W. Zhao, "Optimal Synchronous Capacity Allocation for Hard Real-Time Communications with Timed Token Protocol", *Proceedings of the 13th IEEE Real-Time Systems Symposium*, pp.198-207 (December 1992).

[Gro82] Grow, R.M., "A Timed Token Protocol for Local Area Networks", *Proceedings of Electro'82, Token Access Protocols, Paper 17/3*, (May 1982).

[Mal93] Malcolm, N. and W. Zhao, "Guaranteeing Synchronous Messages with Arbitrary Deadline Constraints in a FDDI Network", *Technical Report*, Department of Computer Science, Texas A&M University, (March 1993).

[Sae92] Controller Area Network (CAN), an In-Vehicle Serial Communication Protocol-SAE J1583, SAE Handbook, vol. II, 1992.

[Sev87] Sevcik, K.C. and M.J. Johnson, "Cycle Time Properties of the FDDI Token Ring Protocol", *IEEE Transactions on Software Engineering*, SE-13(3), pp. 376-385, (March 1987).

[Vas94] Vasques, F. and G. Juanole, "Pre-Run-Time Schedulability Analysis in Fieldbus Networks", *Proceedings of IECON'94*, pp. 1200-1204, (September 1994).

[Vas96] Vasques, F., "Sur L'Intégration de Mécanismes d'Ordonnancement et de Communication dans la Sous-Couche MAC de Réseaux Locaux Temps-Réel", *PhD Thesis*, available as technical Report LAAS Nº 96229, Toulouse, France, (June 1996).