Real-Time Communications over Hybrid Wired/Wireless PROFIBUS-based Networks*

Mário Alves, Eduardo Tovar
Polytechnic Institute of Porto
Porto, Portugal
{malves@dee,emt@dei}.isep.ipp.pt

Francisco Vasques
University of Porto
Porto, Portugal
vasques@fe.up.pt

Gerhard Hammer, Klaus Röther
Siemens AG
Karlsruhe, Germany
klaus.roether@khe.siemens.de

Abstract

PROFIBUS is an international standard (IEC 61158) for factory-floor communications, with some hundreds of thousands of world-wide installations. However, it does not include any wireless capabilities. In this paper we propose a hybrid wired/wireless PROFIBUS solution where most of the design options are made in order to guarantee the proper real-time behaviour of the overall network. We address the timing unpredictability problems placed by the co-existence of heterogeneous transmission media in the same network. Moreover, we propose a novel solution to provide inter-cell mobility to PROFIBUS wireless nodes.

1. Introduction

Fieldbus networks are becoming increasingly popular in industrial computer-controlled systems, allowing field devices like sensors, actuators and controllers to be interconnected at low cost, using less wiring and requiring less maintenance than point-to-point connections. PROcess FIeld BUS (PROFIBUS) [1] is one of the most popular fieldbuses, and has been granted the status of a real international standard by CENELEC [2].

The PROFIBUS medium access control (MAC) protocol is based on a token passing procedure (simplified version of the timed token protocol [3]) used by masters to grant the bus access to each one of them, and a master-slave procedure used by masters to communicate with slaves. Similar to other timed token MAC protocols, such as IEEE802.4 [4] or FDDI [5], PROFIBUS supports two categories of messages: high priority (or synchronous) and low priority (or asynchronous).

However, there is a major difference in the PROFIBUS MAC when compared to those two. In PROFIBUS there is no synchronous bandwidth allocation, which means that if a master station receives a late token (real token rotation time greater than target token rotation time), then only one high priority (synchronous) message will be transmitted. As a consequence, real-time approaches such as those described in, e.g., [6-8] can not be applied to the case of PROFIBUS networks.

PROFIBUS networks are widely used, with several hundreds of thousands of installations currently in operation worldwide. Recently, the timing properties of the protocol have been a focus of research. In [10] the authors suggest two different approaches to guarantee the real-time behaviour of the synchronous traffic in the PROFIBUS networks. In one of the approaches – the *Unconstrained Low Priority Traffic Profile*, the real-time requirements for the synchronous traffic are satisfied, even when only one synchronous message is transmitted per token visit, independently of the asynchronous traffic load. In this way, it is possible to have a guaranteed real-time approach for the message streams provided that the relative deadline for the synchronous message streams is larger than the worst-case message response time, which is given by:

$$R_i^k = Q^k + Ch_i^k = nh^k \times T_{cycle}^k + Ch_i^k \qquad (1)$$

where nh^k is the number of synchronous message streams generated in master k, T^k_{cycle} is the worst-case token rotation time and Ch_i^k is the worst-case duration of synchronous message cycle i issued by master k. The exact characterisation of the cycle time properties of the PROFIBUS token is described in [11], which permits the evaluation the T^k_{cycle} parameter in equation (1).

Implicit to equation (1) is the FCFS (First-Come First-Serve) behaviour of PROFIBUS MAC message queues. Additional work can be found in the literature on how the real-time capabilities of PROFIBUS networks can be improved if priority-based strategies are implemented for serving the synchronous traffic. In [12], guaranteed

* This work was partially supported by the European Commission, under the project IST-1999-11316 (RFieldbus).

approaches for both fixed priorities and deadline-based priorities are described.

The research works [10-12] on the timing behaviour of PROFIBUS networks have proved the capabilities of this fieldbus standard to support distributed computer-controlled systems with stringent real-time requirements. More recently, there is the eagerness of extending the capabilities of PROFIBUS to cover functionalities not previously considered in such type of networks: industrial wireless communications and ability to support industrial multimedia traffic. One example is reflected in the IST (Information Society Technology) project RFieldbus (High Performance Wireless Fieldbus in Industrial Multimedia-Related Environment) [13], supported by the European Commission.

In this paper we will focus on how it is possible to provide real-time communication over hybrid wired/wireless PROFIBUS-based networks.

2. Hybrid Wired/Wireless PROFIBUS Networks

A traditional fieldbus network consists of several end-systems (ES) physically connected through a wired bus. Therefore, and due to the market penetration, thinking about wireless means considering hybrid wired/wireless solutions able to interoperate with legacy (wired) systems.

We assume hybrid wired/wireless network topologies such as the one depicted in Figure 1.

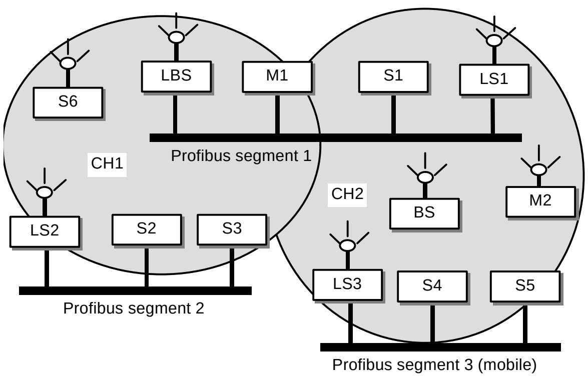

Figure 1: Hybrid wired/wireless network components

Wired PROFIBUS network master (M) and slaves (S) nodes communicate with wireless/mobile nodes through Link Stations (LS), Base Stations (BS) and Link Base Stations (LBS). The latter merge the functionality associated with LS and BS. Additionally, mobility of wired PROFIBUS segments (and associated nodes) is also considered (segment 3). LBS and BS operate in different radio channels (LBS in Radio Channel 1 and BS in Radio Channel 2) in order to have a structured wireless network, supporting inter-cell mobility. Inter-cell mobility will be further addressed in Section 5.

Wireless LAN technologies such as IEEE802.11b [14] or Bluetooth [15] could seem appropriate to support real-time wireless communications in such a hybrid wired/wireless network. This would be possible if all wireless stations had “traditional” PROFIBUS RS-485 interfaces. Nevertheless, in the proposed architecture, the goal is to support wireless stations with a specific wireless Physical Layer (non-PROFIBUS), which requires wired and wireless stations to have the same DLL protocol, i.e. PROFIBUS.

In the proposed architecture, Intermediate Systems act as repeaters, which leads to both reduced system complexity and reduced message response times. As a consequence, a simple and real-time handoff mechanism may be implemented (refer to Section 5). Simple because no explicit registering mechanism is necessary, reducing mobility management almost to radio channel assessment and switching. Real-time since the duration of the handoff mechanism is bounded and small.

In PROFIBUS, a master station is able to perform transactions during the token holding time. A transaction consists on the request frame and the associated acknowledgement or response frame. Transactions are atomic since requests are followed by a synchronous response (positive or negative). All PROFIBUS masters have a Data Link Layer (DLL) parameter, the Slot Time - T_{SL}, which must be set (in all masters) before the system is put into operation. This parameter defines the timeout before which a response/acknowledgement must arrive, and is also used for the token recovery mechanism.

The T_{SL} parameter assumes a particular importance. On one hand, T_{SL} must be set large enough to cope with the extra latencies introduced by the intermediate systems (even though they are MAC-less). On the other hand, T_{SL} must be set small enough to cope with the system responsiveness to failures, enabling the detection of a message/token loss or a station failure within a bounded time interval. Moreover, as T_{SL} is a component of the duration of a message transaction (*C*), its value will highly impact its worst-case response time.

Therefore, in order to minimise message response times, the use of intermediate systems acting as repeaters (MAC-less) is envisaged. The intermediate systems relay messages from one port to another in a store-and-forward or cut-through fashion, depending on the instant they start relaying PDUs.

As intermediate systems relay messages between different transmission media (e.g., different bit rates and frame format), there is the need for traffic adaptation mechanisms to avoid unpredictable and unbounded communication latencies due to traffic congestion. This problem will be addressed in the next section.

3. Congestion Control by Inserting Extra Idle Time

The heterogeneity in broadcast networks, in terms of bit rates and frame formats, imposes the consideration of traffic adaptation mechanisms. In many kinds of LANs this problem is solved by intermediate systems, usually acting as gateways, routers or bridges controlling traffic generation in transmitting stations or just discarding frames [16-18]. However, in a broadcast fieldbus network, where there are strict real-time and reliability requirements, a different approach must be followed.

To our best knowledge, there is no related work focusing this problem. In this section, we propose a solution where the responsibility of traffic adaptation is given to the end systems (ES), based on the insertion of additional idle time between consecutive frames. It is also important to note that this analysis can be applied to any type of broadcast network composed by heterogeneous transmission media.

Throughout the analysis we assume that the intermediate systems (IS) behave as store&forward repeaters (frames are forwarded after being completely received). However, a similar analysis may be applied to repeaters with cut-through behaviour, provided that all the Intermediate Systems have the same timing model.

3.1 Traffic Congestion in Intermediate Systems

The timing diagram depicted in Figure 2 illustrates a sequence of transactions between an initiator – I (who issues the request) and a responder – R (who issues the response), both in the same domain (D_a), and the resulting frames in the other domain (D_b). One intermediate system interconnects the two domains and it is assumed that the frame duration in D_b is twice the frame duration in D_a.

Clearly, if a request from an initiator in D_a and a responder in D_b appears after the last response shown in Figure 2, this transaction will be affected by the cumulative queuing delay in the intermediate system. The queuing delay in such an intermediate system depends on the number and duration of consecutive transactions where initiator and responder belong to D_a. Even a sequence of short frames may lead to very long message response times. For instance, a sequence of token passing between master stations that have nothing to transmit may also cause traffic congestion.

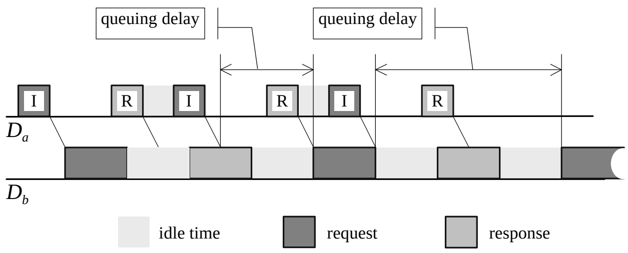

Figure 2: Increasing queuing delay in an IS

A way to avoid traffic congestion in intermediate systems (and long message response times) is through the insertion of an additional idle time before initiating a transaction. Obviously, the insertion of this additional idle time reduces the number of transactions per time unit when the responder is not in the same domain as the initiator. Nevertheless, the advantage of avoiding traffic congestion is enormous. It leads to a better responsiveness to failure (when an error occurs, retransmissions are undertaken sooner) and to bounded and smaller worst-case message response times.

3.2 Two Different Idle Time Parameters

In every master station, two different DLL idle time parameters must be defined - T_{ID1} and T_{ID2}, related to, respectively, acknowledged and unacknowledged requests. T_{ID1} is the time at the initiator between the receipt of a response frame's last bit and the instant when a new frame's bit is transmitted. T_{ID2} is the time interval between transmitting the last bit of an unacknowledged frame and transmitting the first bit of the next frame. The reasoning beyond these two distinct idle time parameters is explained next.

For a single segment network, all stations may set their idle time parameters to a minimum value, usually long enough to cope with bit synchronisation requirements. In the following subsections, we assume that all stations set these "minimum" idle time parameters to the same value, i.e. $T_{ID1}=T_{ID2}=T_{ID}$. Note that this is the idle time all intermediate systems will use, when relaying traffic from one port to the other ('T' is expressed in bit times and 't' is expressed in seconds).

Then, we compute the additional idle time that each station must insert, in order to perform the traffic adaptation. These inserted idle times are represented by t_{ID1+} and t_{ID2+}. Finally, we merge the corresponding components into single parameters – T'_{ID1} and T'_{ID2}.

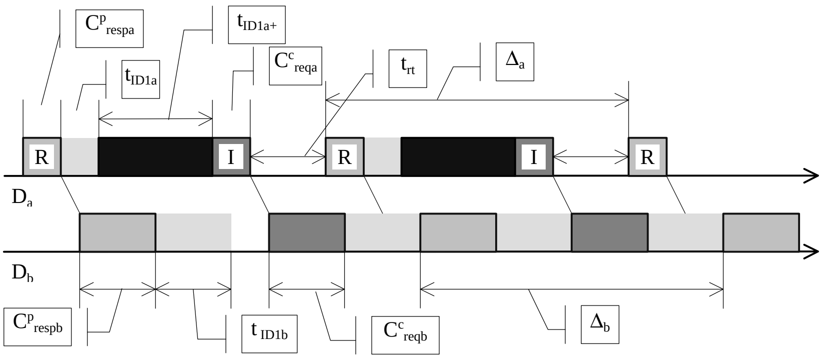

Figure 3: Inserting additional idle time (acknowledged request sequence)

As we will see, a master station could hold a unique idle time, i.e. wait the same idle time after receiving response frames or sending unacknowledged requests. Nevertheless, this would demand this unique idle time to be the maximum between T'_{ID1} and T'_{ID2}. Obviously, this would lead to a non-optimal situation. Indeed, previous numerical results [19] indicate that T'_{ID2} is usually smaller than T'_{ID1}. If we consider a unique idle time (that is the maximum between the two), the unacknowledged requests would be penalised (inserting more idle time than needed). The following subsections show how to set both idle times.

3.3 Computing the Inserted Idle Time After Receiving a Response

In order to compute the inserted idle time after receiving a response frame (t_{ID1a+}), please consider the scenario presented in Figure 3, where a sequence of message cycles including the inserted idle time is presented. For the sake of simplicity, the timing diagram depicted in Figure 3 assumes that the frame duration in D_b is twice the frame duration in D_a. The responder's turnaround time is represented by t_{rt} (assumed to be constant) and the superscript indexes *p* and *c* correspond to *previous* and *current* (transaction), respectively.

Clearly, the increase in the idle time (t_{ID1a+}) guarantees that there will be at most two messages in an intermediate system's queue, one being processed and the other one waiting to be served.

Reporting to Figure 3, D_a should be greater or equal to D_b, in order to be able to avoid the increase in the queuing delay. That is, if:

$$\Delta_a = C^p_{respa} + t_{ID1a} + t_{ID1a+} + C^c_{reqa} + t_{rt}$$

$$\Delta_b = C^p_{respb} + t_{ID1b} + C^c_{reqb} + t_{ID1b}$$

then:

$$t_{ID1a+} \geq \left(C^p_{respb} - C^p_{respa}\right) + \left(C^c_{reqb} - C^c_{reqa}\right) + \left(2 \cdot t_{ID1b} - t_{ID1a}\right) - t_{rt} \qquad (2)$$

In order to compute the value for t_{ID1+} for a given master station, there is the need to know the characteristics of the message streams related to that master. Therefore, we must know the length of the different DLL request/response PDUs.

3.4 Computing the Inserted Idle Time After Issuing an Unacknowledged Request

The condition expressed in (2) is just related to acknowledged request frames. The case of a sequence of non-acknowledged request (or token) frames must also be analysed.

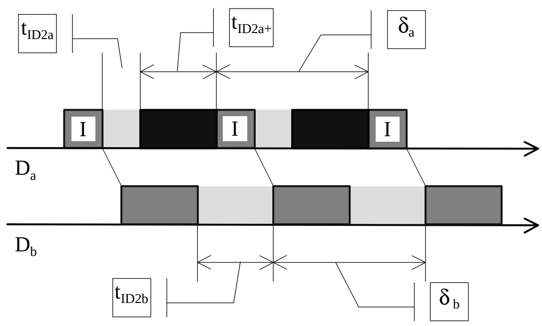

Figure 4: Inserting additional idle time (unacknowledged request sequence)

Figure 4 shows a sequence of unacknowledged requests, where d_a should be greater or equal to d_b. That is, if:

$$d_a = C_{reqa} + t_{ID2a} + t_{ID2a+}$$

$$d_b = C_{reqb} + t_{ID2b}$$

then:

$$t_{ID2a+} \geq \left(C_{reqb} - C_{reqa}\right) + \left(t_{ID2b} - t_{ID2a}\right) \quad (3)$$

Thus, the length of the different DLL unacknowledged requests for that master must be known, in order to compute t_{ID2+}.

Finally, if there is a restriction of a single register for each T_{ID}, it is necessary to merge both the "conventional" idle time and the inserted idle time:

$$t'_{ID1a} = t_{ID1a} + t_{ID1a+} \wedge t'_{ID2a} = t_{ID2a} + t_{ID2a+} \quad (4)$$

or, in bit times:

$$T'_{ID1a} = T_{ID1a} + T_{ID1a+} \wedge T'_{ID2a} = T_{ID2a} + T_{ID2a+} \quad (5)$$

We consider also that a master station will insert T'_{ID2} after receiving a token frame[1]. This is necessary since a sequence of token passing between master stations that have nothing to transmit may also "jam" the intermediate system.

The methodology presented in Section 3.3 and 3.4 permits to set both idle time parameters in a per-station basis, taking into account all possible transactions (message streams) for that master stations. In this sense, each master station in the network would have a unique pair (T'_{ID1}, T'_{ID2}) of idle time parameter values. An algorithm that returning the idle time parameter values for all masters stations in a given domain (therefore, in a per-domain basis) was proposed in [19]. In this paper we consider a worst-case scenario where maximum and minimum frame lengths for the overall network were taken in consideration.

4. Computing the Duration of Message Transactions

In this section, we evaluate the duration of message transactions in hybrid wired/wireless fieldbus networks, that operate in a broadcast fashion. Such duration includes both the duration of the message itself and the duration of its transmission time. The duration of a message transaction is mainly dependent on the duration of the request and response frames and on the number and type of physical mediums that the frames must cross between initiator and responder. It is also dependent on the extra idle time that must be inserted between consecutive frames in the network.

4.1 System Turnaround Time

In order to evaluate the duration of a transaction, it is necessary to determine the the time interval between the end of the request's transmission and the beginning of the response's reception (system turnaround time). To clarify this definition, consider the network topology depicted in Figure 5.

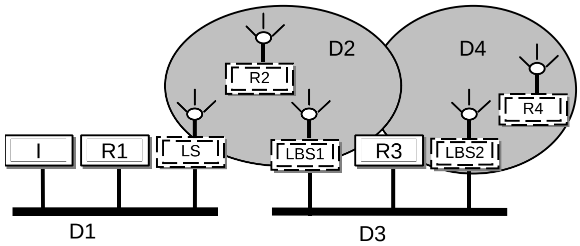

Figure 5: Example of a hybrid wired/wireless network topology

Assume that D1 and D3 have the same type of PhL (PDU format and bit rate) and D2 and D4 have a different type of PhL. Figure 6 depicts the timing diagram for the longest transaction between I and R4 (considering no queuing delays). Both request and response frames must be relayed by 3 intermediate systems (LS, LBS1 and LBS2).

Clearly, if for a transaction the responder belongs to the same domain as the initiator (e.g. I and R1), the system turnaround time – t_{st} – equals the responder turnaround time. Oppositely, when there is one or more intermediate systems between initiator and responder (e.g. I and R4), the system turnaround time will increase. This is the case depicted in Figure 6. Nevertheless, the timing diagram is simplified, since the request frame may be delayed by the previous frame, if the duration of the previous frame is higher than the duration of the request frame.

We assume a *pessimistic situation*, i.e., for each master it is considered that the request PhL PDU is always longer than the previous PhL PDU in the network (previous response frame or unacknowledged request frame). This way we are able to easily determine an upper bound for the system turnaround time, since there is no need to compute the queuing delay in intermediate systems.

In Figure 6, t_{bd} represents the buffering delay of the intermediate systems and the responder turnaround time (t_{rt}) is assumed to be constant for every station.

[1] This demands the decoding of the DLL PDU, in order for the master station to know if it received a token frame.

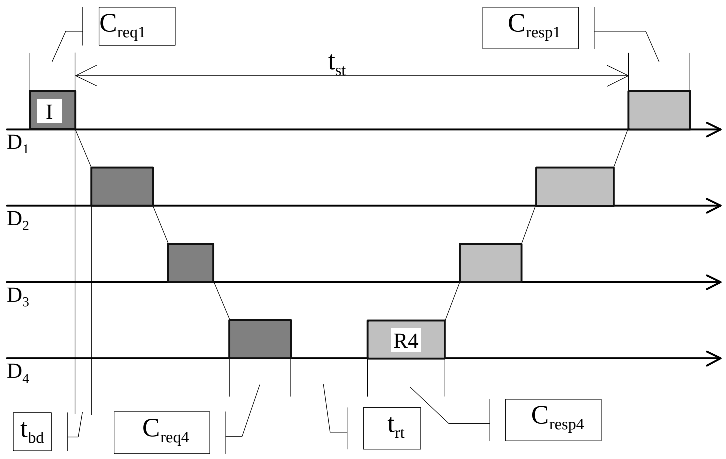

Figure 6: Timing diagram for transaction between I and R4

The worst-case system turnaround time may be evaluated as follows:

$$t_{st} = \left(C_{req2} + C_{resp2} + 2 \cdot t_{bd}\right) + \left(C_{req3} + C_{resp3} + 2 \cdot t_{bd}\right) + \\ + \left(C_{req4} + C_{resp4} + 2 \cdot t_{bd}\right) + t_{rt}$$

In the general case:

$$t_{st} = \sum_{i=2}^{n} \left(C_{reqi} + C_{respi} + 2 \cdot t_{bd}\right) + t_{rt} \qquad (6)$$

where i represents the domains involved in the transaction and n represents the domain of the responder. C_{reqi} is the duration of the longest request PDU of the initiator, in domain i. C_{respi} is the duration of the actual response PDU in domain i.

4.2 Computing the duration of transactions

Finally, the duration of a transaction can be easily evaluated summing its components. That is, the duration of a acknowledged request/response transaction (C_{ack}) is the sum of the duration of both the request and response frames, plus the system turnaround time and the inserted idle time (T'_{ID1}). This is depicted in Figure 7.

The duration of an unacknowledged transaction (C_{unk}) is just the sum of the duration of the frame plus the inserted idle time (T'_{ID2}).

The duration of the request/response transaction depicted in Figure 7 can be evaluated as follows:

$$C_{ack} = C_{req1} + t_{st} + C_{resp1} + t'_{ID1}$$

In the general case:

$$C_{ack} = \sum_{i=1}^{n} \left(C_{reqi} + C_{respi}\right) + (n-1) \cdot 2 \cdot t_{bd} + t_{rt} + t'_{ID1} \qquad (7)$$

Remember that since we opted for the pessimistic approach (guaranteeing no queuing delays in intermediate systems), the request PhL PDU duration is always equal the maximum PhL PDU duration in the network. A graphical presentation of the variables involved in the computation is given in Figure 7, for a transaction between I and R3.

The duration of an unacknowledged transaction does not depend of the number of intermediate systems between the initiator and the responder, since there is no need to wait for any response/acknowledge to proceed (issue another request or pass the token). Such duration can thus be evaluated as

$$C_{unk} = C_{req} + t'_{ID2} \qquad (8)$$

Equation (1) presented in Section 1, can still be used to evaluate the message's worst-case response time in hybrid wired/wireless PROFIBUS networks, where the values for the worst-case message transaction times are evaluated as presented in Section 4. Note also that the T^{k}_{cycle} parameter in equation (1) is also a function of the transaction durations (see [11] for further details).

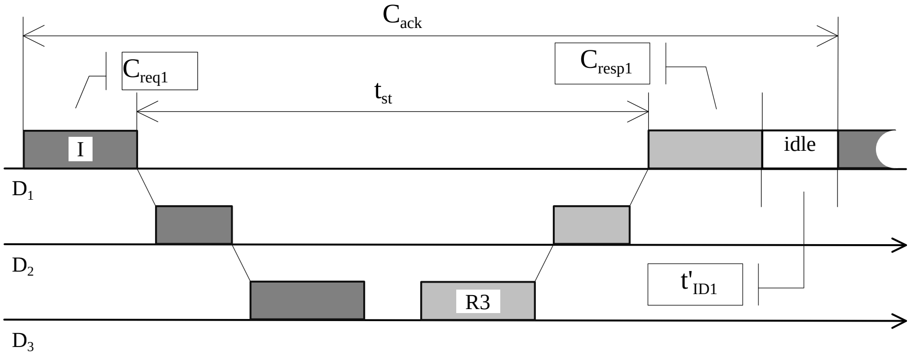

Figure 7: Duration of a Transaction

5. A Real-time Solution for Inter-cell Mobility

Since the underlying communication platform is PROFIBUS and Intermediate systems act as repeaters (there is a single logical ring), only one node in the overall network is able to communicate at a given time instant. The proposed solution provides a seamless handoff for all types of mobile stations (mobile master/slave/LS), where there is no need for an explicit registering mechanism. Therefore, the mobility management mechanism just encompasses a procedure for radio channel assessment and switching. Importantly, the proposed mobility management mechanism guarantees no loss of data PDUs due to mobility of nodes and permits to fulfil stringent real-time requirements. In fact, mobility management is restricted to a reduced and bounded period of time (less than 2 ms for topologies with considerable topological complexity).

5.1 General Description

One specific station - the *mobility master (MobM)* - is the responsible for triggering the mobility management procedure (Figure 8). Within a certain period – *beacon period*, all mobile stations are expected to assess the quality of the different radio channels, switching to the best quality channel.

The mobility master (i.e., the master that has the responsibility of triggering the handoff procedure) broadcasts a special (unacknowledged) frame – the *beacon trigger*, with a periodicity that is dependent on the maximum speed of the mobile stations. The reception of the beacon trigger causes each base station to send a number of beacons in its radio channel. Mobile stations listen to these PDUs, assess the signal quality of all radio channels and switch to the best channel. This is roughly depicted in Figure 8.

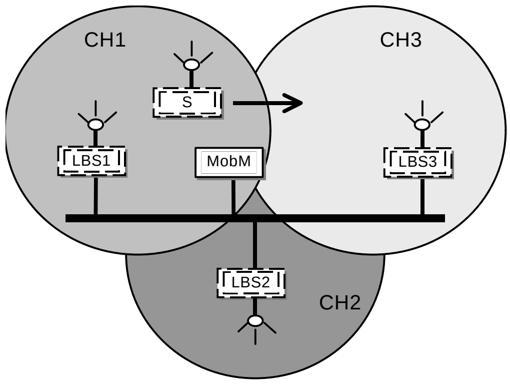

Figure 8: The mobility master (MobM)

The beacon trigger (sent by the mobility master) is received (and relayed) by the (Link) Base Stations, which then start to send beacons (special frames) in their own frequency, enabling the mobile stations to do the channel assessment and handoff. Considering the scenario presented in Figure 8, where the mobile station is moving towards LBS3, it must perform channel assessment and switch to CH3 (Figure 9). After this mobility management period, the mobility master is able to pass the token to another master.

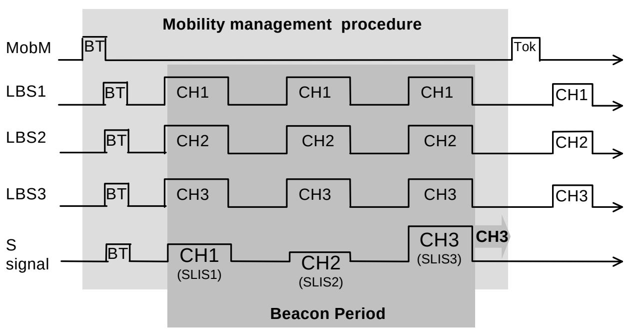

Figure 9: Mobility management procedure timing diagram

5.2 Computing the Mobility Management Duration

It is fundamental to evaluate the maximum duration of the mobility management procedure, in order to insert the appropriate idle time before passing the token. The mobility master must guarantee that the last mobile station to receive the BT has still enough time to perform the handoff procedure.

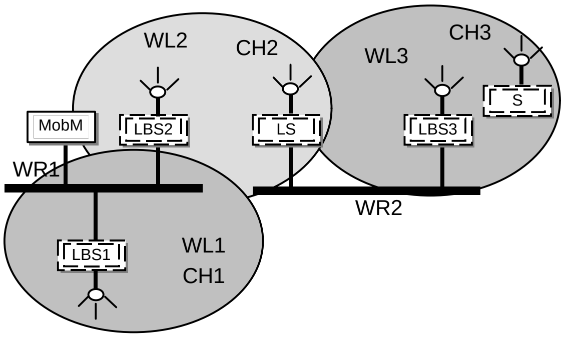

Figure 10: Example of a network topology

Let us consider a scenario (Figure 10) with three cells and three different radio channels, where the maximum duration of the mobility management procedure may be determined considering a wireless station (S) in domain WL3, as *MobM* is located in domain WR1. Figure 11 presents the timing diagram of the related mobility management.

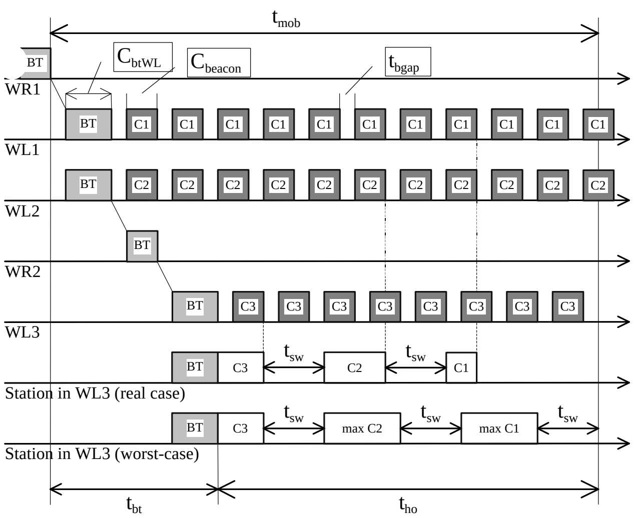

Figure 11: Mobility management timing diagram

We assume that the mobile station starts the handoff procedure immediately after receiving the BT, beginning the assessment in the current channel (CH3, in the example). After that, the station switches to another channel (CH2) and does the assessment, switches to the other channel (CH1) and does the assessment and finally switches to the best channel. Considering the worst-case situation when assessing CH1 and CH2, i.e. the mobile station starts assessing the channel immediately after the beginning of the beacon frame, the maximum assessment period for each of those channels is:

$$2 \cdot C_{beacon} + t_{bgap}$$

where C_{beacon} is the duration of a beacon frame and t_{bgap} is the interval between beacon frames. A timer will monitor this time interval to guarantee the correct processing in the case of missing a beacon frame,. Considering that the number of radio channels (to assess) is denoted as *m* and the switching time is defined as t_{sw}, the maximum duration of the handoff procedure in the mobile station is:

$$t_{ho} = t_{bgap} + C_{beacon} + t_{sw} + (m-1)\cdot\left(2\cdot C_{beacon} + t_{bgap} + t_{sw}\right) = \\ = (2\cdot m - 1)\cdot C_{beacon} + m\cdot\left(t_{bgap} + t_{sw}\right) \quad (9)$$

In the case of *m=3* (Figure 10) the handoff takes:

$$t_{ho} = 5\cdot C_{beacon} + 3\cdot t_{bgap} + 3\cdot t_{sw}$$

Nevertheless, the handoff procedure duration is only one of the two components of the mobility management period. It is also necessary to determine the time needed for the beacon trigger frame to arrive to the "most distant" wireless station – t_{bt},:

$$t_{bt} = \sum_{i=2}^{n}\left(t_{bd} + C_{bti}\right) \quad (10)$$

where *i* represents the range of domains involved from the mobility master to the "most distant" one and C_{bti} is the duration of the beacon trigger PhL PDU in domain *i*. Finally, the mobility management period can be computed as the sum of the previous components:

$$t_{mob} = t_{bt} + t_{ho} \quad (11)$$

5.3 Computing the Number of Beacons

In order for the mobility management procedure to work properly, the base stations must know the exact number of beacons they must issue, upon reception of a BT, which may vary depending on the base station. Moreover, considering that the beacon transmission is non pre-emptive (i.e., once a base station starts to transmit a beacon, it must complete the transmission until the end), there will be the need to adjust the mobility management period. Therefore, it is necessary to evaluate the (integer) number of beacons for each base station in the network and then make the related adjustment to the mobility management duration.

Consider the following figure:

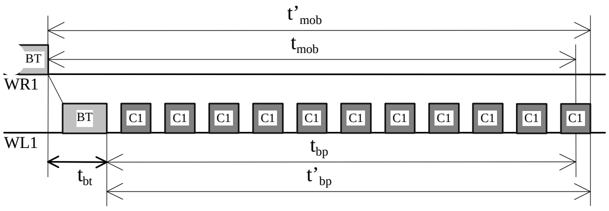

Figure 11: Mobility management timing diagram

We define the minimum duration of the beacon period for the correspondent base station as:

$$t_{bp}(BS) = t_{mob} - t_{bt}(BS) \quad (12)$$

The number of beacons (n_b) that must be sent by the base station may be computed as:

$$n_b(BS) = \left\lceil \frac{t_{bp}(BS)}{t_{bgap} + C_{beacon}} \right\rceil \quad (13)$$

The actual duration of the beacon period is thus:

$$t'_{bp}(BS) = n_b(BS) \cdot \left(t_{bgap} + C_{beacon}\right) \quad (14)$$

imposing a minimum mobility management time of:

$$t'_{mob}(BS) = t_{bt}(BS) + t'_{bp}(BS) \quad (15)$$

This procedure must be undertaken for all base stations, considering the maximum t'_{mob}. Then, in the mobility master, the idle time parameter T_{ID} should be set to a minimum value of:

$$T_{ID} = t'_{mob} \cdot r \quad \text{bit times} \quad (16)$$

where r represents the bit rate in the physical layer of the mobility master.

5.4 Location of the Mobility Master

Since the mobility management duration depends on the location of the mobility master, this latency should be minimised at the system design phase. As a rule of thumb, the mobility master functionality should be responsibility of a master located in the "central" domain, i.e., in a domain that is "equidistant" from the "most distant" wireless domains.

6. Conclusions

In order to guarantee the real-time behaviour of a distributed system, it is mandatory to evaluate the worst-case message response times. In a token-passing fieldbus network, the response time of a particular message is mainly dependent on the medium access delay and on the duration of the transaction. In a hybrid wired/wireless fieldbus network working in a broadcast fashion, the duration of a transaction is potentially higher than in a single-segment fieldbus network, since one or more Intermediate Systems may exist between the two communicating peers. Moreover, taking into account that wired and wireless physical layers have different physical layers (bit rate and PDU format), increasing queuing delays will occur in the intermediate systems (repeaters). This would lead to unpredictable and unbounded system turnaround times, if an appropriate congestion elimination mechanism was not provided.

We started by defining some architectural features of the hybrid wired/wireless fieldbus network, namely on the fieldbus protocol (PROFIBUS), system components and network topology. Then we introduced the general problem of network congestion and proposed an innovative congestion elimination mechanism based on the insertion of inactivity (idle) times by master stations. These inactivity times allow the intermediate systems to relay messages without increased queuing in the intermediate systems (repeaters), permitting to reduce and bound system turnaround times, allowing to compute the duration of message transactions. Finally, we presented an innovative mobility management mechanism that uses native PROFIBUS features and provides a seamless handoff for mobile master and slave stations and also for mobile link stations (mobile segments). One of the pros of this mechanism is its timing determinism, since the duration of the mobility management "period" may be easily determined *a priori*.

References

[1] PROFIBUS Standard DIN 19245 part I and II, translated from German, PROFIBUS Nutzerorganisation e.V., 1992.

[2] EN 50170 - General purpose field communication system, CENELEC, 1996.

[3] Grow, R., "A Timed Token Protocol for Local Area Networks", in Proceedings of Electro'82, Token Access Protocols, Paper 17/3, 1982.

[4] IEEE, "IEEE Standard 802.4: Token Passing Bus Access Method and Physical Layer Specification", IEEE, 1985.

[5] ISO 9314-2, "Information Processing Systems - Fibre Distributed Data Interface (FDDI) - Part 2: Token Ring Media Access Control (MAC)". ISO, 1989.

[6] Agrawal, G., Chen, B., Zhao, W. and Davari, S., "Guaranteeing Synchronous Message Deadline with the Timed Token Medium Access Control Protocol", IEEE Transactions on Computers, Vol. 43, No. 3, pp. 327-339, 1994.

[7] Montuschi, P., Ciminiera. L. and Valenzano, A., "Time Characteristics of IEE802.4 Token Bus Protocol", IEE Proceedings, Vol. 139, No. 1, pp. 81-87, 1992.

[8] Zheng, Q. and Shin, K., "Synchronous Bandwidth Allocation in FDDI Networks", IEEE Transactions on Parallel and Distributed Systems, Vol. 6, No. 12, pp. 1332-1338, 1995.

[9] S. Zhang, E. Lee, A. Burns and H. Yang, Finding the Minimum Available Transmission Time for the Timed Token Medium Access Control Protocol, Proceedings of the 11th Euromicro Conference on Real-Time Systems, York, IEEE Society Press, pp185-194, June 1999.

[10] Tovar, E. and Vasques, F., "Real-Time Fieldbus Communications Using PROFIBUS Networks", IEEE Transactions on Industrial Electronics, vol. 46, nº 6, December 1999.

[11] Tovar, E. and Vasques, F., "Cycle Time Properties of the PROFIBUS Timed-Token Protocol", Computer Communications 22 (1999), pp. 1206-1216, Elsevier.

[12] Tovar, E. and Vasques, F., "Non Pre-Emptive Scheduling of Messages on SMTV Token-Passing Networks", proceedings of the 12th IEEE Euromicro Conference on Real-Time Systems, Stockholm, Sweden, pp. 209-218, June 2000.

[13] Haehniche, J., Rauchhaupt, L., "Radio Communication in Automation Systems: the R-Fieldbus Approach", in Proc. of the 2000 IEEE International Workshop on Factory Communication Systems, pp. 319-326, September 2000.

[14] O'Hara, B., Petrick, Al, "802.11 Handbook – A Designer's Companion", IEEE Press, 1999.

[15] Bilstrup, U., Wiberg, P.-A., "Bluetooth in Industrial Environment", in Proceedings of the 2000 IEEE International Workshop on Factory Communication Systems, pp. 239-246, September 2000.

[16] Van Jacobson, "Congestion Avoidance and Control", Proc. of SIGCOMM'88, Standforf, CA, USA, August 1988.

[17] Raj Jain, "Congestion Control in Computer Networks: Issues and Trends", IEEE Network Magazine, May 1990, pp. 24-30.

[18] R. Jain, K. Ramakrishnan, D.-M. Chiu, "Congestion Avoidance in Computer Networks With a Connectionless Network Layer", Digital Equipment Corporation Technical Report DEC-TR-506, 1988.

[19] Mário Alves, Eduardo Tovar, Francisco Vasques, On the Adaptation of Broadcast Transactions in Token-Passing Fieldbus Networks with Heterogeneous Transmission Media, 4th IFAC Conference on Fieldbus Systems and their Applications – FeT'01, Nancy, France, November 2001, pp. 278-284.