Evaluating the Duration of Message Transactions in Broadcast Wired/Wireless Fieldbus Networks

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

Francisco Vasques
FEUP, University of Porto
R. Roberto Frias, Porto, Portugal
vasques@fe.up.pt

Abstract

Determining the response time of message transactions is one of the major concerns in the design of any distributed computer-controlled system. Such response time is mainly dependent on the medium access delay, the message length and the transmission delay. While the medium access delay in fieldbus networks has been thoroughly studied in the last few years, the transmission delay has been almost ignored as it is considered that it can be neglected when compared to the length of the message itself. Nevertheless, this assumption is no longer valid when considering the case of hybrid wired/wireless fieldbus networks, where the transmission delay through a series of different mediums can be several orders of magnitude longer than the length of the message itself. In this paper, we show how to compute the duration of message transactions in hybrid wired/wireless fieldbus networks. This duration 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. A case study of a hybrid wired/wireless fieldbus network is also presented, where it becomes clear the interest of the proposed approach.

1. Introduction

Most computer-controlled systems are also real-time systems. In general, the issue of guaranteeing real-time requirements is the one of checking, prior to run-time, if the worst-case execution time of each of its tasks is smaller than the admissible response time. In distributed computer-controlled systems, where some of the application tasks are communicating tasks, the evaluation of the message's response time is of paramount importance.

The message's response time is mainly dependent on the medium access delay (contention due to other messages in the queue and due to other stations holding the token), the message length and the transmission delay. While the medium access delay in fieldbus networks has been thoroughly studied in the last few years [1,2,3,4], the transmission delay has been almost ignored as it is considered that it can be neglected when compared to the length of the message itself.

Nevertheless, this assumption is no longer valid when considering the case of hybrid wired/wireless fieldbus networks, where the transmission delay through a series of different mediums can be several orders of magnitude longer than the length of the message itself.

In this paper, 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 [5]. This additional inactivity time is necessary in order to avoid network congestion, since we are addressing hybrid wired/wireless networks containing different physical layer PDU formats and bit rates.

This paper is organised as follows: Section 2 presents the main characteristics of the considered wired/wireless fieldbus network. Section 3 presents general formulae to compute the duration of transactions in hybrid wired/wireless fieldbus networks with the previously described characteristics. For that purpose, a way to compute the PhL PDU duration, the idle time and the system turnaround time is also outlined in that section. Section 4 introduces a hybrid wired/wireless fieldbus system based on the EN50170 Profibus profile [6], which is under development - the RFieldbus system [7]. As some default network parameters can be considered in the RFieldbus system, in Section 5 we present a simplified timing analysis for RFieldbus transactions. The last section draws some conclusions.

2. Network Model

2.1 MAC, Message Transactions and Linking Devices

We consider a hybrid wired/wireless fieldbus network where the medium access control (MAC) protocol is based in a token-passing procedure used by master stations to grant the bus access to each other. A master station is able to perform message transactions during the token holding time. A message transaction consists of the request from a master (initiator) and the

associated response frame (positive or negative) immediately issued by the responder (master or slave). The master will only process another transaction (or pass the token) upon completion of the ongoing transaction and waiting a pre-defined idle time. If an erroneous response frame is received or a timeout (before receiving any response) occurs, the master station may retry the request.

A master station can also send unacknowledged requests. In this case, as there is no associated response frame, it will be able to start another transaction (or pass the token) just after a pre-defined idle time. The idle times between consecutive frames in the network should always be respected due to physical layer (PhL) requirements (namely for synchronisation).

In order to have a broadcast network, linking devices must act as repeaters. We assume a store-and-forward behaviour, i.e. a frame must be completely received by one port of the linking device before being re-transmitted to the other port. Obviously, the linking devices must support encapsulation/decapsulation operations, due to different PhL PDU (protocol data unit) formats, and receiving/transmitting at different bit rates.

2.2 PhL PDU length and duration

In a hybrid wired/wireless fieldbus network, linking devices interconnect domains that use the same data link layer (DLL) protocol, but have different physical layer (PhL). Therefore, we have to define specific parameters meaningful for each domain:

Parameter	Description	Units
L	Length of the DLL PDU	chars
k_a	Length of a char in the PhL of domain *a*	bits/char
l_{a+}	PhL overhead of domain *a* (header, preamble, SFD)	bits
r_a	Bit rate in domain *a*	Mbit/s

Table 1: Parameters for frame length and duration

A character (char) is defined as the smallest unit of information in the DLL. A DLL PDU (Protocol Data Unit) is a set of chars delivered to the PhL for transmission. In order to proceed with this transmission, the PhL may have to introduce additional information (header) or synchronisation (preamble, start frame delimiter) bits. Moreover, a character of the DLL may have different lengths at the PhL, depending on the type of PhL.

Taking into account the parameters outlined in Table 1, we can define C_a as the duration of a PhL PDU in domain *a*:

$$C_a = \frac{L \cdot k_a + l_{a+}}{r_a}$$

3. Computing the Duration of Message Transactions

3.1 Inserted Idle Time

Broadcast networks that are characterised by having different physical layers (PhL) demand some kind of traffic adaptation between segments, in order to avoid traffic congestion in linking devices. In [5], the authors propose a method on the insertion of an appropriate idle time before a station issuing a request frame. In this way, it is guaranteed that the linking devices' queues do not increase in a way that the timeliness properties of the overall system turn out to be unsuitable for the targeted applications. The inserted idle time for a certain master station mainly depends on the characteristic of the message streams of that master and on the PhL PDU formats and bit rates in the network. In this subsection, the way to compute the idle time is summarised.

3.1.1 Two different idle time parameters

In every master station, two different DLL idle time parameters are defined - T_{ID1} and T_{ID2}, related to acknowledged and unacknowledged requests, respectively. T_{ID1} is the time that expires at the initiator after receipt of a response frame's last bit, until a new frame's bit is transmitted on the medium. T_{ID2} is the time that expires between transmitting the last bit of an unacknowledged frame and transmitting the first bit of the next frame.

For a single segment network, all stations may set their idle time parameters to a minimum value, usually big enough to cope with synchronisation requirements. In the following subsections, we are going to assume that all stations set these "minimum" idle time parameters to the same value[1], i.e. $T_{ID1}=T_{ID2}=T_{ID}$. Then, we compute the additional idle time each station must insert, in order to perform 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}.

3.1.2 Computation of the inserted idle time after receiving a response

In order to compute the inserted idle time after receiving a response frame (t_{ID1a+}), we will refer to Figure 1. A sequence of transactions including the inserted idle time is presented. For the sake of simplicity, the timing diagram assumes that the frame duration in D_b is twice the frame duration in D_a.

[1] Note that this is the idle time all linking devices will use, when relaying traffic from one port to the other.

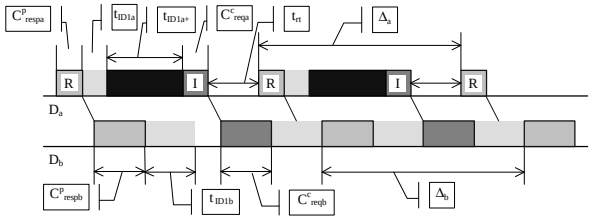

Figure 1: Inserting additional idle time (acknowledged request sequence)

The superscript indexes *p* and *c* correspond to *previous* and *current* (transaction), respectively, and the responder turnaround time is represented by t_{rt}.

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

Reporting to Figure 1, we can state that D_a should be greater or equal to D_b, in order to be able to avoid the increase in the queue. That is, considering that $\Delta_a = C^p_{respa} + t_{ID1a} + t_{ID1a+} + C^c_{reqa} + t_{rt}$ and $\Delta_b = C^p_{respb} + t_{ID1b} + C^c_{reqb} + t_{ID1b}$, then:

$$t_{ID1a+} \geq \left(C^p_{respb} - C^p_{respa}\right) + \left(C^c_{reqb} - C^c_{reqa}\right) + \left(2 \cdot t_{ID1b} - t_{ID1a}\right) - t_{rt} \quad (1)$$

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

3.1.3 Computation of the inserted idle time after issuing an unacknowledged request

The condition expressed in (1) is just related to acknowledged request frames, though. The case of a sequence of non-acknowledged request (or token) frames must also be analysed. Figure 2 shows a sequence of unacknowledged requests and the variables that are necessary to evaluate this second idle time.

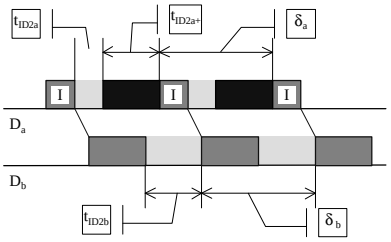

Figure 2: Inserting additional idle time (unacknowledged request sequence)

In this case, d_a should be greater or equal to d_b. That is, considering $d_a = C_{reqa} + t_{ID2a} + t_{ID2a+}$ and $d_b = C_{reqb} + t_{ID2b}$, then:

$$t_{ID2a+} \geq \left(C_{reqb} - C_{reqa}\right) + \left(t_{ID2b} - t_{ID2a}\right) \quad (2)$$

Similarly, the different DLL unacknowledged request PDUs lengths for that master must be known, in order to compute t_{ID2+}.

Finally, assuming only one register for T_{ID1} and one register for T_{ID2}, there is the need to merge both the "minimum" idle time with the inserted idle time in one variable, i.e.:

$$t'_{ID1a} = t_{ID1a} + t_{ID1a+} \wedge t'_{ID2a} = t_{ID2a} + t_{ID2a+}$$

Or, in bit times:

$$T'_{ID1a} = T_{ID1a} + T_{ID1a+} \wedge T'_{ID2a} = T_{ID2a} + T_{ID2a+}$$

We are considering also that a master station will insert T'_{ID2} after receiving a token frame[2]. This is true since a sequence of token transmissions from stations that have nothing to transmit may also cause traffic congestion in the linking devices.

Concerning the idle time used by the linking devices when relaying frames, they only insert the "conventional" idle times, i.e. T_{ID1} and T_{ID2}. In order to avoid the need for the linking devices to decode the DLL PDU (to know if it is a acknowledged or unacknowledged request), both "conventional idle times should be set to the same value (i.e. T_{ID1}=T_{ID2}=T_{ID}).

3.2 System Turnaround Time

In order to evaluate the duration of a transaction, it is necessary to determine the system turnaround time associated to that transaction, that is, the time interval between the end of the request's transmission and the beginning of the response's reception. To clarify this definition, consider the network topology depicted in Fig. 2.

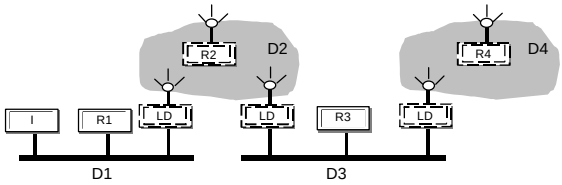

Figure 3: Example of a hybrid wired/wireless network topology

Assume that D1 and D3 have the same type of PhL (PDU format and bit rate) and D2 and D4 have another type of PhL. Fig. 4 depicts the timing diagram for the longest (no queuing delays) transaction between I and R4. Both request and response frames must be relayed by 3 linking devices.

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

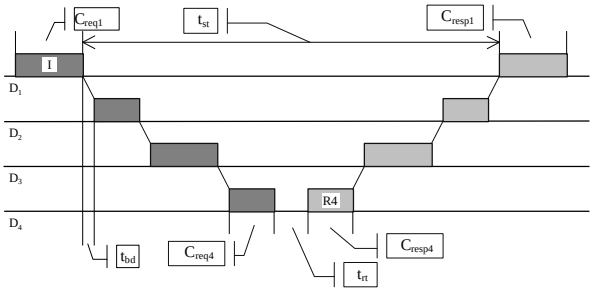

Figure 4: Timing diagram for transaction between I and R4

In Fig. 4, t_{bd} represents the buffering delay of the linking devices and is assumed equal in all the linking devices) and the responder turnaround time (t_{rt}) is assumed to be constant for every station. Clearly, if for a transaction the responder belongs to the same domain as the initiator, the system turnaround time - t_{rt} – equals the responder turnaround time. Oppositely, when there is one or more linking devices between initiator and responder, the system turnaround time will increase. This is the case depicted in Fig. 4. Nevertheless, the timing diagram is simplified, since the request frame may be delayed by the previous frame (either unacknowledged request or response), if the duration of the previous frame is higher than the duration of the request frame. In this case, the queuing delay in all linking devices involved in the transaction may be computed, but introduces a significant amount of complexity to the computation of the system turnaround time. Instead, we will 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 of message transactions, since there is no need to compute the queuing delay in linking devices.

For the previous scenario (Fig. 4), the worst-case system reaction 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:

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

Where i represents the domains involved in the transaction and n represents the domain of the responder. The duration of each PhL PDU depends on the PhL parameters defined in Table 1, for each domain.

3.3 Computing the duration of transactions

Finally, the duration of a transaction can be easily evaluated summing its components (refer to Fig. 5 for the case of an acknowledge transaction). 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}). 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}).

3.3.1 Duration of an acknowledged transaction

The duration of a request/response transaction (Fig. 5) can be evaluated as follows:

$$C_{ack} = C_{req1} + t_{st} + C_{resp1} + t'_{ID1} = \\ = \sum_{i=1}^{n} \left(C_{reqi} + C_{respi}\right) + (n-1) \cdot 2 \cdot t_{bd} + t_{rt} + t'_{ID1}$$

Remember that since we opted for the pessimistic approach (no queuing delays in linking devices), 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 Fig. 5, for a transaction between I and R3.

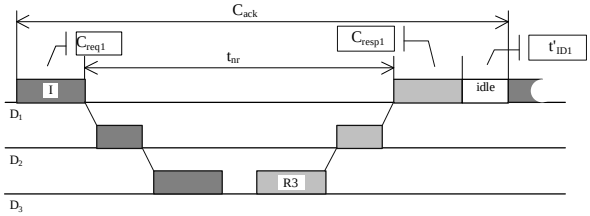

Figure 5: Duration of a Transaction

3.3.2 Duration of a unacknowledged transaction

The duration of an unacknowledged transaction does not depends of the number of linking devices between the initiator and the responder, since there is no need to wait for any response/acknowledge to proceed. Such duration can thus be evaluated as follows:

$$C_{unk} = C_{req} + t'_{ID2}$$

4. The RFieldbus System

The RFieldbus System is being specified in the scope of the European Union Project IST-1999-11316 RFieldbus - High Performance Wireless Fieldbus in Industrial Multimedia-Related Environment [7]. Within this project, Profibus was chosen as the fieldbus platform. Essentially, extensions to the current Profibus standard are being developed in order to provide Profibus with wireless, mobility and industrial-multimedia capabilities. In fact, providing these extensions means fulfilling strong requirements, namely to encompass the communication between wired (currently available) and wireless/mobile devices and to support real-time control traffic and multimedia traffic in the same network.

4.1 RFieldbus network topology

The RFieldbus network topology [8] is exemplified in Figure 6.

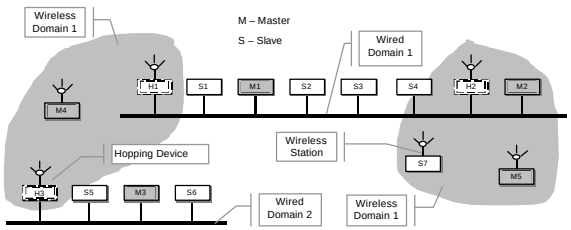

Figure 6: RFieldbus network topology and components

A domain consists of a set of stations communicating between them via a shared communication channel. No registering mechanisms are needed since we assume that wireless domains operate in different radio channels. It is also important to note that inter-domain mobility is supported if in each wireless domain messages are relayed through a base station (with up-link and down-link channels instead of direct communication between the wireless nodes) and mobile stations are able to perform channel assessment and channel switching.

4.2 Wired and wireless PhL parameters

The wired domains are expected to have a bit rate of 1,5 Mbit/s, and the asynchronous version of Profibus (RS485). Each PhL PDU consists of a number of characters – the UART characters, that are composed of 11 bits each (8+3). The wired PhL adds no overhead. When relaying a Profibus PhL PDU to a wireless domain, the linking device removes every extra 3 bits and encapsulates the entire DLL PDU in the data part of the wireless PDU, adding a specific header. Also, there is the need to insert a preamble and a start frame delimiter (SFD) to the wireless PhL PDU, in order to allow its correct reception. A 2 Mbit/s bit rate is envisaged for the wireless communications.

Considering that in RFieldbus there is a 10 bytes header plus a 53 µs preamble and SFD (resulting in a total overhead of 186 bit times, at 2 Mbit/s) in the wireless PhL PDU, and referring to Table 1, we get:

Wired domain	Wireless domain
k_{wr}=11 bits/char	k_{wl}=8 bits/char
l_{wl+}=0 bits	l_{wl+}=186 bits
r_{wl}=1,5 Mbit/s	r_{wl}=2 Mbit/s

Table 2: Parameter values in the RFieldbus system

Thus, the duration of wired and wireless PhL PDUs can be computed as:

$$C_{wr} = \frac{L \cdot 11}{1{,}5} = \frac{22}{3} \cdot L \ \ (ms) \ \wedge \ C_{wl} = \frac{L \cdot 8 + 186}{2} = 4 \cdot L + 93 \ \ (ms)$$

Table 3 presents the PhL PDU duration for some DLL PDU lengths:

PDU Type	L (chars)	C_{wr} (µs)	C_{wl} (µs)
Short acknowledge	1	7.3	97
Token	3	22	105
Fixed length no data	6	44	117
50 data octets	59	432.7	329
100 data octets	109	799.3	529
150 data octets	159	1166	729
246 data octets	255	1870	1113

Table 3: Frame length and duration

From Table 3, it is clear that short frames have a longer duration in wireless domains while long frames take longer to transmit in wired domains.

5. Computing the Duration of a RFieldbus Transaction

To be able to compute the duration of transactions within the RFieldbus context, there is an additional set of network parameters that must be set:

Description	Symbol	Value
Responder turnaround time	t_{rr}	100 µs
Buffering delay	t_{bd}	25 µs
Idle time after receiving frame (default value)	T_{ID1}	50 bit times
Idle time after transmitting frame (default value)	T_{ID2}	50 bit times

Table 4: Additional network parameters

Throughout the rest of this document, we denote the number of linking devices that a transaction must cross as *n*.

5.1 Idle Time Parameters

To compute the idle time parameters, there is the need to define the maximum/minimum lengths for Profibus FDL request/response frames. Therefore, the following additional set of parameters is defined:

Description	Symbol	Value
Maximum length of Profibus FDL request	L_{req}^{max}	-
Maximum length of Profibus FDL response	L_{resp}^{max}	-
Minimum length of Profibus FDL request	L_{req}^{min}	6 octets
Minimum length of Profibus FDL response	L_{resp}^{min}	1 octets

Table 5: Additional Timing Parameters

Considering the set of network parameters, and knowing that the smallest PDU lengths are as defined in and assuming that $L_{req}^{max} = L_{resp}^{max} = L^{max}$, the resulting idle time values are presented in Table 6.

L^{max} **(octets)**	**59**	**109**	**159**	**255**
t'_{ID1WL} (µs)	174	507,3	840,7	1481
t'_{ID1WR} (µs)	112,7	112,7	112,7	112,7
t'_{ID2WL} (µs)	137	303,7	470,3	790,3
t'_{ID2WR} (µs)	108	108	108	108

Table 6: Idle Time Values

From Table 6 it is clear that, for a wired initiator, the maximum length of the DLL request and response PDUs does not have any effect on the evaluated values for the idle time parameters. The reason is that for the case of wired initiators, the worst trade-off between the length of the wired frame and the related wireless frame is for the case of the *smaller frames*, and thus the length of the longest frames does not affect the calculations.

For the case of a wireless initiator, the evaluated idle time parameters are roughly proportional to the length of the longest frames, as expected.

5.2 Duration of RFieldbus Transactions

Finally, the duration of RFieldbus transactions can be easily evaluated, for the specific case of each transaction path. That is, for each transaction we need to consider the specific number of linking devices crossed by the transaction and also the characteristics of each of the communication mediums. In this subsection, we present several examples of both acknowledged and unacknowledged transactions, considering 4 different DLL request/response PDU maximum lengths.

5.2.1 Duration of acknowledged transactions

Again, it is considered that any request frame is always longer than the previous frame on the network (previous response frame or unacknowledged request frame), in order to avoid the need to compute the queuing delay in linking devices.

The duration of a request/response transaction can be computed as follows:

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

Now, three tables supplying values for the duration of a RFieldbus transaction are presented. These correspond to maximum DLL PDU lengths of 255, 109 and 59 octets. In each table, we can get the (ceiling) duration of a transaction depending on both the transaction path (which wired and wireless domains are between the initiator and the responder) and the length of the DLL response PDU. Let us first consider the maximum DLL PDU length that is allowed in Profibus, i.e., for $L_{req} = L_{req}^{max} = L_{resp}^{max} = 255$ octets:

	Response length				
Transaction Path	**255 octets**	**159 octets**	**109 octets**	**59 octets**	**1 octet**
WR	3953	3249	2882	2516	2090
WR/WL	6229	5141	4574	4008	3350
WR/WL/WR	10019	8227	7294	6317	5278
WR/WL/WR/WL	12295	10119	8986	7853	6538
WL/WL	6083	5315	4915	4515	4051
WL/WR	7597	6509	5943	5376	4719
WL/WR/WL	10630	9158	8392	7625	6736
WL/WR/WL/WR	13663	11487	10354	9221	7906

Table 7: Transaction Duration Values 1 (µs)

Considering $L_{req} = L_{req}^{max} = L_{resp}^{max} = 109$ octets:

	Response length		
Transaction Path	**109 octets**	**59 octets**	**1 octet**
WR	1812	1445	1020
WR/WL	2920	2353	1696
WR/WL/WR	4568	3635	2552
WR/WL/WR/WL	5676	4543	3228
WL/WL	2774	2374	1910
WL/WR	3314	2748	2090
WL/WR/WL	4422	3656	2766
WL/WR/WL/WR	6071	4938	3623

Table 8: Transaction Duration Values 2 (µs)

Finally, for $L_{req} = L_{req}^{max} = L_{resp}^{max} = 59$ octets:

	Response length	
Transaction Path	**59 octets**	**1 octet**
WR	1079	653
WR/WL	1787	1129
WR/WL/WR	2702	1619
WR/WL/WR/WL	3410	2095
WL/WL	1640	1176
WL/WR	1848	1190
WL/WR/WL	2556	1666
WL/WR/WL/WR	3471	2156

Table 9: Transaction Duration Values 3 (µs)

5.2.2 Duration of unacknowledged transactions

The duration of an unacknowledged request (SDN) can be computed as follows:

$$C_{unk} = C_{req} + t'_{ID2}$$

Here, the duration does not depend on the path until the addressee, but only on where the initiator is located (wired or wireless domain). Similarly to the acknowledged request case, we consider three different maximum lengths of the DLL request/response PDU (255, 109 and 59 octets) that is.

Therefore, for $L_{req} = L_{req}^{max} = L_{resp}^{max} = 255$ octets :

	SDN length				
Master Type	**255 octets**	**159 octets**	**109 octets**	**59 octets**	**6 octets**
WR	1978	1274	908	541	152
WL	1904	1520	1320	1120	908

Table 10: SDN duration values 1 (μs)

For $L_{req} = L_{req}^{max} = L_{resp}^{max} = 109$ octets :

	SDN length		
Master Type	**109 octets**	**59 octets**	**6 octets**
WR	908	541	152
WL	833	633	421

Table 11: SDN duration values 2 (μs)

Finally, considering $L_{req} = L_{req}^{max} = L_{resp}^{max} = 59$ octets :

	SDN length	
Master Type	**59 octets**	**6 octets**
WR	541	152
WL	466	254

Table 12: SDN duration values 3 (μs)

As can be seen from the tables, the duration of unacknowledged requests sent by wired master stations does not depend on the maximum DLL PDU length in the network. For instance, the duration of a 109 octet length DLL PDU sent by a wired master is the same (dashed cells), when the maximum length DLL PDU is 255 and when it is 109 octets. This is true since the idle time is computed taking into account the shortest unacknowledged request frames.

6. Conclusion

In order to guarantee the real-time behaviour of a distributed system, it is necessary 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 linking devices may exist between the two communicating peers.

This paper presented a way to compute the duration of message transactions in hybrid wired/wireless token-passing fieldbus networks working in a broadcast fashion. We explained the concepts of inserted idle time and system turnaround time and how these parameters can be calculated, since they are components of the duration of a transaction. Then, we introduced the most relevant characteristics of a hybrid wired/wireless system – the RFieldbus system – that is under development in the scope of the IST programme. Finally, we defined default RFieldbus parameters such as physical layer PDU formats and bit rates, in order to get some numerical figures for the duration of RFieldbus transactions.

7. References

[1] Tovar, E. and Vasques, F., “Cycle Time Properties of the Profibus Timed Token Protocol”, in Computer Communications, Elsevier Science, 22(13), pp. 1206-1216, 1999.

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

[3] Raja, P., Ruiz, L., Decotignie, J.D., “On the Necessary Real-Time Conditions for the Producer-Distributor-Consumer Model”, in Proc. of the 1st IEEE Workshop on Factory Communication Systems (WFCS’95), Leysin, Switzerland, 1995.

[4] Tindell, K., Hansson, H., Wellings, A., “Analysing Real Time Communications: Controller Area Network (CAN)”, in Proc. of the IEEE Real-Time Systems Symposium, pp. 259-263, December 1994.

[5] Alves, M., Tovar, E., Vasques, F., “On the Adaptation of Broadcast Transactions in Token-Passing Fieldbus Networks with Heterogeneous Transmission Media”, HURRAY-TR-0120, May 2001.

[6] “General Purpose Field Communication System, Volume 2” – Profibus, European Norm EN 50170, 1996.

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

[8] RFieldbus Deliverable D1.3, “General System Architecture of the RFieldbus”, Technical Report, June 2000.