Polling-based MAC Protocols for Improving Real-Time Performance in a Wireless PROFIBUS

Accepted for Publication in IEEE Transactions on Industrial Electronics

Andreas Willig, *Member, IEEE*

Abstract— **The idea of having a wireless PROFIBUS is appealing, since this can bring benefits like reduced cabling need and mobile stations to the factory floor. But unfortunately wireless transmission is error-prone, which affects the timeliness and reliability behavior users expect from a fieldbus system (hard real-time). In this paper we compare two different approaches for the medium access control and link layer of a wireless PROFIBUS system with respect to their so-called real-time performance in the presence of transmission errors. Specifically, we compare the existing PROFIBUS MAC and link layer protocol with a simple round-robin protocol. It shows up that round-robin delivers significantly better real-time performance than the PROFIBUS protocol under bursty error conditions. In a second step we propose three add-ons to round-robin and we show that they further increase the real-time performance of round-robin. The add-ons take certain characteristics of the wireless medium into account.**

Keywords— **PROFIBUS, Wireless PROFIBUS, Real-Time Performance, MAC Protocols, Polling**

I. Introduction

THE idea of having a wireless fieldbus system is appealing, since wireless technology allows for mobile stations and reduces the need for cabling, which is beneficial in terms of plant setup times and costs. It becomes more easy to set up prototype plants or to reconfigure existing plants. Furthermore, a wireless fieldbus system is a natural choice for applications involving mobile subsystems or in areas with explosive chemicals. Unfortunately, wireless transmission is known to be error-prone and to have time-variable error characteristics; link outages in the range of tens of seconds can be observed. An immediate consequence of this is that wireless transmission is not well suited for industrial applications with very high reliability requirements, e.g., where a deadline miss can cause harm to human life or expensive equipment. For applications where a deadline miss is not a catastrophe but just annoying or undesirable, there is a tradeoff between the frequency of deadline misses and the benefits obtainable with wireless transmission. An example for this class of applications is automated material handling [1]. The fewer deadline misses, the better. More precisely: the percentage of important high priority messages which can be transmitted reliably (i.e., acknowledged) within a pre-specified timebound should be as high as possible, e.g., $> 99.x\%$, even at the cost of other performance measures like throughput or mean delay. In this paper we define this criterion more precisely as a set of performance measures, called the *real-time performance measures*. They provide a means for capturing simultaneously the timeliness and reliability capabilities of a wireless fieldbus system.

A very popular fieldbus system is the widely used and standardized PROFIBUS (PROcess FIeld BUS). It is designed to deliver real-time services in harsh, industrial environments. Its specification covers layers 1 (physical, PHY), 2 (link layer and medium access control, MAC) and 7 (application) of the OSI reference model. On the MAC and link layer, which in fact is a key layer for implementing timing and reliability guarantees, it employs a token passing protocol on top of a broadcast medium.

There are several ways to create a wireless PROFIBUS system. A natural approach is to replace the given PROFIBUS PHY with a wireless PHY, while leaving the remaining protocol stack unmodified. This approach is attractive, since design efforts are only required for the PHY. Furthermore, integrating wireless and wired stations in a single local area network (LAN) can be carried out with simple repeater- or bridge-like devices. One can further reduce the design efforts by choosing an already available, mature and successful PHY like the standardized IEEE 802.11 wireless local area network (WLAN) PHY. However, the solution proposed in this paper is different. We demonstrate that for an IEEE 802.11 type PHY and for bursty error characteristics (as they are typical for wireless channels) the real-time performance of the PROFIBUS protocol is unsatisfactory. Polling-based protocols can achieve significantly better real-time performance under the same conditions. Therefore, we advocate another general approach towards a wireless PROFIBUS: for the wireless stations the PROFIBUS MAC- and link layer protocol is replaced by a specifically tailored protocol, working on top of an IEEE 802.11 PHY and offering the same link layer interface to upper layers as the PROFIBUS protocol does. The protocol stack for the wired stations is not changed. For integrating wired and wireless stations into a single PROFIBUS LAN a special coupling element is introduced. We call such a system an *integrated PROFIBUS*. In this paper we propose polling-based protocols for the wireless part. Since the link layer interface does not change, the same application layer software can be used on both wired and wireless stations.

This approach opens a large design space. In this paper we focus on the wireless MAC- and link layer protocol. The point of departure is a simple round-robin protocol, namely the k-limited round-robin protocol. A key contribution of this paper is to show that this protocol already outperforms the PROFIBUS MAC- and link layer protocol significantly w.r.t. real-time performance under bursty er-

This work was sponsored by the Deutsche Forschungsgemeinschaft (DFG).

A. Willig is with the Telecommunication Networks Group, Department of Electrical Engineering, Technical University Berlin, Germany. E-mail: willig@ee.tu-berlin.de .

rors, as they are typical for wireless channels. In a second step, we show that the real-time performance delivered by round-robin can also be improved significantly. We propose three add-ons to round-robin, namely the simple data relaying (SDR), simple poll relaying (SPR) and priority restriction (PR) schemes. While the PR scheme affects bandwidth distribution between wireless stations, the relaying schemes take advantage of the spatial diversity of the wireless channel. In our opinion the benefits of polling-based protocols in terms of real-time performance outweigh the additional complexities needed for coupling wireless and wired stations into a single LAN.

This paper is structured as follows: in the next Section II we briefly introduce some background on the IEEE 802.11 WLAN technology and its error behavior. In Section III we describe the main characteristics of the PROFIBUS fieldbus system and sketch different approaches for constructing an integrated PROFIBUS. In Section IV we describe the polling-based protocols used in this paper. The real-time performance measures are defined in Section V. They are used in Section VI to compare the PROFIBUS protocol with the k-limited round-robin protocol, and in a second step, to compare the three add-ons to the basic round-robin protocol. In Section VII we give an overview on related work and in Section VIII we conclude the paper.

II. WLAN Technology and Wireless Channels

The notion of wireless local area network (WLAN) systems subsumes systems where data is with radio or infrared waves over short distances and in a packet-switched mode. Most radio-based WLAN technologies use frequencies below 6 GHz. Of special importance are license-free bands, e.g., the Industrial, Scientific and Medical (ISM) bands. In particular, a band of 83.5 MHz of spectrum between 2.4 GHz and 2.4835 GHz is granted by the FCC (Federal Communications Commission) for public use.

An attractive base technology for creating wireless fieldbuses is provided by the IEEE 802.11 WLAN standard [2], which operates in the 2.4 GHz and 5 GHz ISM bands. Specifically the 802.11 direct sequence spread spectrum (DSSS) PHY operating in the 2.4 GHz band is widely deployed; for this standard and its 11 MBit/s extension IEEE 802.11b [3] many components are available off-the-shelf. In the 5 GHz band orthogonal frequency division multiplexing is employed and commercial products start to become available.

A. The IEEE 802.11 DSSS PHY

The DSSS PHY specified in [2] and [3] works on a single frequency band. The unit of transmission is a *packet*. A packet consists of a *PHY preamble*, a *PHY header* and a *data part*, which actually contains the MAC-packet, including all MAC headers and the user data. The PHY preamble is used by the receiver to acquire bit synchronization. The PHY header indicates the packet length and the modulation scheme used in the data part. Both PHY preamble and PHY header are transmitted with 1 MBit/s BPSK modulation. The PHY preamble consists of 128 bits, the PHY header of 64 bits. Hence, the overall PHY overhead is $128 + 64 = 192$ bits, which amounts to 192μs. In the packet's data part four different modulation schemes can be used: BPSK modulation with a data rate of 1 MBit/s, QPSK with 2 MBit/s , and CCK with 5.5 and 11 MBit/s, respectively.

B. Error Behavior of an 802.11 DSSS Wireless Link

Transmission errors on a wireless link[1] can occur due to several mechanisms [4], [5]. Path loss and attenuation on obstacles lead to *slow fading*. Reflection, diffraction, refraction and scattering can cause transmission on multiple paths, which in turn results in *multipath fading* and *intersymbol interference*. A transmitted signal can be further distorted by adjacent channel interference, co-channel interference, thermal or man-made noise. Finally, the transmitter and receiver circuitry can also create errors. While thermal noise is present in almost any communications channel, fast fading and slow fading are specific for wireless transmission. Man-made noise in industrial environments can have several sources, e.g., remote controls, motors, or microwave ovens.

However, the MAC- and link layer instances of a wireless station usually do not cope directly with transmitted waveforms. Instead, they are exposed to the digital bits and packets delivered by the PHY. In [6] the results of bit- and packet-level measurements taken in an industrial environment are reported. It showed up that beneath bit errors the phenomenon of *packet losses* is also of importance. Packet losses occur when the receiver fails to acquire bit synchronization, i.e. when the PHY preamble is not received properly. They cannot be avoided by adding redundancy to a packet (e.g., forward error correction codes), since this affects only the data part of a packet, not its PHY preamble or header.

Both bit error rates and packet loss rates vary strongly over time. We illustrate this for packet losses. Specifically, in Figure 1 we show some results obtained for a measurement spanning more than four hours. The measurement is subdivided into 180 so-called *traces*, in each trace 20000 packets of fixed size were sent. The first 90 traces are taken with PHY data scrambling, the remaining 90 traces without. All other parameters were identical, which allows to compare the traces. In the figure we show the packet loss rate per trace. It can be seen that packet losses show a "seasonal" behavior with periods of many and strongly varying losses and other periods with almost no losses. The time-scales involved are in the order of minutes and hours.

Both packet losses and bit errors are *bursty*. Bursts of bit errors alternate with periods of error-free operation and bursts of packet losses alternate with packet loss free periods. All burst length distributions are highly variable. For example, when we concatenate the packet loss information of all traces shown in Figure 1 and determine the distributions of the lengths of packet loss bursts and packet loss-free periods, it shows up that these distributions have

[1]The notion *wireless link* is used as an abstraction for the ensemble of the radio modems and the actual wireless channel.

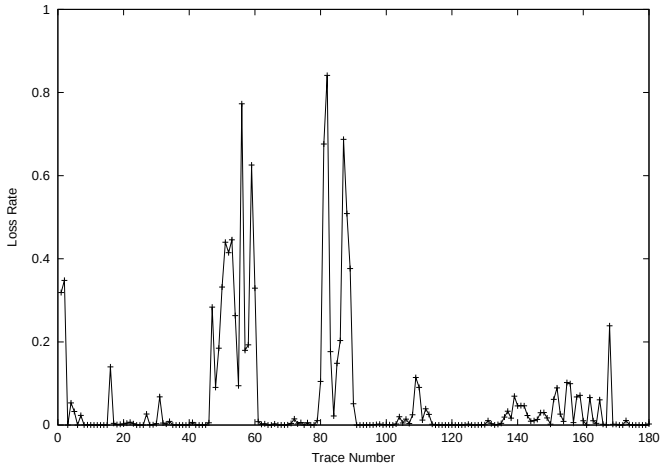


Fig. 1. Rates of lost packets for **longterm1** measurement ($>$ 4 hours)

a high variability. The coefficient of variation (CoV) of the packet loss bursts is ≈ 17, the packet loss-free periods have a CoV of ≈ 20. Both distributions have much mass on short burst or period lengths, but occasionally very long bursts/periods can be observed: the longest packet loss burst lasts for more than 14000 packets, the longest packet loss-free period was of more than 100000 packets length. We believe that even with newer modem designs the problems of time-variability and burstiness will not disappear, since they are grounded in inherent characteristics of the wireless channel itself (path loss, multipath fading, intersymbol interference, interferers).

One may suspect that bit errors and packet losses are correlated, but our traces show no evidence for this.

The wave propagation effects result in *spatial diversity*: since the propagation environments between a transmitting station A and two receiving stations B and C at different locations depend on the respective distances, number of paths and their respective attenuation, etc., likely B and C have different propagation environments. Hence, they experience a different error behavior. It might happen that a frame sent by A is properly received by B, but arrives in error at C. This property is exploited by the relaying schemes proposed in Section IV.

III. PROFIBUS and Wireless PROFIBUS

In this section we first give some brief background information on the PROFIBUS. Following this, we depict a general framework for integrating wired and wireless stations into a single logical fieldbus LAN.

A. PROFIBUS

The PROFIBUS is a well-known, standardized and widely used fieldbus [7], [8], [9]. We focus on PROFIBUS-FMS, which allows for multiple master stations and can be used for interconnecting several intelligent controllers. The standard defines several PHY's, however, the RS-485 PHY is most commonly used. It is a broadcast medium and supports physical tree topologies.

The combined MAC- and link layer is called *fieldbus data link* (FDL) layer. On the MAC level two approaches are combined: a "request/response"-type of protocol for exchange of user data or management frames, and a token-passing protocol for arbitrating the right to initiate frame transmissions. The token is passed along a *logical ring*, which is formed by ascending station addresses. Only ring members receive the token and are allowed to initiate data transmissions. For passing the token a station a sends a *token frame* to its successor in the ring b. If b receives the token, it has to start immediately with generating frames, otherwise a repeats the token. Station a performs at most three trials to pass the token. If these are unsuccessful, a assumes that station b is dead and passes the token to b's successor in the logical ring. Station b has to be re-included later by the ring maintenance mechanism.

Besides token passing every ring member carries out additional tasks for ring maintenance: detection and repair of lost tokens, including new stations into the ring, etc. The current ring members are responsible for including new stations into the ring. A ring member polls approximately every $gap\_factor \cdot T_{TTRT}$ seconds the address space between its own address and the address of its logical ring successor. If a new station is found, it is included into the ring. For detecting lost tokens a station is required to monitor the medium for times without activity. More specifically, upon every start of a medium idle period a station starts a so-called *timeout timer*. The timeout value depends on the stations PROFIBUS address. If the timer expires, the station restarts the logical ring by starting transmissions and passing the token to its successor.

The link layer offers three acknowledged (semi-reliable) and one unacknowledged service to its users via the *link layer interface*. A service is requested by the service user with a *request primitive*. As soon as the outcome of the request is known, the service user is notified with a *confirmation primitive*. For the acknowledged services the outcome is determined with the help of MAC layer immediate acknowledgement frames (ack frames). In the SDA service (Send Data with Acknowledgement) the ack frame carries no answer data, in the other acknowledged service types the ack may carry data. In the PROFIBUS two priority classes are distinguished: high and low, the low priority class is further subdivided into asynchronous and cyclic data. Upon every token arrival a station computes its token holding time T_{HT} according to a modified version of the timed-token protocol: a station subtracts the measured token rotation time T_{RT} from the configured *target token rotation time* T_{TTRT} and sets the token holding timer to the value $T_{HT} = T_{TTRT} - T_{RT}$. If $T_{HT} > 0$, then the station is allowed to initiate data transfers. However, even in the case of late tokens ($T_{HT} < 0$) it is allowed to handle one high priority service request. After a service request has been finished, the station checks the token holding timer. If there is still time available, the station is allowed to start the next service request. The station handles the high priority requests first, then proceeds with the low priority re-

quests. Once the low priority service has started, it cannot be interrupted by newly arriving high priority requests. A service request is processed according to a variant of the alternating bit protocol (ABP), with one separate protocol instance per target station. The maximum number of retransmissions is bounded by the parameter *max_retry*. A service request is handled in an atomic way, i.e., it cannot be interrupted by other requests or by passing the token to another station.

B. Wireless Fieldbus and Wireless PROFIBUS

In an integrated PROFIBUS wired and wireless stations will be combined in a single LAN such that the existing wired stations see no difference between wired and wireless stations. Hence, wired stations can run the same protocol stack. In the following we call a station with a wired transceiver a *wired station*, accordingly a station with a wireless transceiver a *wireless station*.

The integration can be done in several ways. In the *wireless repeater approach* all stations are attached to a cable, hence, are wired stations. A piece of cable is replaced by a wireless link, no station needs to be aware of this. In the *wireless bridging approach* the integration of wired and wireless stations is done at the physical layer. A bridge-like device translates the different framing rules used on wired and wireless media. The MAC and link-layer protocol and all application layer protocols are the same for both wired and wireless stations. In the *integrated scenario* the integration takes place at the MAC- and data link-layer, with the original PROFIBUS protocol on the wired side and a specifically tailored protocol on the wireless side. The wireless protocol can take the wireless channel characteristics explicitly into account. However, both protocols offer the same link layer interface and are not distinguishable for upper layers. Hence, application layer software needs not to be modified.[2] Finally, in the *wireless gateway approach* the integration is done in the application layer. Of course, these approaches can be mixed.

In this paper we advocate the integrated scenario. Both the wireless repeater and the wireless bridging approach require the transport of PROFIBUS MAC frames over a wireless medium, which should be avoided due to problems with the stability of the logical ring, see Section VI. On the other hand, with the wireless gateway approach a tight coupling of the timing behavior between wired and wireless stations is hard to achieve.

The integrated scenario operates as follows: the existing PROFIBUS protocol runs on the wired segment, while on the wireless segment a different MAC- and link layer protocol is operated. No explicit token-passing should take place on the wireless segment, nor should any token-passing related PROFIBUS frames appear there. Using different protocols on both sides puts special responsibility on the intermediate station, the *base station/interworking unit* (BS-IWU): it has to translate between both sides. More specifically, it has to act on the wired segment on behalf of the wireless stations such that the wired stations encounter no difference and have a view on the logical ring that includes the wireless stations in a consistent manner. To achieve this, the BS-IWU performs so-called *mimicry functions*. As an example, the BS-IWU generates token frames on the wired segment which include the wireless stations PROFIBUS addresses, it includes new members into the logical ring on behalf of the wireless stations, etc. Stated differently: the BS-IWU provides a *virtual ring extension* to the wired stations.

[2]It can be argued, however, that applications can be improved if they are *channel-aware*, i.e., if they adapt to varying channel conditions and outages. Hence, the link-layer interface of wireless stations should be enriched with appropriate signals. An example is an adaptive middleware [10].

IV. Polling-based Protocols

In this section we introduce a set of MAC mechanisms and a link layer protocols, targeted to serve as a replacement of the existing PROFIBUS protocol. The class of polling-based protocols is attractive due to its deterministic operation. In polling-based protocols a central station polls a number of terminals according to a certain schedule. In this paper we will denote the central station as *base station* (BS) and the terminals as *wireless terminals* (WT).

We use a k-limited round-robin variant as the basic protocol (abbreviated as rr-k). Furthermore, we propose three different add-ons to rr-k, which can be switched on and off independently of each other. Two of these modifications, the so-called *relaying approaches*, take advantage of the spatial diversity of the wireless error behavior. As discussed in Section II-B there can be very long periods of sustained packet losses on the channel $C(a, b)$ between stations a and b. One possibility to overcome this problem is to let another station c relay the communication between a and b, hoping that the channels $C(a, c)$ and $C(c, b)$ are currently in a good state. In both proposed schemes the BS is the main actor. A channel $C(a, b)$ between two stations (WT or BS) a and b is an abstraction of the error behavior: a packet transmitted over this channel from a to b is subjected to the operation of an error model. Different channels can have separate error models.

We consider a wireless-only scenario with one base station (BS) and a number N of wireless terminals (WT), together constituting a *cell*. All stations are equipped with an IEEE 802.11 DSSS-compliant PHY. All WT's are considered to be active stations in the sense that they require the right to initiate transmissions from time to time. The BS maintains a list of WT's which are members of its cell and which are to be polled. To become a member of this list, every WT has to register itself once at the BS, using special registration frames on a separate (physical or logical) channel. It is important to note that with successful one-time registration a WT is a cell member for potentially indefinitely long time. This is in contrast to the PROFIBUS protocol, where a station can loose its LAN membership in every token cycle. Hence, the permanent danger of getting lost from the LAN is eliminated in the

polling-based approach.[3] A more thorough description of the system model and the polling-based protocols is given [11, chap. 7].

A. k-limited Round-Robin

The k-limited round-robin protocol belongs to the class of cyclic polling protocols (as opposed to Markovian or table-based protocols [12, chap. 9]). The WT's are served in round-robin fashion, and the BS allocates a maximum of $k \in \mathbb{N}$ contiguous slots to a single WT before proceeding to the next one. The value k is called *round-robin bound*.

A *data slot* accommodates a `data` frame and (hopefully) an acknowledgement (`ack`) frame. If a WT has no data to send, it simply keeps quiet. This is used by the BS to determine that a WT has finished its work for this poll cycle, and to proceed with the next WT. The BS uses the absence of a signal on the medium for a certain time as a criterion. As a special rule, the BS keeps track of WT a's data slots. Specifically, it can determine the end of a `data` frame by the presence of a short medium idle period. It can also determine whether the recipient generates an `ack` or not by performing carrier sensing at an appropriate time instant. If the BS detects the lack of an `ack` frame, it can take control over the channel by starting a transmission. In this case, WT a defers and waits for the next time being polled.

The decision to keep a quiet in case of empty queues is for efficiency reasons. The protocols proposed in the following sections would work in the same way if a announces empty queues with special frames instead.

B. Simple Data-Relaying (SDR)

Let us consider the case that WT a sends a `data` frame to WT b over the channel $C(a, b)$. The packet can get lost or WT b detects bit errors. In these cases b does not transmit an `ack` frame. The BS and WT a can determine this by performing carrier sensing at an appropriate time after WT a finished its `data` frame.

However, when the channel $C(a, z)$ from a to the BS z is not distorted, BS z has successfully received and stored the `data` frame. In this case and if the `data` frame was of high priority, the BS sends the `data` frame on the channel $C(z, b)$ to b immediately after it recognized the missing `ack` frame. If b responds to this with an `ack` frame, the BS tries to receive it and to send the `ack` frame to a. Hence, a may receive the same `ack` frame two times, once on the channel $C(b, a)$ and once on the channel $C(z, a)$. This requires the ability of a to detect and ignore duplicate `ack` frames. Furthermore, a does not count z's `data` frame as a separate retransmission trial. After this operation the BS polls a again with the remaining number of slots available to a.

This approach is denoted as *simple data relaying (SDR)*. The term "simple" comes from the restriction to let only the BS play the role of a data relayer. If this restriction is removed, some synchronization between potential relayers is needed to avoid collisions, e.g., by setting timeouts corresponding to a stations address.

[3]Clearly, when considering mobility and multiple wireless segments in a single integrated PROFIBUS LAN, a more sophisticated registration/deregistration scheme is needed. But nonetheless, these schemes can be designed such that LAN membership is not a critical issue.

C. Simple Poll Relaying (SPR)

A critical issue not resolved by the SDR scheme is the case of a heavily distorted channel between the BS z and a WT a. This may lead to the situation that a does not receive a `poll` frame for long time. Since WT a does not transmit anything in case of empty queues, the BS cannot determine whether a's queues are empty or the channel to a is distorted. To exclude bad channel conditions, the BS insists on getting a short answer from time to time. This is done using a special kind of `poll` frame, which urges a to start a data slot or to send a `null` frame in case of empty request queues. However, if WT a keeps quiet for more than T_{Quiet} seconds despite the special `poll` frames, the BS assumes that its channel to a is in a bad state. The goal to poll a despite the bad channel $C(z, a)$ can be achieved with the help of other WT's. More specifically, the BS selects a station u (the *poll relayer*), which acts temporarily on behalf of BS z and polls WT a on behalf of the BS, urging a to start a data slot or to send a `null` frame.

A central question is how to select the poll relayer u. In [11, chap. 7] a scheme is described, which tries to find the "best" relayer, i.e., a station u, which has the best channel quality in both the channels to the BS $C(z, u)$ and to the target station $C(u, a)$. The channel quality is estimated based on channel history information, which the BS collects according to certain heuristics.

D. Priority Restriction (PR)

In the PROFIBUS protocol the transmission of low-priority data is (partially) suppressed in high-load situations by the timed-token method, saving the remaining bandwidth for the more important high priority requests. To implement a similar mechanism with the polling-based protocols, the `poll` frame is equipped with a *priority restriction bit*. If this bit is set, the polled WT is only allowed to handle high priority requests. If this bit is not set, the WT may handle arbitrary requests. The BS determines the value of this bit according to a simple policy: if among the last M polled WT's at most $\alpha \cdot M \cdot k$ (with $0 < \alpha \leq 1$) slots are actually used for data transmission, then this bit is not set for the next station to poll. However, various other policies can be implemented. This mechanism is called *priority restriction* (PR).

E. Link Layer Protocol

In PROFIBUS a WT a maintains a separate instance of the alternating bit protocol (ABP) for every target WT. Furthermore, once the low priority service has started, it cannot be interrupted by newly arriving high priority requests. This can be a problem if the *max_retry* parameter is set to a high value for reliability reasons. In the polling-based protocols we allow for preemptive service. If the high priority request queue is not empty, it is always

served first. This means that the service of a low priority request can be interrupted for servicing a high priority request. If the low priority request and the interrupting high priority request are destined to the same target WT, a single alternating bit does not suffice to distinguish the requests properly. Hence, in the polling-based protocols for each target WT and for each priority a separate ABP instance is maintained, which doubles the number of ABP instances a station has to maintain. In [11, chap. 7] it is discussed how these two different variants can be translated into each other in the integrated scenario, where occasionally frames from wired stations to wireless stations have to be exchanged.

V. Real-Time Performance Measures

In this section we introduce the real-time performance measures. They provide a means for simultaneously capturing the timeliness and reliability of data transmission in a wireless fieldbus system. Timeliness and reliability are the main constituents for providing real-time services. The real-time performance measures are related to the notion of predictability in real-time systems [13]. We assume two traffic priorities: important and safety-critical critical messages are of high priority, everything else is of low priority. The measures are defined with respect to the service primitives (request, confirmation) exchanged at the link layer interface, and also with respect to certain load models, which are presented first.

In real systems the traffic often behaves in a certain manner: in the "normal" mode the underlying industrial process runs without problems and the PROFIBUS traffic is dominated by process data, file transfers, etc., which are all of low priority. When something critical happens, the industrial process enters the "emergency" mode, and many high priority messages are exchanged in addition to the low priority traffic ("alarm storm"). Thus, with respect to the arrival of high priority messages, we can distinguish between "idle" and "storm" periods. While in the middle of a storm period the throughput of high priority frames is of prime importance, in the beginning a low latency is desired to communicate the critical situation as fast as possible.

The latency is captured by the *confirmation delay* measure. This measure evaluates the delay between issuing and receiving confirmation for sporadic high priority (alarm) messages, when some low priority background load is present in the system. The main measure used in this paper is based on the confirmation delay. A second important measure, the *consecutive confirmation delay* [11, chap. 4] evaluates the system throughput in alarm storm situations.

A. Load Model

The load model for the main measure of this paper reflects the above considerations, it is called the **smooth** model. There is some low priority background load[4] of 10% and 50%, corresponding to a lightly loaded and a heavily loaded system, respectively. The background load consists of periodic process data and aperiodic data like file transfers, etc. We assume that the periodic data amounts to 50% of the background load. For the aperiodic data we have used a Poisson arrival process. We assume that all nodes generate the same amount of traffic. One could use a more bursty traffic model for the background load, however, the 50% background load case already provides a worst-case traffic scenario, since with the additional overhead caused by the polling protocol or token passing, data retransmissions and other management traffic the system is close to saturation.

The important requests are of high priority and arrive asynchronously. We are interested in the latency they can achieve in the presence of the background load. For the high priority request we assume Poisson arrivals with an overall load value of 10%, uniformly distributed over all stations. The traffic sources generate high priority requests for a station only when there is no pending high priority request in this station. Hence, we do not take queueing delays into account. Furthermore, the confirmation delays seen by high priority requests at a single station can be assumed to be independent of each other and to have the same distribution. Neither of this would be true for batch arrivals.

B. Measure Definition

Due to the sometimes harsh and variable error conditions on certain types of wireless links it is appropriate to express the requirements stochastically.

For a fixed station i the confirmation delay measure $D_C(i)$ denotes the time between the arrival of a high priority request at i's link-layer interface and the time instant when the corresponding confirmation primitive is generated, i.e., the transmission outcome is known. The assumed traffic scenario is the **smooth** scenario. Hence, there is no bias from queueing delays. Now the main optimization target for this work can be stated conveniently. Denote for a given station i by $F_{D_C(i)}(x) = \Pr[D_C(i) \leq x]$ the distribution function of the confirmation delay values, its 99% quantile is given by $x_{99}(i) = F^{-1}(0.99)$. We want to minimize the quantity $\widetilde{D_C}$ (which we denote as *overall confirmation delay*):

$$\widetilde{D_C} := \max\{x_{99}(i) : i \in \{1, \ldots, N\}\}$$

i.e. the maximum of the 99% $D_C(i)$ quantiles over all stations. The $\widetilde{D_C}$ measure can be defined in the same way for other quantiles, e.g. a 99.9% quantile.

As a side measure, we look at the remaining bandwidth for background traffic B_L in the **smooth** scenario with 50% background load. If two MAC protocols show the same performance with respect to $\widetilde{D_C}$, the scheme which offers

[4]The notion of *load* is defined as follows: for a single priority a load value of x percent has the meaning that in the case of no errors and without packets of other priorities present in the system the load offered via the link-layer interface to all stations is such that the time spent for transmitting the corresponding `data` frames (or their PROFIBUS counterpart) is x percent of the theoretical link bandwidth (including overhead).

Parameter	Values
Number of active stations N	2, 4, 8, 12, 16, 20
T_{TTRT}	0.005, 0.01, 0.015, 0.02, 0.4
gap factor	1, 2, 4, 6, 8
max_retry parameter	20
round-robin bound k	1, 2, 4, 6
Bitrate	2 MBit/s QPSK
PHY overhead (preamble, header)	192 μs
Request size (both priorities)	40 bytes
Inter-Arrival time	depends on load

TABLE I
SIMULATION PARAMETERS

more bandwidth to the background is preferrable. The B_L measure is always given as a fraction of the available theoretical bandwidth, and includes data and overhead of low priority frames.

It is assumed that the maximum number of retransmissions (*max_retry* parameter) is set to a high value ≥ 20, in order to achieve high reliability. Hence, the *negative confirmation rate*, where the MAC-/data link-layer has to report about finally unacknowledged requests, should be very low. If the *max_retry* parameter is chosen to a lower value (e.g., 3 to 5), then the negative confirmation rate would be of much interest.

It is exactly the small relaxation from a deterministic delay measure to 99% quantiles which accounts for the variability and error behavior of the wireless link, while simultaneously expressing hard delay requirements.

VI. Performance Evaluation

In this section we present the results of a simulation study of the real-time performance delivered by the PROFIBUS protocol and the polling-based protocols over different types of wireless links. In the next Section VI-A we discuss the simulation approach, and in Section VI-B the wireless link error models for generating bit errors and packet losses. The real-time performance results are presented in two steps: in Section VI-C we compare the PROFIBUS protocol and the rr-k protocols. In Section VI-D we show that the real-time performance delivered by rr-k can be improved significantly with the SDR, SPR and PR schemes.

A. Simulation Approach

The evaluation of the real-time performance is done via simulation. More specifically, two different simulation models are used: a PROFIBUS model (described in [14]) and a simulator for the polling-based protocols defined in Section IV. Both simulators use the CSIM discrete event simulation library [15], and they have the physical layer (IEEE 802.11 DSSS PHY), the channel error models (Section VI-B) and the request sources for load generation in common. Please refer to Table I for the most important simulation parameters.

The simulation run length was chosen such that the 99% quantile of the confirmation delay distribution for a single station could be obtained with good accuracy. For every station we collected a minimum of 100000 confirmation delay values. Hence, the remaining 1% of the delay distribution is represented by at least 1000 values. We have used a warm-up period of 100 simulated seconds before starting data collection. The validation of the PROFIBUS simulator is discussed in [14], while the polling simulator was validated by careful inspection of the generated traces and comparison against well-known analytical results for certain simple scenarios (no errors, uniform load).

In both models there are N wireless terminals (WT), to each station a number of request sources is attached. The polling model has an additional BS. A fully meshed topology is assumed, i.e., all stations hear each other. All stations have a fixed position, there are no mobile stations. The distance between the stations is small, therefore the propagation delay can be neglected. All stations are active stations, i.e., they require the right to initiate transmissions. The request sources issue SDA requests of both priorities to the respective link layer interface, according to the **smooth** load scenario. The request size is fixed, the load is varied by varying the (mean) inter-arrival times (Table I).

The PROFIBUS frames are embedded into IEEE 802.11 PHY packets, hence, the PHY overhead of 192 μs is added (Section II-A). The MAC frames of the polling-based protocols consist of a MAC header of eight bytes length and an optional MAC data part. The polling MAC frames are also embedded into IEEE 802.11 PHY packets, they have the same overhead. A MAC header of eight bytes offers enough place to accommodate source and destination address, ABP information, a length field, several flags (e.g., priority restriction bit) and a separate MAC header checksum. For simplicity, the checksum is assumed to be perfect. The MAC user data is protected by a separate perfect checksum.[5] There is no segmentation and reassembly.

B. Error Models

Between every pair of stations there is a separate channel with its own error model. The channel error models between different station pairs are assumed to be stochastically independent. This assumption aims at modeling the different propagation environments between spatially separated stations.[6] If the channels are in addition stochastically identical, this is called the *homogeneous* case, other-

[5] A separate MAC header checksum gives the receiver some chance to recover at least some information of an erroneous packet, if the MAC header is not hit by errors. For example, the MAC header of alarm messages can have a separate field carrying an alarm identification tag, while additional alarm information is stored in the MAC data part. This increases the receivers chance to acquire at least partial information quickly.

[6] The validity of this assumption depends on the relative impact of the different mechanisms leading to errors on the wireless channel. If the error behavior is dominated by multipath fading and two stations have a distance of more than a half wavelength, this assumption is reasonably accurate. In fact, current IEEE 802.11 equipment providing receiver diversity uses a spacing of 6-7 cm between the two antennas, which corresponds to a half wavelength at a frequency of 2.4 GHz (12.5cm). On the other hand, if two stations are close to an interferer, they will likely experience correlated error behavior.

mean bit error rate	0.000370
mean packet loss rate	0.1099
mean error burst length	114.807 bits
CoV error burst length	2.341
mean error-free burst length	11514.112 bits
CoV error-free burst length	57.775
mean packet loss burst length	2.68 packets
CoV packet loss burst length	1.848
mean packet-loss free burst length	21.74 packets
CoV packet-loss free burst length	15.9

TABLE II
SUMMARY STATISTICS OF TRACE 24 (COV = COEFFICIENT OF VARIATION)

wise we have the *inhomogeneous* case.

The error model for a single channel takes bit errors and packet losses into account by devising separate sub-models: first the packet loss sub-model is applied to a packet. If no packet loss occurs, the bit error model is applied, marking the packet as error-free or erroneous. The bit error and packet loss sub-models are stochastically independent (see Section II-B). Furthermore, for the models used in this paper we assume that the stochastic processes used for generating packet losses and bit errors are of the same type. For example, in the *Gilbert-Elliot model* both the bit errors and packet losses of a single channel are obtained from two-state Markov models. However, the Markov models have in general different parameters.

Four different stochastic error models of varying statistical accuracy are used: the *independent* model, the *Gilbert-Elliot model*, a *Semi-Markov* model, and a *complex* model. The independent, Gilbert-Elliot and Semi-Markov model are homogeneous channel models, the complex model is an inhomogeneous model. All models are parameterized from Trace 24 of the **factorial** measurement reported in [6]. The complex model uses four different traces. In the following we describe the application of these models to bit errors, the packet loss sub-models are constructed similarly.

The independent model has only a single adjustable parameter and can only match the mean bit error rate (MBER) of Trace 24 (MBER = 0.00037). The Gilbert-Elliot and Semi-Markov model switch conceptually between a "good" state without bit errors and a "bad" state. In the bad state bit errors occur independently of each other, a bit is erroneous with a certain probability p. The overall mean bit error rate is the same as for Trace 24. With respect to the bit error bursts and the bit error free periods, the Gilbert-Elliot model can only match the mean values of the respective burst length distributions, while the Semi-Markov model can match mean and variance. The Semi-Markov model uses lognormal distributions for the state holding times, the Gilbert-Elliot model produces necessarily a geometric distribution. The mean and variance of the burst lengths for bit errors and packet losses for Trace 24 are summarized in Table II.

For the complex model we use different traces: Traces 21 and 24 of the **factorial** measurement (Trace 21: mean BER $\approx 2 \cdot 10^{-4}$, mean PLR $\approx 10\%$), Trace 1 of the **longterm1** measurement (mean BER: 0, mean PLR: $\approx 32\%$), and Trace 17 of the **longterm1** measurement (mean BER: 0, mean PLR: 0.004). For each trace both the packet loss and bit error submodels were expressed as bipartite models [16]. These models use more than two states to increase accuracy. For every channel one of the four traces is selected randomly. The complex model, although not directly comparable to the other models, provides a different challenge to the protocols due to its inhomogeneity. Furthermore, it reflects the fact that the propagation environments to different stations can differ substantially.

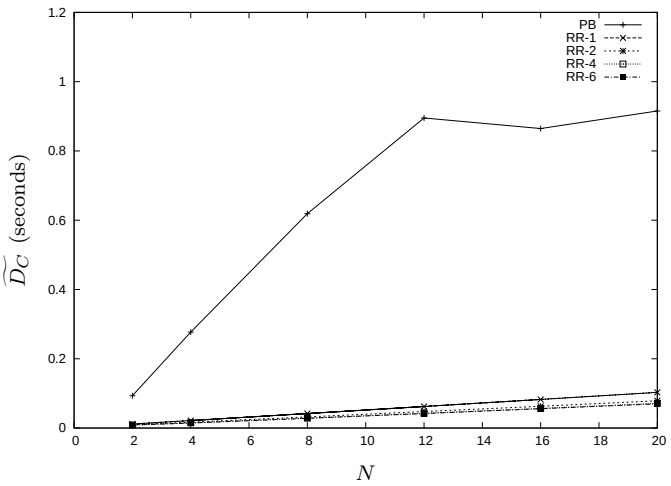


Fig. 2. Overall confirmation delays $\widetilde{D_C}$ for rr-k-protocols and the original PROFIBUS protocol vs. number of WT's N for 10% background load and the Gilbert-Elliot error model

C. Comparison of PROFIBUS and Round-Robin

The set of varied parameters is shown in Table I. The parameters concerning the PROFIBUS model (gap factor, T_{TTRT}) are chosen such that lost stations are re-included fast, at the expense of increased bandwidth need for ring maintenance. Furthermore, the two improvements described in [14] are enabled: a new method for setting the timeout timer avoids that the timer expires first for stations which have no valid view on the logical ring, and the *fast re-inclusion* algorithm tries to re-include lost stations faster than with the original PROFIBUS ring maintenance protocol.

The $\widetilde{D_C}$ performance for the PROFIBUS protocol with a fixed error model and a fixed value number of stations N was obtained by taking the best achievable $\widetilde{D_C}$ value for all gap factors and T_{TTRT} values.

C.1 Gilbert-Elliot and Semi-Markov Model

For the Gilbert-Elliot and Semi-Markov models and for both 10% and 50% background load the rr-k protocols show a significant advantage in terms of $\widetilde{D_C}$ performance over the PROFIBUS protocol. For higher number of stations and 10% background load it reaches up to an order of magnitude, see for example Figure 2, where the $\widetilde{D_C}$ value for

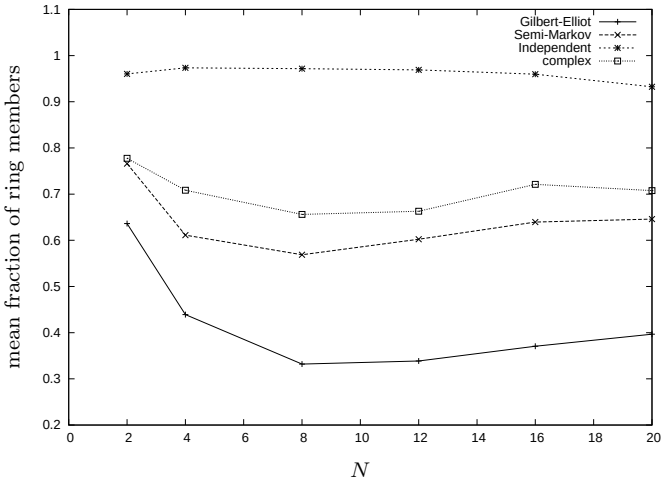


Fig. 3. Fraction of the mean number of PROFIBUS ring members to the overall number of WTs vs. number of WT's for the PROFIBUS protocol and 10% background load

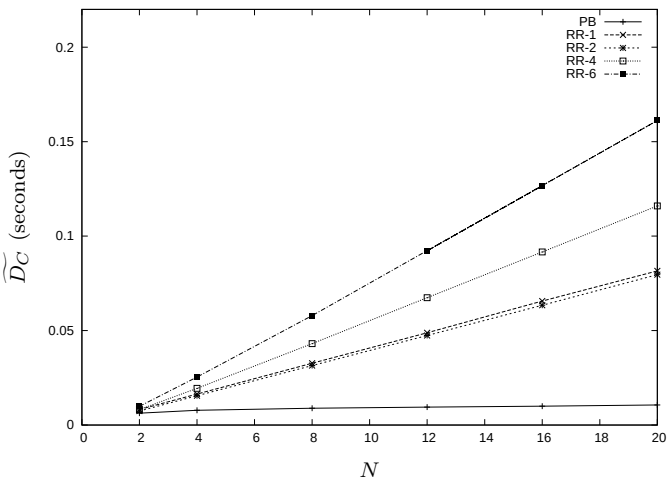


Fig. 4. Overall confirmation delays $\widetilde{D_C}$ for the rr-k-protocols and the original PROFIBUS protocol vs. number of WT's N for 50% background load and the independent error model

the PROFIBUS reaches up to 900 ms, while the rr-k protocols almost never exceed 100 ms. For the 50% background load case the PROFIBUS has between two and four times higher $\widetilde{D_C}$ values than the rr-k protocols. A likely explanation for the poor PROFIBUS performance are problems with the stability of the logical ring [14]: due to repeated losses of token frames stations often are unwantedly excluded from the ring (see Section III-A). To validate this claim, we show in Figure 3 the mean fraction of ring members for the 10% background load case. The mean fraction of ring members is defined as the ratio of the mean number of ring members to the target number of ring members N (again, the best achievable values are shown). The mean fraction of ring members is significantly below one for the bursty error models.

The $\widetilde{D_C}$ performance of the rr-k protocols depends on the protocol parameter k. For the case of 10% background load rr-4 and rr-6 tend to be better than rr-1 and rr-2 (although not much). For the case of 50% background load the opposite is true: rr-1 and rr-2 show much better $\widetilde{D_C}$ performance than rr-4 and rr-6. For example: for the Gilbert-Elliot model, 50% background load and $N = 20$ the $\widetilde{D_C}$ performance for rr-6 is ≈ 0.22 sec, the $\widetilde{D_C}$ performance for rr-1 is ≈ 0.12 sec. For the case of a high background load the advantage of small k values is due to *blocking*: WT ν may handle up to k frames belonging to low priority requests, while the high priority requests at its successor WT $\nu+1$ have to wait. For the case of low background load the advantage of larger k values over the smaller ones can be explained as follows: we observe from Figure 2 that the curve for $k = 2$ is closer to the curves for $k = 4$ and $k = 6$ than to the curve for $k = 1$. A WT can make benefit of $k > 1$ due to having the opportunity to perform immediate retransmissions of high priority frames, instead of having to wait for the next poll cycle. For the high load case this effect is overshadowed by blocking.

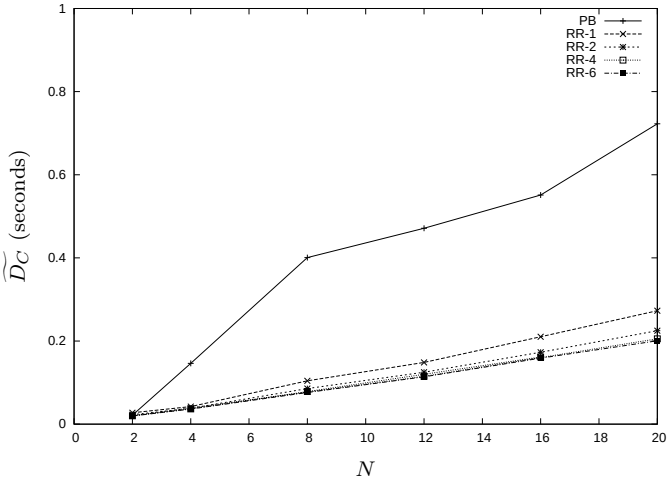


Fig. 5. Overall confirmation delays $\widetilde{D_C}$ for rr-k-protocols and the original PROFIBUS protocol vs. number of WT's N for 10% background load and the complex error model

C.2 Independent and Complex Errors

For the independent error model we achieve surprising results. The $\widetilde{D_C}$ performance for the 50% background load case is shown in Figure 4. The absolute $\widetilde{D_C}$ values for both the rr-k protocols and the PROFIBUS protocol for the independent error model are much lower than for the other error models. In fact, the PROFIBUS shows the best ring stability for the independent error model, as can be seen from Figure 3). Hence, fewer high priority requests are blocked in the queues of non-ringmember stations waiting for re-inclusion. Furthermore, the PROFIBUS protocol (and the rr-k protocols with $k \geq 2$) can perform retransmissions immediately, which is most appropriate for independent errors. In this case, postponing a retransmission gives not a better probability of success, as can be expected for bursty channels after waiting long enough. Here, postponing only increases the delay.

The truly interesting observation is that the PROFIBUS

protocol achieves better results than the rr-k protocols. The PROFIBUS achieves the best results for T_{TTRT} = 5ms. This points towards an explanation: for small T_{TTRT} and large N a station often computes a negative token holding time upon token arrival, which suppresses transmission of low priority frames. In contrast, in the rr-k protocols a WT is not restricted in handling low priority requests, giving rise to blocking effects. Indeed, careful inspection of the simulation results shows that low priority traffic starves out under the PROFIBUS protocol for increasing number of WT's. For the rr-k protocols the low priority traffic always gets a fixed share of bandwidth, depending on k. This behavior can be changed with the priority restriction (PR) mechanism.

The $\widetilde{D_C}$ performance for the complex error model and 10% background load is shown in Figure 5. Again, the rr-k protocols have a clear advantage over the PROFIBUS protocol. In contrast, for 50% background load the PROFIBUS protocol outperforms rr-6 and rr-4 for $N \geq 16$. However, even for this case rr-1 has only $\approx 50\%$ of the $\widetilde{D_C}$ value of PROFIBUS, and rr-2 has only 70% of its value.

The rr-k protocols show for all error models a linear dependence of $\widetilde{D_C}$ performance from N, only the slope varies with the background load and error model. This observation reflects a simple property of the round-robin protocols: each station gets the same share of bandwidth and can make purely local decisions on its usage, without considering the behavior of other stations.

D. Comparison of Round-Robin with the Add-Ons

In this section we compare rr-k with each of the protocols rr-k+SDR (simple data relaying), rr-k+SPR (simple poll relaying) and rr-k+PR (priority restriction). More specifically, for a fixed set of parameters we compute the ratio $\frac{\widetilde{D_C}(rr-k+x)}{\widetilde{D_C}(rr-k)}$. For example: if the ratio is 0.7, we say that rr-k+x has 30% better $\widetilde{D_C}$ performance than rr-k.

The common simulation parameters for all the protocols are the same as we have used for comparing rr-k with the PROFIBUS protocol, see Table I. The T_{Quiet} parameter for the SPR scheme is set to three times the maximum poll cycle duration (see Section IV-C).

Before discussing $\widetilde{D_C}$ performance results, we state the result that for both rr-k+SPR and rr-k+SDR the additional costs in B_L performance are always below 4% as compared to rr-k, see [11, chap. 7]. Hence, these mechanisms impose no significant penalty in terms of available bandwidth for low priority data.

D.1 rr-k vs. rr-k+SPR

The $\widetilde{D_C}$ performance of the rr-k+SPR protocol as compared to the rr-k protocol depends on the background load. For the case of 50% background load rr-k+SPR achieves significant gains, especially for $k > 1$. The larger the value of k, the larger the gain. For the Gilbert-Elliot model and the Semi-Markov model savings of up to 60% are achieved with rr-6+SPR, up to 50% for rr-4+SPR and more than 30% for rr-2+SPR. We show the results for the complex error model in Figure 6, the advantage of rr-k+SPR over rr-k is again significant. The savings for the independent model are less impressive, but for $k > 1$ and larger N up to 20% are achieved.

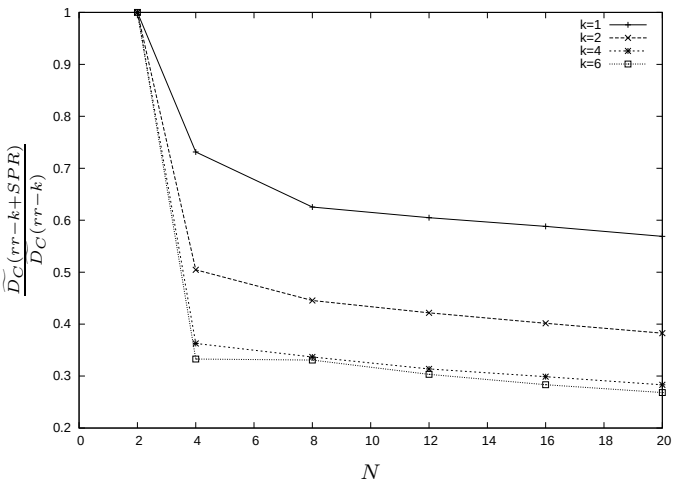


Fig. 6. Ratio of the Overall confirmation delay for the rr-k+SPR protocol and the rr-k protocol $\frac{\widetilde{D_C}(rrk+spr)}{\widetilde{D_C}(rrk)}$ vs. number of WT's N for 50% background load and complex error model

In the case of 10% background load the rr-k+SPR protocol achieves up to 30% better performance for $k \geq 2$ and both the Gilbert-Elliot and Semi-Markov models. The rr-2+SPR protocol gives the best results, rr-6+SPR gives the smallest gains. The rr-1+SPR protocol performs often worse than rr-1. For the independent error model all rr-k+SPR protocols are worse than rr-k, with rr-1+SPR being the poorest. The rr-4+SPR and rr-6+SPR protocols loose only 2-3%, rr-1+SPR up to 20% against rr-k, increasing with N. For the complex model and $N > 2$ the rr-k+SPR protocols achieve gains from $\approx 20\%$ ($k = 1$) up to 45% ($k = 6$).

The different behavior for 10% and 50% background load can be explained by the operation of the rr-k protocols (Section IV-A): if the BS polls a WT a having empty queues, the WT keeps quiet. The BS cannot distinguish between a having empty queues and poor channel conditions between the BS and a. If the WT's load is low it experiences a higher probability for no newly arriving requests within a certain time. Hence, the probability of the SPR mechanism querying an empty WT is higher. In these cases the SPR mechanism is pure overhead without any gain. On the other hand, when the WT is highly loaded, the probability of having empty queues upon reception of a `poll` frame is low, and the SPR mechanism is likely invoked due to a poor channel. We have verified this explanation by counting the number of SPR cycles for the two different loads. As an example, for $N = 20$, $k = 1$ in the case of 10% background load we have observed ≈ 15 times more SPR cycles than for the case of 50% background load. This overhead tends to increase delays.

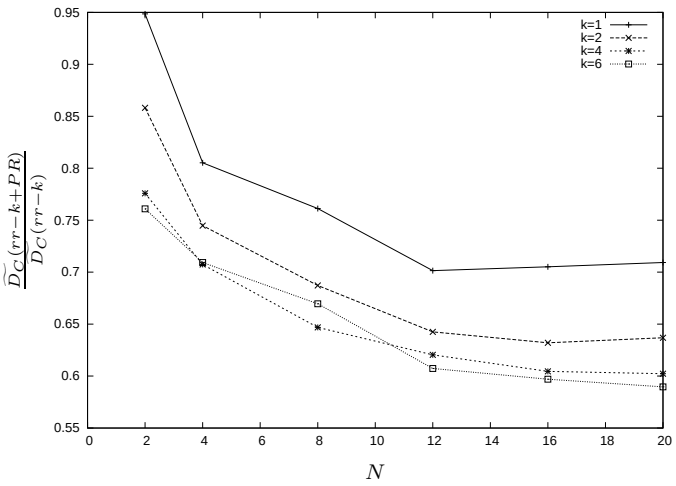


Fig. 7. Ratio of the Overall confirmation delay for rr-k+PR protocol and the rr-k protocol $\frac{\widetilde{D_C}(rrk+PR)}{\widetilde{D_C}(rrk)}$ vs. number of WT's N for 50% background load and Gilbert-Elliot error model

D.2 rr-k vs. rr-k+SDR

For all loads and all error models we can observe the same trends: the biggest gains of rr-k+SDR over rr-k can be observed for $k = 1$, they are typically in the range of 15% to 25%. For $k = 2$ we observe almost always gains, typically between 5% and 15%. Only for the independent model and 50% low priority load the rr-2+SDR and rr-2 protocols show the same performance. The finding that rr-k+SDR achieves the largest gains for $k = 1$ can be explained by the fact that the SDR mechanism performs a kind of immediate retransmission, which otherwise a WT could not achieve for $k = 1$. Without the SDR mechanism the WT would have to wait an entire token cycle before it can retransmit a packet. The probability that a retransmission has to wait for a token cycle is higher for $k = 2$ than it is for $k = 4$ and $k = 6$. Consequently, rr-2+SDR achieves the second-best results. For $k = 4$ and $k = 6$ we observe comparably small gains (up to 10%) or even slight losses (up to 5%). This holds for both background loads and all error models. For higher values of k a WT can typically handle high priority requests fully within one token cycle. Hence, the SDR mechanism can save an entire poll cycle less often.

D.3 rr-k vs. rr-k+PR

We present results only for the 50% background load case. For the sake of example we have chosen $M = N$ and $\alpha = 0.4$. Hence, a WT may transmit low priority data only when in the last poll cycle not more than 40% of the assigned slots were actually used. In Figure 7 we compare the $\widetilde{D_C}$ performance for the case of complex errors. The figure is representative for all error models: for $k = 4$ and $k = 6$ we observe gains of up to 45%, and both curves are close to each other, the difference is smaller than 5%. For $k = 1$ the gains are typically between 25% and 35% for $N \geq 12$, and for $k = 2$ gains between 30% and 40% can be achieved for $N \geq 8$. The advantage of larger k values over the smaller ones is not surprising: the PR mechanism is explicitly targeted at reducing blocking effects, which occur more likely for larger values of k.

E. Summary of Results

The PROFIBUS has serious problems with ring stability in the case of bursty errors (complex, Gilbert-Elliot, Semi-Markov), and thus a poor real-time performance. In contrast, for the case of independent errors the ring stability is better, and the PROFIBUS shows good real-time performance.

Both relaying approaches have shown their effectiveness in reducing the $\widetilde{D_C}$ values, while being cheap in B_L performance. The results for the polling-based protocols suggest to choose $k = 2$ as basic protocol parameter: $k = 4$ and $k = 6$ suffer from blocking problems, while for $k = 1$ a single request occasionally needs several token cycles for completion, therefore increasing delays. The value $k = 2$ allows for one immediate retransmission. This seems to be a good heuristic for bursty channels: if one trial and one retransmission do not suffice, then it is good to proceed with other stations and to wait a while (here: a poll cycle).

VII. Related Work

There exists much literature on polling systems in general, mostly concerned with their queueing analysis. Some references are [17], [18], [12]. For packet polling systems with transmission errors some results (including mean response time) for infinite buffer polling systems are presented in [17], [19]. In [20] a polling strategy was used to maximize throughput on the downlink in high load situations and for the case of mobile stations with pedestrian speeds. Polling was used to periodically estimate the channel quality between the BS and the wireless terminal, and to adjust antenna weights appropriately. The channel is assumed to follow a Gilbert-Elliot error model. The focus was entirely on throughput, message delays were not considered. In [21] a table-based polling protocol is investigated under the Gilbert-Elliot error model. The polling-table accommodates synchronous and asynchronous data as well as retransmissions (one trial per packet and token cycle). Besides packet retransmissions they have also investigated the usage of error correcting codes, but these are found to be too wasteful, except for situations with long channel outage periods. The achievable throughput is shown to depend on the channel outage probability and the mean channel outage period. Furthermore it is shown how the mean delay depends on the utilization, there are no results on delay quantiles or on the influence of the channel outage probabilities on delay.

Several authors have addressed wireless fieldbus systems. Some selected references for other fieldbus systems include [22], where a group at EPFL Lausanne has reported work on an application layer gateway for integrating wireless stations into the FIP fieldbus (Factory Instrumentation Protocol), employing a TDMA scheme on the wireless side. An approach to enhance the MAP/MMS protocol with mobility is presented in [23]. In the proposed system the IEEE 802.11 MAC protocol with its stochastic MAC proto-

col is used, time critical transmissions are not considered. The application of DECT (Digital European Cordless Telephone) for providing mobility support to MAP/MMS is discussed in [24]. For the CAN fieldbus different mappings of the CAN MAC protocol to wireless media are discussed in [25], [26], however, channel errors and retransmissions were not taken into account. The Funbus project [27], [28] investigated different wireless technologies for their potential to provide transparent coupling of wireless and wired field devices. Within the project three different fieldbus technologies (PROFIBUS-DP, INTERBUS-S and CAN) are investigated, IEEE 802.11 DSSS was chosen as wireless technology. A wireless repeater approach was evaluated experimentally. The successor project R-FIELDBUS (*www.rfieldbus.de*) evaluates different wireless technologies for the PROFIBUS, P-NET and WorldFIP fieldbus systems with focus on multimedia support. In [29] a application layer gateway approach for integration of wireless nodes in a PROFIBUS-DP network (single master, many slaves, no token passing) is described. A similar approach is applied to PROFIBUS-FMS in [1], with application to an automated container terminal. Basically, an application layer gateway couples a PROFIBUS master station with a wireless "virtual master" station. The virtual master acts as a proxy for the wireless stations, it polls them using standard IP and IEEE 802.11 protocols. The data between the wireless and the wired side are exchanged asynchronously via intermediate buffers.

The author of the present paper has recently proposed another MAC protocol for a wireless PROFIBUS, namely the adaptive intervals (AI) MAC protocol [30], which is a combination of polling and group testing with time-variable groups. Group testing can be effective in reducing the overhead of polling protocols. With respect to $\widetilde{D_C}$ performance the AI protocols outperform the PROFIBUS protocol with improvements significantly. In fact, the performance achievable with AI is not significantly worse than what can be achieved with the protocols presented in this paper.

VIII. Summary and Outlook

The results presented in this paper indicate that polling-based protocols are a good candidate for a wireless PROFIBUS, since even the basic k-limited round-robin protocol gives substantially better real-time performance than the PROFIBUS protocol in the presence of bursty errors. We believe that the gains in real-time performance outweigh the additional complexities of running two different protocols in a single LAN. However, the design space for this integration needs further exploration.

With respect to the polling-based MAC protocols, there are many opportunities for further research. One issue not addressed here is the behavior of the proposed SPR, SDR, and PR mechanisms when they are composed. The choice of parameters for the rr-k+x protocols used in this paper is based on an "educated guess", there is likely a potential to achieve even better real-time performance of the protocols by careful parameter tuning and by introducing better means for adapting to the channel and load situations. The assumption that the channels between each pair of stations are independent of each other needs not to be true in environments where the error behavior is dominated by interference. This case deserves careful investigation. It needs also to be investigated how the protocols behave in case of overlapping cells, where single stations can be in the range of two or more base stations. Another research opportunity is to attack the overhead of polling protocols, e.g. by using group testing. A combination of the mechanisms presented in this paper with the AI protocol seems very promising.

The results obtained for the PROFIBUS and independent errors show that this protocol can deliver good real-time performance, if only the ring is stable. Hence, good schemes for increasing the stability of the logical ring in the presence of bursty errors are needed. This, however, appears to be a challenging problem, specifically when one wants to integrate wired and wireless stations without changing the wired stations protocol stack. And likely these schemes do not resolve another drawback of the PROFIBUS token passing protocol: it requires that all active stations can hear each other, which is not always easy to achieve.

IX. Acknowledgements

The author wishes to thank the anonymous reviewers for their valuable comments.

References

[1] Suk Lee, Kyung Chang Lee, Man Hyung Lee, and Fumio Harashima. Integration of mobile vehicles for automated material handling using Profibus and IEEE 802.11 networks. *IEEE Transactions on Industrial Electronics*, 49(3):693–701, June 2002.

[2] The Editors of IEEE 802.11. *IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*, November 1997.

[3] The Editors of IEEE 802.11. *IEEE Standard for Information Technology - Telecommunications and information exchange between systems - Local and Metropolitan networks - Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Higher speed Physical Layer (PHY) extension in the 2.4 Ghz band*, 1999.

[4] K. Pahlavan and A.H. Levesque. *Wireless Information Networks*. J. Wiley and Sons, 1995.

[5] W. C. Jakes. *Microwave Mobile Communications*. Wiley, New York, 1974.

[6] Andreas Willig, Martin Kubisch, Christian Hoene, and Adam Wolisz. Measurements of a wireless link in an industrial environment using an IEEE 802.11-compliant physical layer. *IEEE Transactions on Industrial Electronics*, 2001. accepted for publication.

[7] German Institute of Standardization (DIN). *PROFIBUS Standard Part 1 and 2*, 1991.

[8] PROFIBUS Nutzerorganisation e.V., PROFIBUS Nutzerorganisation e.V., Haid-und-Neu-Str. 7, Karlsruhe, Germany. *Implementation Guide to DIN 19245 Part 1*, August 1994.

[9] Union Technique de l'Electricité. *General Purpose Field Communication System, EN 50170, Volume 2: PROFIBUS*, 1996.

[10] Andrew T. Campbell. QOS-aware middleware for mobile multimedia communications. *Journal on Multimedia Tools and Applications*, 7(1-2):67–82, July 1998.

[11] Andreas Willig. *Investigations on MAC and Link Layer for a wireless PROFIBUS over IEEE 802.11*. PhD dissertation, Technical University Berlin, Department of Electrical Engineering and Computer Science, May 2002.

[12] Boudewijn R. Haverkort. *Performance of Computer Communication Systems – A Model Based Approach*. John Wiley and Sons, Chichester / New York, 1998.

[13] Kang G. Shin and Parameswaran Ramanathan. Real-time computing: A new discipline of computer science and engineering. *Proceedings of the IEEE*, 82(1):6–24, January 1994.

[14] Andreas Willig and Adam Wolisz. Ring stability of the PROFIBUS token passing protocol over error prone links. *IEEE Transactions on Industrial Electronics*, 48(5):1025–1033, October 2001.

[15] Mesquite Software, Inc., T. Braker Lane, Austin, Texas. *CSIM18 Simulation Engine – Users Guide*, 1997.

[16] Andreas Willig. A new class of packet- and bit-level models for wireless channels. In *Proc. 2002 IEEE PIMRC*, 2002. accepted for publication.

[17] Hideaki Takagi. Queueing analysis of polling models: an update. In Hideaki Takagi, editor, *Stochastic Analysis of Computer and Communication Systems*, pages 267–318. Elsevier, Amsterdam, 1990.

[18] Hideaki Takagi. *Analysis of Polling Systems*. MIT Press, Cambridge, Massachusetts, 1986.

[19] Hideaki Takagi. Analysis and applications of a multi-queue cyclic service system with feedback. *IEEE Transactions on Communications*, 35(2):248–250, February 1987.

[20] Zhensheng Zhang and Anthony S. Acampora. Performance of a modified polling strategy for broadband wireless lans in a harsh fading environment. *Telecommunication Systems*, 1:279–294, 1993.

[21] A. Hoffmann, R. J. Haines, and A. H. Aghvami. Performance analysis of a token based MAC protocol with asymmetric polling strategy ('TOPO') for indoor radio local area networks under channel outage conditions. In *Proc. International Conference on Communications (ICC)*, pages 1306–1311, New Orleans, Louisiana, 1994. IEEE.

[22] Philip Morel and Alain Croisier. A wireless gateway for fieldbus. In *Proc. Sixth International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 95)*, 1995.

[23] Philip Morel and Jean-Dominique Decotignie. Integration of wireless mobile nodes in MAP/MMS. In *Proc. 13th IFAC Workshop on Distributed Computer Control Systems DCCS95*, 1995.

[24] Philip Morel. Mobility in MAP networks using the DECT wireless protocols. In *Proc. 1995 IEEE Workshop on Factory Communication Systems, WFCS'95*, Leysin, Switzerland, 1995.

[25] A. Kutlu, H. Ekiz, and E. T. Powner. Performance analysis of MAC protocols for wireless control area network. In *Proc. Intl. Symp. on Parallel Architectures, Algorithms and Networks*, pages 494–499, Beijing, China, June 1996.

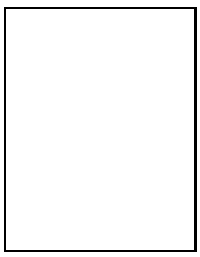
[26] A. Kutlu, H. Ekiz, M. D. Baba, and E. T. Powner. Implementation of "Comb" based wireless access method for control area network. In *Proc. 11th Intl. Symp. on Computer and Information Science*, pages 565–573, Antalaya, Turkey, November 1996.

[27] Projektkonsortium Funbus. Das Verbundprojekt Drahtlose Feldbusse im Produktionsumfeld (Funbus) – Abschlußbericht. INTERBUS Club Deutschland e.V., Postf. 1108, 32817 Blomberg, Bestell-Nr: TNR 5121324, October 2000. http://www.softing.de/d/NEWS/Funbusbericht.pdf.

[28] L. Rauchhaupt Jörg Hähniche. Opportunities and problems of wireless fieldbus extensions. In *Proc. FeT'99: Feldbustechnik – Fieldbus Technology*, Magdeburg, 1999. Springer Verlag, Wien / New York.

[29] Kyung Chang Lee and Suk Lee. Integrated Network of PROFIBUS-DP and IEEE 802.11 wireless LAN with Hard Realtime Requirement. In *Proc. IEEE 2001 International Symposium on Industrial Electronics*, Korea, 2001. IEEE.

[30] Andreas Willig. The adaptive-intervals MAC protocol for a wireless PROFIBUS. In *Proc. 2002 IEEE International Symposium on Industrial Electronics*, 2002. accepted.



Andreas Willig (M '97) is currently a research assistant at the Telecommunication Networks Group at Technical University Berlin. He earned the diploma degree in computer science from University of Bremen in 1994 and the Dr.-Ing. degree from Technical University Berlin in 2002. His research interests include wireless and cellular networks, ad-hoc and sensor networks, (wireless) fieldbus systems, realtime systems and active networks.