Halmstad University Post-Print

Reliable hard real-time communication in industrial and embedded systems

Magnus Jonsson and Kristina Kunert

N.B.: When citing this work, cite the original article.

©2008 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

Jonsson M, Kunert K. Reliable hard real-time communication in industrial and embedded systems. In: SIES'2008 Third International Symposium on Industrial Embedded Systems : June 11-13, 2008 : Montpellier-La Grande Motte, France. IEEE; 2008. p. 184-191.

DOI: <http://dx.doi.org/10.1109/SIES.2008.4577698>

Copyright: IEEE

Post-Print available at: Halmstad University DiVA
<http://urn.kb.se/resolve?urn=urn:nbn:se:hh:diva-2145>

Reliable Hard Real-Time Communication in Industrial and Embedded Systems

Magnus Jonsson and Kristina Kunert

CERES (Centre for Research on Embedded Systems), School of Information Science, Computer and Electrical Engineering
Halmstad University
Halmstad, Sweden
{Magnus.Jonsson, Kristina.Kunert}@hh.se

Abstract— This paper presents a framework for how to use ARQ (Automatic Repeat Request) in combination with real-time worst-case scheduling analysis to be able to support reliable hard real-time communication. We show how to handle retransmissions of erroneous data packets, while still not jeopardizing stated delay guarantees of other packets. We demonstrate this by taking a point-to-point link as an example. Through our simulation studies we have shown that a reduction of the message error rate by several orders of magnitude is possible with a reasonable utilization penalty.

Keywords—Real-time communication; reliability; ARQ; delay bound guarantee

I. INTRODUCTION

In many future industrial and embedded systems, different parts of the system (e.g., sub-systems, processors or sensors) will communicate with each other. Moreover, many such systems have strict timing requirements in terms of, e.g., maximum delay from the moment antenna signals arrive at a radio base station until data or voice information is forwarded out on a network. To guarantee that such timing requirements are met, real-time communication systems are used inside these products. Reliability in combination with real-time performance has this far, however, almost only been addressed for safety-critical systems, in which considerable amounts of redundancy are added. In the work reported in this paper we aim for increased reliability with only small amounts of extra resource usage. In this way, future embedded and industrial systems with tough timing constraints can get improved reliability at minimal cost.

The paper focuses on how to use ARQ (Automatic Repeat Request) in combination with real-time worst-case scheduling analysis. We propose a framework to give hard real-time communication support, where no deadline misses are acceptable neither for the first transmitted instance of a packet, nor for any retransmitted instances. Our framework supports retransmission of erroneous packets as long as it is meaningful according to their deadlines (i.e., as long as retransmitted packets will arrive in time), while the extra network capacity consumed will not jeopardize the stated real-time guarantees of any existing traffic flow in the network.

Even though there has been a lot of work done on both ARQ for QoS (Quality of Service) communication and guaranteed real-time communication, the approach of taking a holistic view on those two areas together, with the aim of being able to calculate guaranteed performance bounds, has been studied less thoroughly. Since we aim for hard real-time communication in embedded and distributed systems, our approach is to use traffic models and develop analysis methods originating from the real-time systems field. In our framework we are not only targeting a long-term statistical delay bound but want to give a guarantee for every single packet by using worst-case delay analysis. This is necessary in order to be able to support delay bounds for periodic traffic in real-time systems. Therefore we do not allow retransmissions of a packet if it jeopardizes the stated real-time guarantees of other packets, which would be possible in standard ARQ schemes.

Former research on ARQ for real-time communication includes work where only average performance, like average delay, is considered [1]. Deadline dependent coding is a promising technique to incorporate error correcting codes and ARQ in real-time communication [2] [3]. However, no attempt to establish a real-time scheduling analysis framework around it has been made.

Butt tries to decrease the bit error rate of hard real-time traffic by retransmitting erroneous packets as long as they still will meet their hard deadline [4]. The solution is build upon an idle RQ approach and is only described on the packet level. Also, no queuing or scheduling analysis is provided. In contrast to idle RQ, which usually leads to a higher penalty in link utilization, our retransmission scheme is more similar to a continuous RQ protocol. Additionally, our delay bound analysis on real-time channel level, including queuing delay analysis, enables the provision of end-to-end delay guarantees.

Giancola et al. made some efforts to provide real-time guarantees including retransmissions [5]. However, their methods use flow analysis, which has been shown to not fully exploit the available network capacity compared to real-time scheduling analysis [6]. Additionally, their analysis does not include the timing details needed to support hard real-time communication in embedded and industrial real-time systems, while we are targeting an as exact analysis as possible. Moreover, the solution proposed in [5] requires traffic regulators to be implemented in each node on the path to

Figure 1. Layering

control the traffic. In our proposal, we draw up a more general framework to combine ARQ with real-time scheduling analysis. We will here consider a point-to-point link with the traffic scheduled by the Earliest Deadline First (EDF) policy, but we plan to extend the framework to address multi-hop networks and other scheduling policies as well.

The rest of the paper is organized as follows. An overview of the framework is given in Section 2. The protocol together with a timing analysis is then presented in Section 3. In Section 4, the real-time scheduling analysis is described, and in Section 5, we discuss extensions to the basic protocol. In Section 6, we present results from our simulation experiments, while the paper is concluded in Section 7.

II. FRAMEWORK OVERVIEW

We assume a network (represented by a point-to-point link in this paper, but it can be a multi-hop network instead) with an existing real-time analysis method for error-free communication. As ARQ schemes working end-to-end over a network normally reside in the transport layer, we assume so in this paper as well. All parameters used in the paper are summarized in Table I. Traffic over the network is specified through Q logical real-time channels (RT channels), τ_i , $1 \leq i \leq Q$, each specified by source, destination, period, message length (neglecting headers in this section) and end-to-end delay bound: $\tau_{N,i} = \{m_{s,i}, m_{d,i}, P_{N,i}, L_{N,i}, D_{N,i}\}$, where N denotes the network level. In a corresponding way we have such RT channels requested by the application layer (or the designer of the system) from the transport layer: $\tau_{T,i} = \{m_{s,i}, m_{d,i}, P_{T,i}, L_{T,i}, D_{T,i}\}$, where T denotes the transport layer. The layering is shown in Fig. 1. By mapping the transport-layer RT channels directly onto network-layer RT channels, we can use the existing analysis to check whether we can guarantee the requested performance for all the RT channels on the network level, assuming error-free communication. When mapping directly in this way, however, the transport protocol does not take care of retransmissions. Below, we will describe our retransmission request (ARQ) scheme, and how to combine a timing analysis of it with a real-time scheduling analysis.

As exemplified in Fig. 2, we divide the transport-layer delay bound, $D_{T,i}$, into one delay bound for ordinary transmission of the packets belonging to the message (three packets in the example), $T_{D_ord,i}$, and one delay bound for the possible retransmission of one of the packets, $T_{D_retr,i}$ (remark our notion with D and P in RT channel specifications and T for other time notations). We will thereby have:

$$D_{T,i} = T_{D_ord,i} + T_{D_retr,i} . \quad (1)$$

Figure 2. Example of ARQ time-sequence diagram

We then set all parameters of $\tau_{N,i}$ to the corresponding values of $\tau_{T,i}$ except for:

$$D_{N,i} = T_{D_ord,i} . \quad (2)$$

As seen in the figure, we do not wait for an acknowledgement (ACK) packet for a retransmitted packet. The reason is that we, in the scope of this paper, do not support more than one packet retransmission per message at the most. Network resources are allocated separately for possible retransmissions through a dedicated RT channel (further elaborated on below).

III. PROTOCOL DEFINITION AND TIMING ANALYSIS

In this section, we define the protocol and show the timing analysis necessary to be able to continue with a worst-case real-time scheduling analysis, analyzing the queuing delay, in the next section. We define a rather straightforward ARQ protocol, for which we assume that both flow control and congestion control can be omitted. Congestion control can be omitted because we are guaranteed to avoid congestion as long as we comply with the real-time scheduling analysis (see below). Further on, we argue that flow control is unnecessary considering that we already have agreed on receiving traffic at a certain packet rate [7] [8]. We also assume a dedicated channel for acknowledgement packets. In our case of a point-to-point link this translates to a dedicated physical point-to-point link in the opposite direction for ACK packets. Sequence numbers are assumed to be included in the header. When applicable for both $\tau_{T,i}$ and $\tau_{N,i}$ we refer to a RT channel as τ_i .

Each message is divided into packets. The maximum length (in bits) of a packet including header is:

$$L_{pack} = L_{data} + L_{header} \quad (3)$$

where L_{data} is the maximum amount of pure data per packet and L_{header} is the header length. The number of packets per message is:

$$N_{pack,i} = \left\lceil \frac{L_{T,i}}{L_{data}} \right\rceil \quad (4)$$

TABLE I. FRAMEWORK AND ANALYSIS PARAMETERS

Parameter	Description
Q	Number of RT channels
τ_i	Logical real-time channel i
$\tau_{N,i}$	Network level logical real-time channel i
$m_{s,i}$	Source of logical real-time channel τ_i
$m_{d,i}$	Destination of logical real-time channel τ_i
$P_{N,i}$	Period of network level logical real-time channel $\tau_{N,i}$
$L_{N,i}$	Message length of network level logical real-time channel $\tau_{N,i}$ (exkluding header)
$D_{N,i}$	End-to-end delay bound of network level logical real-time channel $\tau_{N,i}$
$\tau_{T,i}$	Transport level logical real-time channel i
$P_{T,i}$	Period of transport level logical real-time channel $\tau_{T,i}$
$L_{T,i}$	Message length of transport level logical real-time channel $\tau_{T,i}$ (exkluding header)
$D_{T,i}$	End-to-end delay bound of transport level logical real-time channel $\tau_{T,i}$
$T_{D \text{ ord},i}$	End-to-end delay bound for ordinary transmission of logical real-time channel τ_i
$T_{D \text{ retr},i}$	End-to-end delay bound for the possible retransmission of one of the packets of logical real-time channel τ_i
L_{pack}	Maximum length (in bits) of a packet including header
L_{data}	Maximum amount of pure data (in bits) per packet
L_{header}	Header length (in bits)
$N_{\text{pack},i}$	Number of packets per message belonging to logical RT channel τ_i
$L_{T,i}$	Amount of pure data per message belonging to logical RT channel τ_i
$N_{\text{pack_max},i}$	Of $N_{\text{pack},i}$ packets belonging to logical RT channel τ_i , the first $N_{\text{pack_max},i}$ packets are maximum sized packets
$L_{\text{last},i}$	Length of the last packet of a message belonging to logical RT channel τ_i
T_x	Transmission time of a full-length packet
R	Bit rate of the physical link
$T_{x \text{ last},i}$	Transmission time of the last packet belonging to logical RT channel τ_i
$T_{\text{ACK},i}$	Transmission time of an ACK packet acknowledging a packet belonging to logical RT channel τ_i
T_{prop}	Propagation delay over the physical link
$T_{x \text{ tot},i}$	Total transmission time of a message belonging to logical RT channel τ_i
$T_{\text{proc } 1}$	Processing time between the reception of a packet until its corresponding ACK packet is sent
$T_{\text{proc } 2}$	Processing time between the timeout without reception of an ACK packet until the retransmission can be initiated
T_{timeout}	Timeout instance
T_{margin}	Safety margin between the expected ACK reception and the timeout instance
$T_{d \text{ ord},i}$	Maximum queuing delay of a packet belonging to logical RT channel τ_i
$\tau_{\text{retr},i}$	Dedicated logical retransmission RT channel i
$m_{\text{retr } S,i}$	Source of logical retransmission real-time channel $\tau_{\text{retr},i}$
$m_{\text{retr } D,i}$	Destination of logical retransmission real-time channel $\tau_{\text{retr},i}$
$P_{\text{retr},i}$	Period of logical retransmission real-time channel $\tau_{\text{retr},i}$
$L_{\text{retr},i}$	Message length of logical retransmission real-time channel $\tau_{\text{retr},i}$
$D_{\text{retr},i}$	End-to-end delay bound of logical retransmission real-time channel $\tau_{\text{retr},i}$
$L_{\text{max},i}$	Maximum length of a packet belonging to logical RT channel τ_i
$T_{x \text{ max},i}$	Maximum transmission time of a maximum length packet belonging to logical RT channel τ_i
$L_{\text{retr},1}$	Maximum packet length among all packets belonging to any RT channel
$T_{x \text{ retr},1}$	Maximum transmission time among all packets belonging to any RT channel
$T_{d \text{ retr},1}$	Maximum queuing delay, i.e., the deadline of the retransmission
U	Utilization by periodic real-time traffic
M	number of retransmission RT channels

where $L_{T,i}$ is the amount of pure data per message belonging to RT channel τ_i . Of the $N_{\text{pack},i}$ packets, the $N_{\text{pack_max},i}$ first packets are maximum sized packets, where:

$$N_{\text{pack_max},i} = \left\lfloor \frac{L_{T,i}}{L_{\text{data}}} \right\rfloor. \quad (5)$$

Furthermore, we define $L_{\text{last},i}$ as the length of the last packet of a message, including header, if being shorter than L_{pack} , otherwise as zero:

$$L_{\text{last},i} = \begin{pmatrix} N_{\text{pack},i} - N_{\text{pack_max},i} \\ (L_{T,i} - N_{\text{pack_max},i} \cdot L_{\text{data}} + L_{\text{header}}) \end{pmatrix}. \quad (6)$$

In Fig. 3, the timing of a message transmission is introduced. The transmission time of a full-length packet is:

$$T_x = \frac{L_{\text{pack}}}{R} \quad (7)$$

Figure 3. Timing of a message transmission

where R is the bit rate of the physical link. If the last packet is shorter than L_{pack} , the transmission time of this packet is:

$$T_{x_last, i} = \frac{L_{last, i}}{R} \quad (8)$$

Also shown in the figure are the transmission time of an ACK packet, $T_{ACK, i}$, and the propagation delay over the physical link, T_{prop} . The total transmission time of a message is:

$$T_{x_tot, i} = \frac{N_{pack_max, i} \cdot L_{pack} + L_{last, i}}{R} \quad (9)$$

The timing of the original transmission of a message, including ACK, is further detailed in Fig. 4. We denote the processing time between the reception of a packet until its corresponding ACK packet is sent as $T_{proc, 1}$. Furthermore, we denote the processing time between the timeout without reception of an ACK packet until the retransmission can be initiated as $T_{proc, 2}$. Negative ACK (NACK) could also be used but would not change the worst-case timing analysis.

We assume all packets of a message having the same timeout instance since we will only retransmit one packet of a message at the most. If we at the timeout instance encounter the rare occasion that more than one packet is lost, it is pointless to retransmit any packet belonging to the message since the message will still be incomplete. The timeout instance is therefore defined in relation to the deadline of the ordinary message transmission:

$$T_{timeout} = T_{D_ord, i} - T_{proc, 2} \quad (10)$$

Further, we define a safety margin between the expected ACK reception and the timeout instance as T_{margin} . The queuing delay caused by interference with the transmission of other messages will be elaborated on in the next section.

IV. REAL-TIME SCHEDULING ANALYSIS

Below, we will introduce the real-time scheduling analysis in order to analyze the queuing delay. We assume packets are

Figure 4. Timing when retransmission is included

queued in order of increasing deadlines, i.e., the EDF policy is used. Fig. 5 exemplifies the queuing delay with EDF queuing. The real-time scheduling analysis only considers the queuing delay, including the transmission time $T_{x_tot, i}$ of all packets of a message. Therefore, we need to subtract all other delays from the delay bound to isolate the maximum queuing delay, $T_{d_ord, i}$. We calculate the maximum queuing delay (the deadline of the ordinary transmission) as:

$$T_{d_ord, i} = T_{D_ord, i} - 2 \cdot T_{prop} - T_{proc, 1} - T_{proc, 2} - T_{ACK} - T_{margin} - T_x \quad (11)$$

where T_x is added as the maximum blocking time of one packet due to already initiated (non-preemptive) transmission of a packet with longer deadline.

As stated above, we assume a dedicated RT channel for retransmissions. We denote such a RT channel as $\tau_{retr, i} = \{m_{retr, S, i}, m_{retr, D, i}, P_{retr, i}, L_{retr, i}, D_{retr, i}\}$. In our case of a point-to-point link, we will only have one such RT channel, $\tau_{retr, 1}$ (see Section 5 for an extension with several retransmission RT channels). The period of the RT channel, $P_{retr, 1}$, is a system parameter with which we can set the minimum time between two retransmission requests that shall be guaranteed, i.e., how often we want to be able to retransmit a packet of an arbitrary message.

Since we want to guarantee the retransmission of one packet belonging to any of the RT channels, we start by defining the maximum length of a packet belonging to RT channel τ_i as:

Figure 5. Example of queuing delay. Message B arrives to the queue after Message A, but with an earlier deadline, thereby delaying packets of Message A. Packets B1 and A3 are retransmitted in the example.

$$L_{\max,i} = \begin{cases} L_{\text{pack}}, & N_{\text{pack_max},i} \geq 1 \\ L_{\text{last},i}, & N_{\text{pack_max},i} = 0 \end{cases} \quad (12)$$

while the corresponding maximum transmission time is:

$$T_{x_max,i} = \frac{L_{\max,i}}{R} \quad (13)$$

We can then get the longest packet length among all RT channels as:

$$L_{\text{retr},1} \geq \text{MAX}_{i=1}^Q (L_{\max,i}) \quad (14)$$

while the corresponding maximum transmission time among all RT channels is:

$$T_{x_retr,1} = \frac{L_{\text{retr},1}}{R} \quad (15)$$

In other words, $T_{x_retr,1}$ will always correspond to L_{pack} as long as at least one RT channel has a message length of at least one full-size packet.

The delay bound for the retransmission RT channel, $D_{\text{retr},1}$, is a system parameter with which we can set the time allocated for possible retransmission. We therefore set:

$$T_{D_ord,i} = D_{\text{retr},1} \quad 1 \leq i \leq Q \quad (16)$$

Equation (1) can be rewritten in the following way to derive the delay bound for ordinary transmissions:

$$T_{D_ord,i} = D_{T,i} - T_{D_retr,i} \quad (17)$$

To isolate the maximum queuing delay (the deadline of the retransmission), $T_{d_retr,1}$, we subtract the other delays from the delay bound:

$$T_{d_retr,1} = D_{\text{retr},1} - T_{\text{prop}} - T_x \quad (18)$$

where T_x again is added as the maximum blocking time of one packet due to already initiated (non-preemptive) transmission of a packet with longer deadline.

The next step is to check the utilization of the point-to-point link. A necessary, but not sufficient, condition for the traffic allocation over the link to be feasible is that its utilization is no higher than 1 at all times, taking into account both the ordinary RT channels and the retransmission RT channel. In line with EDF scheduling theory [9], the utilization U of periodic real-time traffic is therefore calculated in the following way:

$$U = \sum_{i=1}^Q \left(\frac{T_{x_tot,i}}{P_{T,i}} \right) + \frac{T_{x_retr,1}}{P_{\text{retr},1}} \quad (19)$$

In order to be able to ensure the feasibility of the traffic allocation, a further constraint has to be fulfilled. However, for the description of the feasibility check, a few concepts have to be defined. The hyperperiod HP is the least common multiple of all periods of the RT channels, i.e., the length of time from when all tasks' periods start at the same time, until they start at the same time again. The busperiod BP is any interval within HP in which the resource, in our case the link, is not idle. The traffic demand over the point-to-point link corresponds to the processor demand in a real-time system and is defined by the workload function $h(t)$. Due to the assumption of EDF scheduling, the feasibility analysis suggested in [10] can be used in a similar manner for our network. $h(t)$ is calculated as the sum of the transmission times for all message instances of all real-time channels with an absolute deadline less than or equal to a point in time t , where t signifies the number of time units elapsed since the beginning of HP [11] [12]. $h(t)$, including both the ordinary RT channels and the retransmission RT channel, is computed as follows.

$$h(t) = \sum_{\substack{i \in [1,Q], \\ T_{d_ord,i} \leq t}} \left(1 + \left\lfloor \frac{t - T_{d_ord,i}}{P_{T,i}} \right\rfloor \right) \cdot T_{x_tot,i} + \mathbf{I}(T_{d_retr,1} \leq t) \cdot \left(1 + \left\lfloor \frac{t - T_{d_retr,1}}{P_{\text{retr},1}} \right\rfloor \right) \cdot T_{x_retr,1} \quad (20)$$

where $\mathbf{I}(A) = 1$ if A is true, otherwise zero.

The second constraint mentioned to ensure the feasibility of the schedule was introduced by Spuri [11] [12], and demands that:

$$h(t) \leq t \quad \forall t \quad (21)$$

This restriction introduces a high computational workload, but the number of instances of evaluation can be reduced to the number of integer time values during an interval upper bounded by BP_1 , the first BP in the first HP of the schedule where all periods start at time zero. In other words,

$$t \in \bigcup_{i=1}^Q \{m \cdot P_{T,i} + T_{d_ord,i} : m = 0,1,2,\dots\} \cup \{n \cdot P_{\text{retr},1} + T_{d_retr,1} : n = 0,1,2,\dots\} \quad (22)$$

where

$$t \in [1, BP_1] \quad (23)$$

TABLE II. PARAMETERS OF THE FOUR DIFFERENT TRAFFIC CLASSES USED IN THE SIMULATIONS

Traffic class	$P_{T,i}$	$D_{T,i}$	$L_{T,i}$
1	200 μ s	200 μ s	4 000 bits
2	400 μ s	400 μ s	4 000 bits
3	800 μ s	800 μ s	4 000 bits
4	1 600 μ s	1 600 μ s	4 000 bits

Only when both the utilization constraint and the feasibility constraint are fulfilled can the real-time requirements be guaranteed to be met.

V. EXTENSION TO SUPPORT MORE RETRANSMISSIONS

In this section, we describe how to extend the framework to support several retransmission RT channels for our communication channel. By having M retransmission RT channels, each supporting the retransmission of one packet every period of $P_{retr,i}$ and with a bounded delay of $D_{retr,i}$, we can improve the support of retransmissions. More concretely, we will be able to support up to M packet retransmissions in a row over the physical channel. This upper limit of M packets holds when none of the M retransmission RT channels has been in use for their respective last period of $P_{retr,i}$. More generally, any retransmission RT channel $\tau_{retr,i}$ can be used for a packet retransmission if it has not been in use for the last period of $P_{retr,i}$. As stated above, the retransmission RT channel delay bound is a system parameter with which we can set the time allocated for possible retransmission. We set this delay bound, $D_{retr,i}$, to the same value for all retransmission RT channels. Equation (19) to calculate the utilization is updated in the following way to support several retransmission channels:

$$U = \sum_{i=1}^Q \left(\frac{T_{x_tot,i}}{P_{T,i}} \right) + \sum_{i=1}^M \left(\frac{T_{x_retr,i}}{P_{retr,i}} \right). \quad (24)$$

Furthermore, the workload function in (20) is updated in the following way:

$$h(t) = \sum_{\substack{i \in [1,Q], \\ T_{d_ord,i} \leq t}} \left(1 + \left\lfloor \frac{t - T_{d_ord,i}}{P_{T,i}} \right\rfloor \right) \cdot T_{x_tot,i} + \sum_{\substack{i \in [1,M], \\ T_{d_retr,i} \leq t}} \left(1 + \left\lfloor \frac{t - T_{d_retr,i}}{P_{retr,i}} \right\rfloor \right) \cdot T_{x_retr,i}. \quad (25)$$

Otherwise, the same real-time scheduling analysis as described above can be used to analyze the worst-case queuing delay.

VI. SIMULATION ANALYSIS

To evaluate the performance of the ARQ scheme, we have conducted two types of simulations. The first type of simulation is a schedulability analysis to establish how high utilization of the communication channel we can reach for hard

real-time data traffic, excluding possible retransmissions. In other words, we generate requests for RT channels and see how many can be accepted according to the real-time scheduling analysis. We compare the utilization when using our ARQ scheme with the case of only having ordinary RT channels without any retransmission scheme at all.

The second type of simulation is a packet-level simulation where actual traffic (generated by the first type of simulation) over the communication channel is simulated. In this way, we can get measurements of the average message error rate, both with and without our ARQ scheme.

The simulator, including both types of simulations, is implemented in MatLab. For all simulations we have assumed a full-duplex 100 Mbit/s (in each direction) physical link, a bit error rate of $BER = 10^{-6}$, a maximum packet length of $L_{pack} = 1\,000$ bits, a propagation delay in one direction of $T_{prop} = 1\,\mu$ s (≈ 200 m, i.e., many embedded systems will have an even shorter propagation delay), and the length of each ACK packet to be $L_{ACK} = 100$ bits. The parameters T_{proc_1} , T_{proc_2} and T_{margin} are assumed to be negligible in the simulations and are therefore set to zero. Moreover, the ACK packets are assumed to have enough redundancy added to have negligible error rate. The parameters for each generated ordinary RT channel are always taken from one of four different traffic classes. Table II shows the parameter values for the traffic classes. The choice of traffic class for each RT channel is made randomly with even distribution.

Fig. 6 shows the simulation results when using one retransmission channel with the parameters $P_{retr,i} = 1\,600\,\mu$ s, $D_{retr,i} = 30\,\mu$ s, and $L_{retr,i} = 1\,000$ bits. As shown, when introducing our retransmission scheme for this case, the utilization for ordinary RT channels is just reduced a few percents at saturation. Nevertheless, a reduction of the message error rate of almost an order of magnitude is achieved. For each value on the x-axis, the y-axis value is the average of 100 simulations, each with a simulated duration of 50 hyperperiods.

In Fig. 7, it is shown how the performance changes when the period of the retransmission channel is reduced to $P_{retr,i} = 200\,\mu$ s to accommodate more retransmissions. As seen, the utilization overhead is slightly increased, while the message error rate is significantly further reduced. For each value on the x-axis, the y-axis value is the average of 200 simulations, each with a simulated duration of 100 hyperperiods. The reason for more and longer simulations is that message errors are rarer.

Fig. 8 shows how we can obtain even better results by having several retransmission RT channels. In this case, three retransmission RT channels are used, each with the parameters $P_{retr,i} = 600\,\mu$ s, $D_{retr,i} = 50\,\mu$ s, and $L_{retr,i} = 1\,000$ bits. The reduction of the message error rate when using our retransmission scheme is now approximately three orders of magnitude. The utilization overhead is slightly higher compared to the former cases, but can still be motivated by the large improvements in reliability. Moreover, even if the possible amount of hard real-time data traffic is reduced, the rest of the communication capacity is not used solely for retransmissions. The reason is that the retransmission RT channels have rather short delay bounds compared to their periods. This gives a high amount of allocated communication

Figure 6. Simulation results when having one retransmission channel with the parameters $P_{retr,i} = 1\,600\,\mu s$, $D_{retr,i} = 30\,\mu s$, and $L_{retr,i} = 1\,000$ bits. The utilization (a) and the message error rate (b) are shown for both a link with our retransmission scheme (unbroken lines) and a link without it (dashed lines).

Figure 7. Simulation results when having one retransmission channel with the parameters $P_{retr,i} = 200\,\mu s$, $D_{retr,i} = 30\,\mu s$, and $L_{retr,i} = 1\,000$ bits. The utilization (a) and the message error rate (b) are shown for both a link with our retransmission scheme (unbroken lines) and a link without it (dashed lines).

capacity for retransmissions. However, even if this capacity cannot be allocated for other delay bounded traffic, it can still be used for other non-delay bounded traffic when no delay bounded traffic is transmitted. For each value on the x-axis, the y-axis value is the average of 1 600 simulations, each with a simulated duration of 100 hyperperiods.

VII. CONCLUSIONS

In this paper, we have presented a framework for reliable communication by using a retransmission scheme at the same time as delay bounds for both ordinary transmissions and allowed retransmission attempts are guaranteed. The framework has been exemplified through a point-to-point link with EDF scheduling. We have evaluated the performance of the retransmission scheme through simulations, both in terms of utilization overhead and improvement in message error rate. Our simulation study has shown that a reduction of the message error rate by several orders of magnitude is possible with a reasonable utilization overhead. We plan to extend the

framework to address multi-hop networks and other scheduling policies as well.

REFERENCES

- [1] S. Pejhan, M. Schwartz, and D. Anastassiou, "Error control using retransmission schemes in multicast transport protocols for real-time media," *IEEE/ACM Transactions on Networking*, vol. 4, no. 3, pp. 413-427, June 1996.
- [2] E. Uhlemann, P.-A. Wiberg, T. M. Aulin, and L. R. Rasmussen, "Deadline dependent coding - a framework for wireless real-time communication," *Proc. 7th International Conference on Real-Time Computing Systems and Applications (RTCSA 2000)*, Cheju Island, South Korea, Dec. 12-14, 2000, pp. 135-142.
- [3] E. Uhlemann and L. K. Rasmussen, "Incremental redundancy deadline dependent coding for efficient wireless real-time communications," *Proc. 10th IEEE Conference on Emerging Technologies and Factory Automation (ETFA 2005)*, Catania, Italy, Sept. 19-22, 2005, vol. 2, pp. 417-424.
- [4] M. M. Butt, "Provision of guaranteed QoS with hybrid automatic repeat request in interleaved division multiple access systems," *10th IEEE Singapore International Conference on Communication Systems (ICCS 2006)*, Singapore, Oct. 30 - Nov. 1, 2006, pp. 1-5.

Figure 8. Simulation results when having three retransmission channels, all with the parameters $P_{retr,i} = 600 \mu s$, $D_{retr,i} = 50 \mu s$, and $L_{retr,i} = 1\,000$ bits. The utilization (a) and the message error rate (b) are shown for both a link with our retransmission scheme (unbroken lines) and a link without it (dashed lines). Remark the logarithmic scale on the y-axis of the graph to the right.

- [5] G. Giancola, S. Falco, and M. G. Di Benedetto, "A novel approach to error protection in medium access control design," Proc. 4th International Workshop on Mobile and Wireless Communications Network (MWCN 2002), Stockholm, Sweden, Sept. 9-11, 2002, pp. 337-341.
- [6] X. Fan, "Real-time services in packet-switched networks for embedded applications," Doctoral thesis, Department of Computer Engineering, Chalmers University of Technology, Göteborg, Sweden, June 2007.
- [7] Q. Zheng and K. G. Shin, "On the ability of establishing real-time channels in point-to-point packet-switched networks", IEEE Transactions on Communications, vol. 42, no. 2/3/4, pp. 1096-1105, Feb./ Mar./Apr. 1994.
- [8] D. Ferrari and D. C. Verma, "A scheme for real-time channel establishment in wide-area networks", IEEE Journal on Selected Areas in Communication, vol. 8, no.3, pp. 368-379, Apr. 1990.
- [9] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment", Journal of the ACM, vol. 20, no. 1, pp. 46-61, Jan. 1973.
- [10] H. Hoang and M. Jonsson, "Switched real-time Ethernet in industrial applications - deadline partitioning", The 9th Asia-Pacific Conference on Communication (APCC 2003), Penang, Malaysia, Sept. 21-24, 2003, vol. 1, pp. 76-81.
- [11] M. Spuri, "Analysis of deadline scheduled real-time systems", Technical Report RR No. 2772, Institut National de Recherche en Informatique et en Automatique (INRIA), France, 1996.
- [12] J. A. Stankovic, M. Spuri, K. Ramamritham, and G. C. Buttazzo, Deadline scheduling for real-time systems - EDF and related algorithms. Kluwer Academic Publishers, Boston, MA, USA, 1998.