P R E P R I N T S

ECRTS
July 1-3
Dublin, Ireland
2009

8th Intl. Workshop on

Real Time Networks RTN'09

Dublin, Ireland, June 30, 2009
in conjunction with the 21st ECRTS

http://www.cister.isep.ipp.pt/rtn09/

Co-Chairman:

Eduardo Tovar
CISTER/ISEP Polytechnic Institute of Porto

Jean-Dominique Decotignie
Swiss Center for Electronics and Microtechnology

Program Committee

Advance Program

9:00-9h30	**Registration**
9:30-10h30	**Session 1 - Keynote Talk 1** **(Chair: Eduardo Tovar; Rapporteur: Zheng Shi)** *Temporal, Functional, and Diagnostic Analysis in Sensor Networks* Tarek Abdelzaher University of Illinois at Urbana-Champaign, USA
10h30-11h00	**Coffee Break**
11h00-12h30	**Session 2** **(Chair: Michael Short; Rapporteur: Marcelo Sobral)** *Towards Optimised Retransmission Reservation in Real-Time Wireless TDMA* Mark Gleeson, Stefan Weber Trinity College Dublin, Ireland *A Proposal for a Notion of Timeliness in Wireless Sensor Networks* Ramon Serna Oliver, Gerhard Fohler TU Kaiserslautern, Germany *A Vision of Cyber-Physical Internet* Anis Koubaa, Bjorn Andersson Polytechnic Institute of Porto, Portugal
12h30-14h00	**Lunch**
14:00-15h00	**Session 3 - Keynote Talk 2** **(Chair: Jean-Dominique Decotignie; Rapporteur: Jean-Luc Scharbarg)** *Real-Time Internet for Buildings?* Peter van der Stok Philips Research Laboratories Eindhoven, The Netherlands
15h00-16h00	**Session 4** **(Chair: Björn Andersson; Rapporteur: Ivan Tarasov)** *A Real-Time and Robust Routing Protocol for Building Fire Emergency Applications Using Wireless Sensor Networks* Yuanyuan Zeng, Cormac J. Sreenan University College Cork, Ireland *Assessing the Clusters Formation in the HCT-MAC Protocol* Marcelo Maia Sobral, Leandro Buss Becker Federal University of Santa Catarina, Brazil
16h00-16h30	**Coffee Break**
16h30-17h30	**Session 5** **(Chair: Zdenek Hanzalek; Rapporteur: Ramon Serna Oliver)** *Design and Development of a Reliable Ethernet-based Real-Time Communication Protocol* Greg Bollella, Mike Duigou, Ivan Tarasov Sun Microsystems, United States *Improving Information Throughput in CAN Networks: Implementing a Dual Speed Approach* Imran Sheikh, Michael Short University of Leicester, United Kingdom
17h30	**Closing Remarks**

Session 1 – Keynote Talk 1

Chair: Eduardo Tovar

Rapporteur: Zheng Shi

Temporal, Functional, and Diagnostic Analysis in Sensor Networks

Tarek Abdelzaher

University of Illinois at Urbana-Champaign, USA

Abstract: Sensors networks are a precursor of distributed cyber-physical systems; an emerging category of distributed systems marked by increased interactions with an external physical environment and the convergence of computation, communication, sensing and control. Analysis of temporal and functional behavior of such systems is complicated by virtue of distribution and interactions among large numbers of computational, communication, and physical components. This talk envisions cyber-physical systems of the future, presents challenges in analyzing their behavior, discusses instances of bad component interactions, and highlights tools for diagnosing root causes of problems. Application examples are given from common sensor networks software, as well as recent experimental deployments. Emerging solutions are covered that range from a new schedulability analysis theory for distributed systems to the application of data mining techniques for exploring anomalous behavior. The talk concludes with recommendations for future research on cyber-physical computing.

Session 2

Chair: Michael Short

Rapporteur: Marcelo Sobral

Towards Optimised Retransmission Reservation in Real-Time Wireless TDMA

Mark Gleeson
Distributed Systems Group
Department of Computer Science
Trinity College Dublin
Ireland
gleesoma@cs.tcd.ie

Stefan Weber
Distributed Systems Group
Department of Computer Science
Trinity College Dublin
Ireland
sweber@cs.tcd.ie

***Abstract*—Providing real-time guarantees in wireless networks requires the reservation of transmission time not only for transmissions but also for retransmissions. Retransmissions are required to ensure that real-time reliability targets are achieved due to the unreliable nature of the wireless medium.**

Simplistic approaches such as immediate retransmission over a number of times are wasteful in terms of available transmission time and bandwidth. We propose a resource allocation and retransmission mechanism that takes into account the independence of transmissions to different destinations and the characteristics of the occurrence of burst errors.

In our evaluation, we will show that we are able to reduce the time that must to be reserved for retransmissions by exploiting the knowledge about destinations of transmission and the rearranging of transmission schedules.

I. Introduction

Our real-time medium access control protocol, Hierarchical Distributed Time Devision Multiple Access, HD-TDMA, provides stations with a mechanism to reserve a transmission slot in a TDMA cycle in a wireless ad hoc network. In order for stations to satisfy real-time requirements for transmissions, an admissions component needs to verify that sufficient transmission time is available to satisfy a requested reliability. The calculation of the required transmission time needs to take into account the number of retransmissions that may be required to achieve a given reliability.

The shared nature of wireless medium allows transmissions to be interfered with by transmissions from other sources or general noise in the environment. The interference from other transmissions is generally addressed by employing protocols such as Time Division Multiple Access (TDMA) [1]–[3]. The remaining interference is through noise in the environment takes the form of burst errors of varying durations [4], [5].

In the presence of burst errors, the attempt to immediately retransmit a failed frame has a high probability of failure as the burst error condition that has caused the failure may still be present. Carrier Sense Multiple Access (CSMA) protocols, which view all frame loss as being caused by contention, will attempt to resend a failed frame immediately or after a slight pause.

This approach to retransmission may be adequate for communication over shared wired media where transmissions to two different stations will use the same medium and be exposed to the same environmental factors.

However, where a wireless medium is in use this may not be the case. On a wireless medium, signals to two stations in different locations may propagate over different paths and encounter different environmental factors. This means that even though a transmission to one station may fail because of environmental factors, a transmission to another station will not be affected by these factors.

Based on this observation, we propose a protocol that takes into account the specific characteristics of the wireless medium and attempts to reorder transmission queues following failed transmissions. It will react to the failure of a transmission by postponing transmissions to a given station and promoting transmissions to the front of the queue that have a higher likelihood to succeed.

In order to support the retransmission of frames, a station will require additional transmission time. Our protocol employs a probabilistic admission control process to ensure that sufficient time is reserved for retransmissions to satisfy a requested delivery reliability.

We propose an approach based on binomial distribution which minimises the transmission time resources that must be allocated to meet real-time reliability guarantees by clustering transmissions not only by destination but also by similar probability.

In this paper, we focus on the description of the admission control process and the analytical reasoning behind the time that is reserved for retransmissions. For a complete description of the protocol framework, please see our technical report [6].

The remainder of this paper is laid out as follows: In section II, we will present a brief overview of our HD-TDMA protocol. Section III discusses the causes of transmission failures. This will be followed by an analytical discussion of the time that needs to be reserved for retransmissions in section IV. In section V, we will present an evaluation of our protocol and in section VI, we will present our conclusions and possible future work.

II. Overview of HD-TDMA

The basic structure of the HD-TDMA MAC protocol consists of a TDMA-slotted structure. Each slot is of fixed duration and may contain an arbitrary number of transmissions, thus facilitating variable frame sizes and the ability to implement acknowledged and unacknowledged transmissions. Each transmitting station is allocated at least one slot within which it can transmit a number of frames.

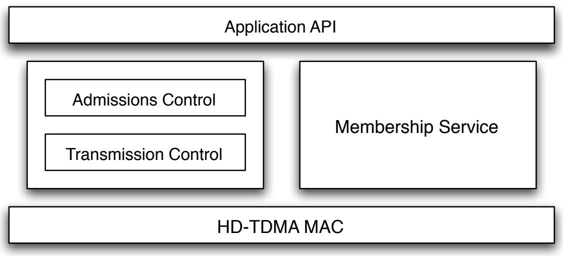


Figure 1. High level view of system

Figure 1 shows the basic layout of our communication architecture. It consists of a MAC layer, a membership service, a combination of an admission and transmission control and an API to interface with both the membership service and the admissions control mechanism. The admissions and transmission control together implement a localised decision process and control together with the membership service the operation of the underlying MAC layer.

A. Slots

Figure 2 illustrates a possible configuration of a HD-TDMA cycle with an N-slot TDMA cycle, the contents of one slot having been expanded. Transmissions within a slot begin with a compulsory beacon frame. Subsequently, a number of frames of varying sizes are sent, followed by space reserved for the recovery from transmission failures.

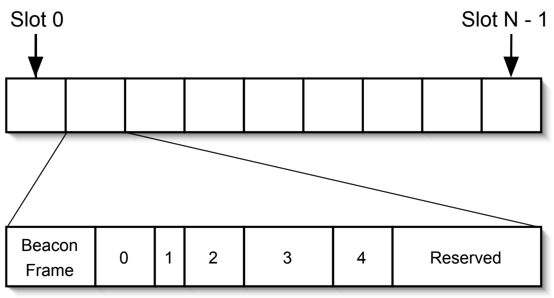


Figure 2. Relationship between a slot, its contents, and the TDMA cycle

B. Admissions Control

We employ a probabilistic admissions control protocol to ensure sufficient transmission time is reserved to accommodate an estimated number of retransmissions. The transmission time reserved for retransmissions will vary depending on factors such as frame size, transmission speed, reliability requirements and number of frames. An evaluation is made at the point of admission based on the characteristics of currently admitted frames and the frame seeking admission.

As a contention free transmission schedule is place, any transmission failure which occurs must be attributed to a dynamic propagation effect. Each failure to receive an acknowledgment or receipt of negative acknowledgment provides the scheduler with channel information.

The admissions control process also considers real-time requirements such as deadlines. All frames that are to be accepted, must have a deadline later than the end of the slot time as to allow reordering of transmissions within a slot at runtime.

III. Burst Error Characteristics

Burst errors are characterised by two separate distributions: Firstly, a distribution of occurrence. Burst errors occur intermittently with a significant time between each burst event. Secondly, a distribution of the length of a burst upon occurrence which has been found experimentally to follow a long-tailed distribution [5].

We consider a channel to be similar in nature to the contention and collisions found in a multiuser, random access communications system. A non real-time data network is characterised by low bandwidth utilisation with occasional random periods of varying but short duration in comparison to the idle time of heavy utilisation. This aligns with

- Errors occur in bursts
- Random occurrence
- Lack of global knowledge
- Burst errors are detected as frame losses just like collisions and contention

Instead of contending with other stations, we are contending with the medium itself. Since the path to each station is different, the propagation effects will also be different. We consider the medium characteristics to be independent for each station.

A. Dependence of Transmissions

If a transmission is made by station A to station B and that transmission fails a retransmission will be required. If a retransmission happens within a short time frame of the failed transmission, there is a greater probability that the transmission will also be lost due to the dynamic behaviour of a channel, bursty error or channel fading due to propagation effects. Practical evaluation by Willig et al [5] showed a 0.71 probability of a further failure following a failure for the evaluated scenario; highlighting the lack of independence of transmissions following a failure.

IV. Calculation of Transmission Time

We consider a fully connected single hop wireless communication structure. All stations are within communication range of each other. As previous work has shown [8], [9], a statistical independence exists for transmissions to two destinations as a result of different propagation paths. Therefore the probability of a successful transmission, $p_{success}$ will vary depending on the source and destination of a transmission. For transmissions between three stations, a, b and c independence results in the following: $p_{success_{ab}} \neq p_{success_{ac}}$ and as wireless communication is not symmetric, $p_{success_{ab}} \neq p_{success_{ba}}$

In an unacknowledged communication system, for a given probability of an error, the number of transmissions required to achieve a defined level of reliability can be determined. However, even if the first attempt succeeds, all the transmissions must be completed as there is no positive feedback as to the success of any individual transmission attempt. We consider a scenario, where all transmissions are acknowledged: thus when an acknowledgment frame is received correctly, no further transmission attempts are made.

We discuss three approaches to determine the number of transmissions required;

- Dedicate time for each individual frame, so that the frame in question may be sent a number of times to meet the reliability requirement.
- Pool the transmission resources for all frames destined for the same destination through the application of binomial distribution.
- Cluster transmission resources for destinations with similar probabilities.

As we combine the transmission requirements, we aim to continue to meet the reliability requirements of transmissions while progressively reducing the overall transmission time allocation for retransmissions.

A. Individual Transmissions

For an acknowledged transmission to be received correctly with probability p_{target}, different probabilities need to be assigned to each element of the communication: the transmission p_{tx}, the failure of the acknowledging station, p_{node_fail} as well as the probability of the successful reception of the acknowledgment frame p_{ack}. These can be combined to give the probability of an error p_{err} as in equation 1.

$$p_{err} = (1 - p_{tx}) \cdot p_{node\_fail} \cdot (1 - p_{ack}) \tag{1}$$

Given that the medium is unreliable, upon failure to receive an acknowledgment, a number of retransmission attempts may be required. Transmission resources must be reserved in advance based on the observed reliability of communication to the destination in question, in order to meet the reliability requirement of the frame. Thus following $n_{attempts}$ attempts the probability that no attempt will have succeeded, p_{fail} is given by equation 2.

$$p_{fail} = p_{err}^{n_{attempts}} \tag{2}$$

Alternatively by taking logs of equation 2 expressed in terms of attempts, $n_{attempts}$, substituting $1 - p_{target}$ for p_{fail}, equation 3 is arrived at.

$$\left\lceil \frac{log(1 - p_{target})}{log(p_{err})} \right\rceil = n_{attempts} \tag{3}$$

The suboptimal nature of this result is clear by the requirement to round up $n_{attempts}$ to an integer value.

B. Transmissions to Same Destination

Considering each frame separately results in excessive resource requirements. We therefore seek to group frames to the same destination together. As there maybe more than one frame, a binomial distribution may be used similar to the approach by Demarch et al [10]. The binomial distribution requires each attempt be a Bernoulli trial, such that each transmission be independent from previous attempts and that the probability of each attempt is constant.

However, wireless communication is subject to significant temporal variations in transmission reliability due to issues such as channel fading, channel propagation, mobility, etc. In the context of our MAC protocol HD-TDMA [6] a per destination deferral process is utilised within each transmission slot following a failure, restoring independence between attempts and enabling the use of a binomial process.

We introduce the following assumptions:

- All frames have the same length
- All frames are acknowledged
- Frames to the same destination have the same probability of success

As positive feedback exists through the acknowledgment frame when a transmission is successful, the goal is to obtain at least k successes from n possible attempts, where k is the number of frames to be transmitted. For each destination we model this problem using a binomial distribution. The probability that exactly y transmissions from n attempts will succeed, is given by equation 4.

$$p_{success} = \binom{n}{y} p^y (1 - p)^{n-y} \tag{4}$$

Forming the cumulative density function, we sum the probabilities of all valid results, where the delivery requirements are satisfied, e.g at least k successes.

$$p_{success} = \sum_{y=k}^{n} \binom{n}{y} p^y (1 - p)^{n-y} \tag{5}$$

Our goal is to minimise the transmission resources, we allocate in advance of transmission. By an iterative approach the target value of p_{target} is achieved by increasing the value of n, the value of $p_{success}$ achieved will exceed p_{target} by small amount. For equal $p_{success}$, n will be less than or equal to the sum of transmissions as determined by equation 3, or for equal n, $p_{success}$ will be greater in the binomial approach than the individual approach.

This calculation is repeated for each destination and the sum of n determined, which is the total number of transmissions required to meet the reliability requirement and thus the time that must be reserved by the admissions control process is known.

C. Varied Probabilities

The probability delivered by adopting the iterative approach in the previous section will be greater than or equal to that required as the number of transmissions is an integer value. We define this excess to be the residual;

$$residual = p_{target} - p_{success} \tag{6}$$

While this represents the optimal solution for a single destination, for multiple destinations the sum of residuals becomes significant.

An initial setup cost also exists within the binomial distribution, as the number of frames to be transmitted increases the ratio between transmissions and retransmissions reduces.

With these two concerns in mind we propose an approach to minimise the number of transmissions calculated to maintain the target reliability.

For transmissions to two stations, a and b, where the probability of transmission success are equal such that, $p = p_a = p_b$, the application of equation 5 will result in n_a and n_b transmissions respectively, where y_a and y_b successes where required. If combined as a single binomial distribution the overall number of transmissions is given by equation 7 as n_{ab}.

$$p_{success} = \sum_{y_a+y_b=k}^{n_{ab}} \binom{n_{ab}}{y_a+y_b} p^{(y_a+y_b)}(1-p)^{(n-y_a-y_b)} \tag{7}$$

As a result of the residual factor and the setup cost of the binomial, it can be shown empirically that equation 8 holds for $p_a = p_b$ independent of the number of transmissions made by hosts a or b.

$$n_{ab} \leq n_a + n_b \tag{8}$$

We may state that for an unknown value of $\epsilon \geq 0$, equation 8 continues to hold.

$$p_b = p_a + \epsilon \tag{9}$$

We propose a search algorithm to cluster groups of destinations with similar probabilities together applying the binomial distribution to the cluster using the lowest probability of success of cluster members.

The clustering algorithm is based on a progressive clustering of destinations or groups of destinations with similar transmission success probabilities. The algorithm begins with each destination considered to be a cluster with a single member, the sum of transmissions required in this case replicates the outcome of section IV-B.

Each round of the algorithm identifies the cluster pair with the least difference in transmission success probability and merges them. A hierarchy is formed by progressively merging clusters in this manner. Within each cluster the destination with the lowest transmission probability becoming the cluster head.

Equation 5 is applied after each round to determine the number of transmissions, with the probability of the cluster head as the p parameter and where y is the sum of frames to be transmitted by all cluster members.

As the lowest probability within the cluster is used as an input into equation 5 the reliability requirements of all cluster members are ensured. After each cluster is formed the overall sum of transmissions across all clusters is determined.

The clustering algorithm continues until there exists only one cluster, that all destinations are considered together using the lowest transmission success probability within the cluster. The lowest calculated sum of transmissions determined during the clustering process is returned. This value is ensured to be less than or equal to the result of section IV-B.

D. Alternative Approaches

The approach proposed in section IV-C requires a computationally significant search with no stopping condition when the optimum solution is found.

An alternative approach is to model the distribution of probabilities of transmission success as a continuous beta distributed variable allowing the application of the beta binomial distribution [11].

However, real-time guarantees would have to be weakened to consider transmissions over a number of slots in order to form a sufficiently continuous function to employ a beta distribution.

V. Evaluation

Our evaluation compares the three options presented in section IV, where each transmission is considered separately, where transmissions to each destination are considered as a group and where the clustering approach was employed.

The scenarios consider between 3 and 5 destinations seeking to transmit either 1 or 2 frames each, table I lists the number of frames for each destination for 6 scenarios.

Each scenario consists of 100 trials in which each destination is assigned a random probability of success between 0.45 and 0.95. The total number of transmissions required to meet the target reliability of 0.95 is then calculated. Results are presented ordered by the transmission count determined for the individual case.

Table I
SCENARIO PARAMETERS

Set	Dest 1	Dest 2	Dest 3	Dest 4	Dest 5
a	1	2	1	0	0
b	1	2	1	1	1
c	1	2	2	0	0
d	1	1	1	1	0
e	2	1	2	0	0
f	1	1	1	1	1

Table II presents an overview of the results from the 6 evaluated scenarios, showing the total number of transmissions calculated from 100 trials. In all cases the inefficiency of considering each transmission separately is clear. Our evaluation shows that clustering approach results in an improvement of between 6.87% and 14.11% compared to considering transmissions by destination alone.

Table II
SCENARIO RESULTS

Set	Individual	By Destination	Clustered	Reduction
a	3985	1150	1062	7.65%
b	5181	1476	1331	9.82%
c	4702	1354	1261	6.87%
d	4488	1271	1123	11.64%
e	4790	1371	1275	7.00%
f	5513	1573	1351	14.11%

The inefficiency of considering each frame individually is shown in Figure 3, with the benefit offered by the binomial approaches clear. Figures 4-8 show the clustered approach consistently provides a improvement over considering transmissions solely by destination. In some cases a reduction of up to 4 transmissions is recorded representing a reduction of 20% in transmission time which must be allocated.

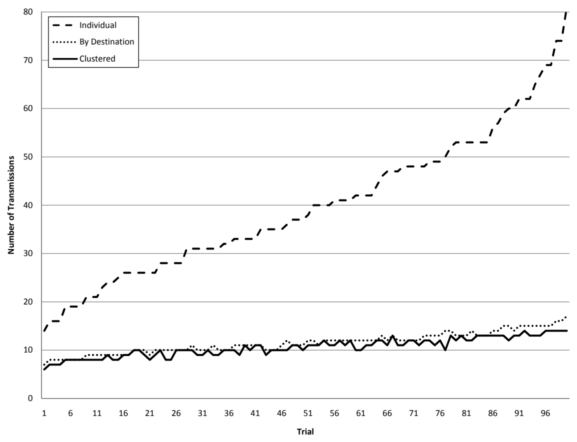


Figure 3. Scenario a

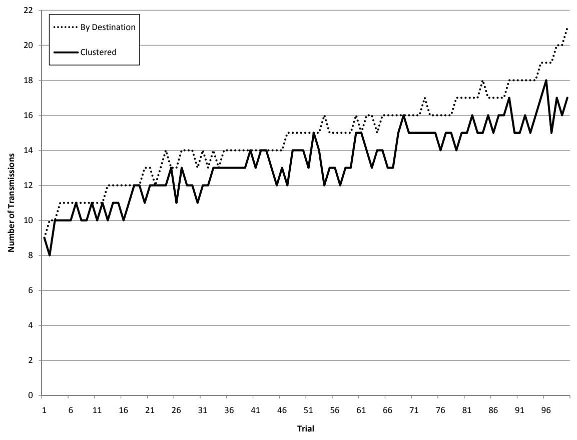


Figure 4. Scenario b

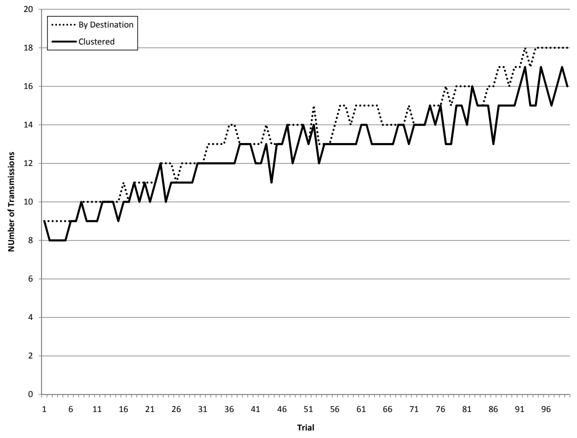


Figure 5. Scenario c

VI. CONCLUSIONS

This paper discussed dealing with channel induced errors in a TDMA-based MAC system. We described the cause and characteristics of errors in a contention-free wireless environment, introducing the bursty error phase as a cause of errors and reception failure.

We discussed a number of approaches to calculating the number of transmissions required to successfully transmit each frame. Through the combination of a flexible TDMA channel access approach and the application of the binomial distribution a significant reduction in the transmission time required was noted.

We highlighted that with a group of destinations the binomial distribution may not be optimal and exploited this by clustering together destinations with similar transmission probabilities in order to minimise the overall number of transmissions.

Our evaluation has validated that in the scenarios presented our clustering approach reduces the transmission time required to meet application reliability requirements,

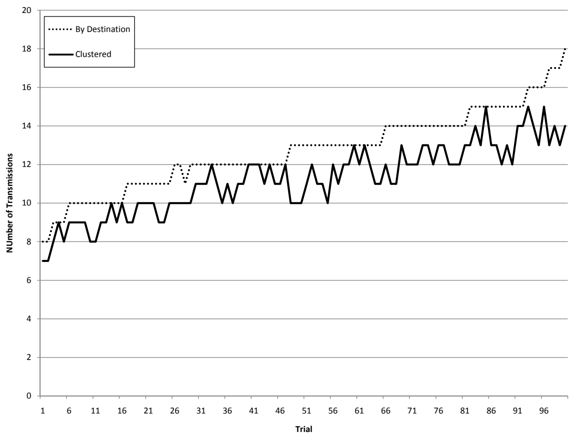


Figure 6. Scenario d

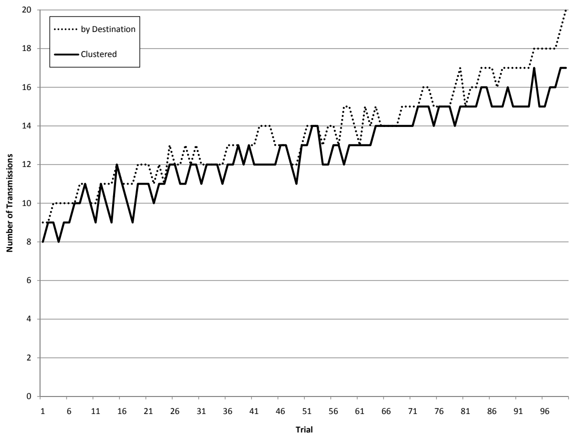


Figure 7. Scenario e

thus enabling a more efficient use of the limited resources available.

A. Future Work

Our future work will focus on the further optimisation of the clustering approach, through the variables considered when determining which destinations to cluster. Given binomial characteristics the variance between transmission success probabilities may be a more efficient variable.

We intend to incorporate the clustering approach as part of the admissions control within HD-TDMA [6] to provide a real-time wireless communication system.

References

[1] M. Caccamo, L. Y. Zhang, L. Sha, and G. Buttazzo, "An Implicit Prioritized Access Protocol for Wireless Sensor Networks," in *Proceedings of the 23rd IEEE Real-Time Systems Symposium. (RTSS 2002)*, TX, USA, Dec 2002, pp. 39–48.

[2] R. Cunningham and V. Cahill, "Time Bounded Medium Access Control for Ad Hoc Networks," in *Proceedings of the 2nd ACM International Workshop on Principles of Mobile Computing*, Toulouse, France, Oct 2002, pp. 1–8.

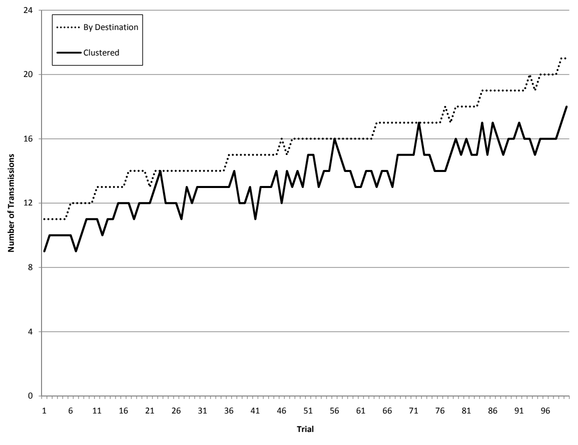


Figure 8. Scenario f

[3] M. M. Sobral and L. B. Becker, "A wireless hybrid contention/TDMA-based MAC for real-time mobile application," in *Proceedings of the 2008 ACM Symposium on Applied Computing (SAC '08)*, Cear, Brazil, Mar 2008, pp. 284–288.

[4] K. Otani, K. Daikoku, and H. Omori, "Burst error performance encountered in digital land mobile radio channel," *IEEE Transactions on Vehicular Technology*, vol. 30, no. 4, pp. 156–160, Nov 1981.

[5] A. Willig, M. Kubisch, C. Hoene, and A. Wolisz, "Measurements of a wireless link in an industrial environment using an IEEE 802.11-compliant physical layer," *IEEE Transactions on Industrial Electronics*, vol. 49, no. 6, pp. 1265–1282, Dec 2002.

[6] M. Gleeson and S. Weber, "Fault Recovery and Redundancy in Real-time Wireless TDMA," University of Dublin, Trinity College, Tech. Rep. TCD-CS-2008-24, May 2008.

[7] B. Huber and W. Elmenreich, "Wireless time-triggered real-time communication," in *Proceedings of the 2nd Workshop on Intelligent Solutions in Embedded Systems*, Graz, Austria, Jun 2004, pp. 169–182.

[8] P. Bhagwat, P. Bhattacharya, A. Krishna, and S. K. Tripathi, "Using channel state dependent packet scheduling to improve TCP throughput over wireless LANs," *Wireless Networks*, vol. 3, no. 1, pp. 91–102, Mar 1997.

[9] B. Fritchman, "A binary channel characterization using partitioned Markov chains," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 221–227, Apr 1967.

[10] D. D. Demarch and L. B. Becker, "An Integrated Scheduling and Retransmission Proposal for Firm Real-Time Traffic in IEEE 802.11e," in *Proceedings of the 19th Euromirco Conference on Real-Time Systems (ECRTS 2007)*, Pisa, Italy, Jul 2007, pp. 146–158.

[11] J. G. Skellam, "A probability distribution derived from the binomial distribution by regarding the probability of success as variable between the sets of trials," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 10, no. 2, pp. 257–261, 1948.

A Proposal for a Notion of Timeliness in Wireless Sensor Networks

Ramon Serna Oliver and Gerhard Fohler

Chair of Real-Time Systems
Technische Universtät Kaiserslautern, Germany
{serna_oliver, fohler}@eit.uni-kl.de

Abstract

The intrinsic properties of Wireless Sensor Networks (WSN) such as their ad-hoc infrastructure, energy constraints, and limited availability of resources, constitute an unfavorable environment for end-to-end timeliness guarantees. Many existing solutions are based on a timeliness notion borrowed from real-time systems, which can only express strict end-to-end deadlines for individual messages. However, it is practically unfeasible to impose these timeliness requirements in WSN without overestimating the network capacity.
In this paper, we present a generalized notion of timeliness suitable for the unpredictable environments of WSN. This notion allows to express a target time interval and the level of confidence of a sequence of messages arriving within the interval. The generalized notion provides means to capture the probability of the end-to-end transmission delays of a sequence of messages within this interval. This notion fits the general requirements of time-sensitive applications while at the same time allows to cope with the unpredictability of real WSN.

1. Introduction

The inherent properties of Wireless Sensor Networks [1] (WSN) constitute an unfavorable environment for timeliness guarantees [2]: ad-hoc infrastructure, strict energy constraints, and limited availability of resources, combined with the exposure to uncontrolled environments (e.g. nature) as well as external interferences (e.g. RF noise) increases the uncertainty of successful transmissions.
The communication mechanism of WSN is based on hop-to-hop message forwarding schemes in which intermediate hops direct messages to one, or several neighbors until the destination is reached. However, the accomplishment of this task is jeopardized by additional aspects such as mobility and lack of global network coordinators.
There is a growing interest in overcoming these restrictions to effectively provide end-to-end timeliness guarantees in WSN [3]. Unfortunately, it is practically unfeasible to determine strict end-to-end delivery delays without overestimating the network capacity.
One fundamental problem is that the adopted notion of timeliness is directly borrowed from classic real-time literature [4]. Hence, the problem to be solved is reduced to providing deadlines to each individual message and a set of additional mechanisms which try to enforce them.
However, the uncertainties of wireless networks, and particularly of WSN, are such that individual messages are always subject to unbounded transmission delays [5]. Satisfying individual deadlines is not feasible a priori, unless additional presumptions about the network are taken [6].
Many of the existing methods introduce implicit assumptions on the underlying models which are required to ensure feasibility. These assumptions are typically related to static and regular topologies [7], symmetry of the radio propagation patterns [8] or absence of environmental interferences. However, by doing so these methods restrict their applicability to specific scenarios which may not be representative of real deployments.
In this paper, we propose a generalized timeliness notion which provides enough flexibility to suit the characteristics of WSN without restrictive assumptions. Instead of aiming at strict deadlines for individual messages, the generalized notion focuses on the timeliness capacity of a sequence of messages. The notion allows to express the end-to-end timeliness requirements by means of a target time interval and a confidence level. Hence, it is possible to relax the requirements imposed by methods based on strict deadlines while still providing valid means to evaluate timeliness performance.
The generalized notion of timeliness is more suitable to the principles of WSN. Unlike the classic notion from real-time, it allows to capture the timeliness performance of a sequence of messages rather than individuals which diminishes the effects of unbounded end-to-end delay transmissions. Note that it is a generalization of the classic notion of timeliness as it also allows to express the same level of strictness.

This work is partially financed by the European Commission under the Framework 6 IST Project "Wirelessly Accessible Sensor Populations (WASP)".

The rest of the paper is organized as follows: Section 2 explores the related work in this field. Section 3 introduces the generalized notion of timeliness with more detail, followed by Section 4 which provides an example to illustrate its applicability. Finally, Section 5 concludes the paper.

2. Related work

Ongoing research to introduce real-time guarantees in WSN is carried out at many different levels. In [3] a survey of the current state-of-the-art is presented. Additionally, an overview of the problems in combined soft and hard real-time solutions covering the whole network stack as well as open challenges are discussed.
At the routing level, work in [9] and [10] assign velocities to messages which must be kept in order to fulfill their timeliness requirements. However, both assume static networks and nodes equipped with localization capabilities. In [11], delay guarantees are provided by means of a TDMA scheme at the expense of limiting the length of routing paths.
Traffic regulation mechanisms are also explored as means to provide end-to-end guarantees using queuing models. In [12], the combination of queuing models and message scheduler, turns into a traffic regulation mechanism that drops messages when they loose their expectations to meet predefined end-to-end deadlines. Additionally, an example is given to approximate the delay distribution of each hop in the event of instability by means of a Gaussian distribution.
Other probabilistic methods to achieve QoS have been approached by different authors. For CPU scheduling, the notion of probabilistic deadlines and execution time distribution is explored in [13]. In [14], different levels of quality of service are considered with respect to timeliness and reliability providing probabilistic multi-path forwarding to ensure end-to-end delays. Note that despite these methods apply probabilistic techniques to their algorithms, they all aim at satisfying strict deadlines for individual messages.
In [15], the authors introduce an analysis of the impact of mobility in achieving timeliness guarantees. Additionally, a prioritized event transmission protocol based on a proactive routing protocol and resource reservation is foreseen, although the authors take the assumption of a predictable medium access protocol.
A common notion of timeliness, based on the assignment of strict end-to-end deadlines to each individual message is applied in the work referred. Not surprisingly, they all present a number of assumptions with respect to the network which restring their deployment.

With respect to the MAC level, much of the existing research is based on TDMA scheduling of neighbor nodes (e.g. [16]), hence constructing a schedule of transmissions with contention free periods. However, although valid results are obtained in controlled environments, the common restriction of these methods is the assumption of error-free communications. Moreover, the complexity of such strategies, specially in mobile networks, forces the addition of global network coordinators, which discourages their use. Alternative approaches exist, such as [17] which achieves hard real-time guarantees given an hexagonal topology of static nodes. This requirement is later relaxed in [18] although it still relies on static nodes. Besides, both methods are built on the assumptions of bounded network density and optimum communication conditions.
Analytical solutions have also been studied. In this direction, [19] approaches a sufficient schedulability condition to guarantee end-to-end delays in multi-hop WSN. Nevertheless, it is based on specific assumptions on the message transmission times and channel transmission speeds, as well as network density and path lengths. Moreover, it is practically unfeasible to produce analytical models capable to capture the dynamics of a real WSN. Assumptions, again, are necessary in order to adjust reality to the models.

3. Notions of timeliness

The concept of timeliness currently exploited in WSN is greatly influenced by the one originated in real-time networks. In particular, attention is centered around temporal guarantees of individual messages by means of fulfilling deadlines. Each message receives an end-to-end deadline which delimits the time to reach the destination. If the message has not been delivered after this instant, it is likely to be dropped at one of the intermediate hops, depending on the routing policy. Certain routing strategies will drop messages before the expiration of the deadline if they estimate that the deadline cannot be met.

3.1. Meaningful notion of timeliness

We explore a different approach to achieve a better alignment between the network capabilities and the desired timeliness requirements. Instead of constraining the methods to fulfill idealized timeliness properties, we propose to relax the concept of timeliness, to suit the particularities of WSN. We considered the following requirements:

1) The way in which timeliness requirements are expressed should not encourage applications to demand unfeasible degrees of performance that the network cannot provide. Hence, given the unfeasibility of WSN to guarantee single deadlines, applications should express their demands at a higher level than individual messages.
2) A notion of timeliness expressing only success or failure, i.e., deadline met or not, is of only limited value to WSN. Rather, a continuous function to embody the

level of conformance with respect to the timeliness performance is more suitable to the properties of WSN.

3) The capability of WSN to enforce strict end-to-end timeliness requirements is reduced and variable at run-time. Hence, a meaningful notion of timeliness should allow applications to express a level of confidence for the aimed timeliness performance.

The generalization of the notion of timeliness that we propose supports these requirements and is composed of the following parts:

1) Our notion expresses timeliness properties of a sequence of messages, which makes it possible to cope with the undeterminism individual delivery delays in WSN and still provide meaningful values. Note that a sequence of message can be any series of messages as long as they follow the same route inside the network.
2) A time interval (t_i, t_j) with $t_j > t_i \geq 0$, which sets the acceptable end-to-end delay bounds for a sequence of messages.
3) The level of confidence for the required end-to-end interval, expressed by means of a probability $0 < p < 1$ of successful arrivals within the interval.
4) The end-to-end delay distribution function, used as a timeliness indicator, which allows to capture the probability density of the sequence of messages arriving within the interval. The function, which can be obtained at run-time, provides sufficient information to determine the probability of sequences of messages arriving within the specified interval.
5) The selection of the probability level and the length of the interval allows the specification of strict timeliness yet providing additional levels of flexibility which suits the particularities of WSN. Thus, our notion is a generalization of the classic timeliness notion.

By considering a sequence instead of individual messages, it is possible to work around the indeterminism of WSN and still provide meaningful values. Furthermore, the selection of the probability level and the length of the interval allows the specification of strict timeliness yet providing additional levels of flexibility which adapt to the peculiarities of WSN. Moreover, this notion is adequate to evaluate the end-to-end timeliness performance as well as to express requirements in a way that does not demand excessive levels of precision that the network cannot achieve.

Figure 1 shows the probability density function (PDF) obtained by simulation of a routing path of length 5 hops as depicted in Figure 2. Each intermediate hop of this path had two neighbors and each hop on the network (including those forming the path) generated traffic with a time between messages following an exponential distribution with parameter $\lambda = 15s$.

The artificial load was set to simulate the effects of cross-

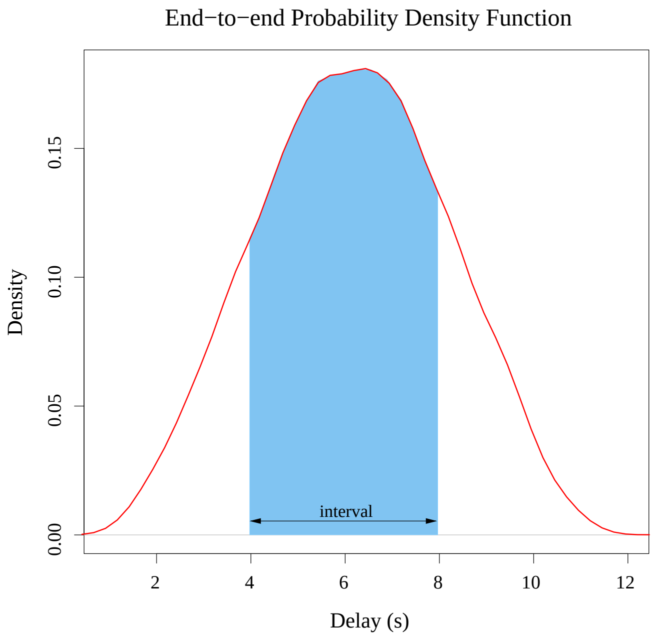


Figure 1. Expressing timeliness by means of the end-to-end distribution (PDF)

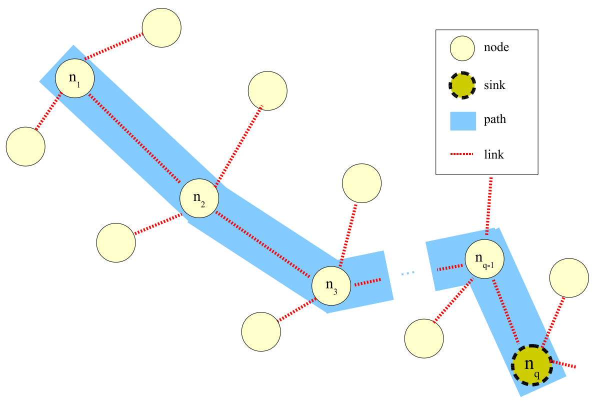


Figure 2. Simulation scenario

traffic on a segment of a big network. Additional messages were generated periodically every $30s$ at the source of the path and its end-to-end delay captured at the sink.

This scenario was simulated by means of the network simulator Omnet++ [20] [21] and Mobility Framework [22]. The routing path was manually fixed for this experiment and all messages on the network were directed to the sink. The chosen MAC protocol was Wisemac [23].

In this example, the timeliness requirements correspond to the interval $(4s, 8s)$. Hence, the area bellow the pdf curve represents the probability of end-to-end delays to fall within the interval. At run-time, it is possible to analyze the percentage of messages from a sequence which fulfill this timeliness requirement.

Figure 3 depicts the estimated cumulative distribution func-

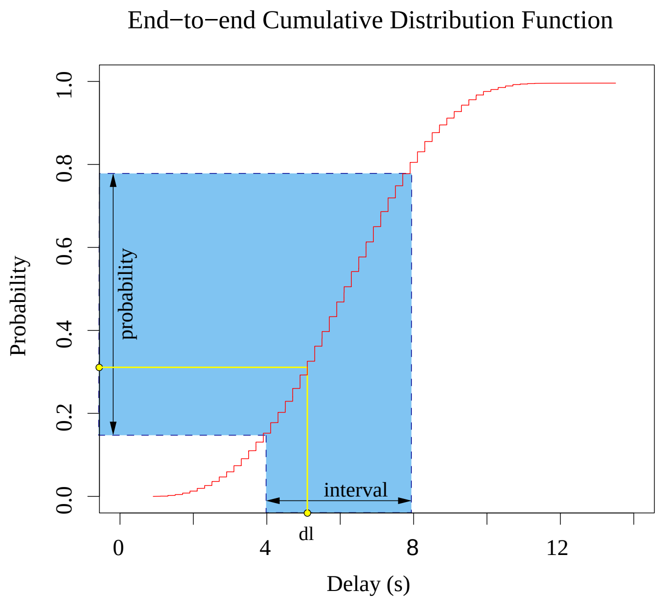


Figure 3. Expressing timeliness by means of the end-to-end distribution (CDF)

tion of the same experiment with an additional line illustrating the classic strict timeliness notion (dl). The probability of fulfilling the timeliness requirements is highlighted and represents approximately 60%.
Both figures, illustrate the relation between the bounds of the time interval and the achieved probability. Notice that both factors are directly dependent of each other.

3.2. Run-time considerations

Existing protocols and methods can be adapted to capture the end-to-end delay distribution used in the timeliness notion presented in this paper. However, there are a number of considerations to take into account to obtain satisfactory results.

- The end-to-end distribution of message delivery delays is not constant over time, thus estimations on the end-to-end delay distribution must continuosly adjusted. Methods to dynamically estimate end-to-end distributions at run-time adapting to dynamic changes of the network must be explored.
- Sensor nodes are equipped with limited resources. Thus, computations must be of low complexity and limited memory usage.
- Sharing global knowledge is expensive in terms of broadcasting messages. The estimation of end-to-end distribution should not generate additional traffic on the network and make use of local information to perform the calculations.

4. Applicability examples

To illustrate the applicability of the proposed notion of timeliness, we derive a sample scenario from a general Elderly Care use-case [24]. The goal of the application is providing automatized monitoring of elderly people in a care house. Sensor nodes are attached to the patients to monitor their general health and well being, with special interest in aspects related to mobility and temperature.

4.1. Scenario

The description of the scenario is as follows:

- Approximately 20 patients living in the same home (20 rooms in four floors).
- Two sensor nodes per patient: one equipped with a 3D accelerometer and a one with a temperature sensor.
- Two operation modes: normal and special.

The data transmission rate depends on the operational mode. In normal mode, nodes process their data after acquisition and transmit messages at a low frequency with respect to the sampling rate (e.g. average temperature), whereas in the special mode data is sent without processing for every sample.
Lets look at a patient recovering from a major fall. In this case, the information from the accelerometer is constantly transmitted to monitor the patient's mobility. The locomotion analysis processes the accelerometer data at a frequency ranging between 20 and 50Hz. The sensor nodes of other patients, which do not require such special monitoring, process their data by means of a local algorithm which generates an output at a much lower frequency (e.g. 1Hz).
Several time constraints appear due to the nature of the measurements. In the normal operation mode, some latency is well tolerated by the system. However, the special mode requires the data to be delivered within a few seconds to allow detailed locomotion monitoring.

4.2. Classic timeliness notion

The classic notion sets end-to-end deadline to each transmitted message. Lets assume that these are set to 15 seconds for normal mode and 2 seconds for the special.
Note that these values are chosen off-line, hence without any concrete knowledge of the network status. Therefore, either if a patient is close to the base station with direct connectivity or it is several hops distant, these deadlines are to be met. Furthermore, the timeliness requirements are expressed in a strict manner, without taking into account whether it is one single patient which requires special monitoring or many of them. However, the bandwidth availability and response time could greatly differ in both situations.
Most existing routing protocols would try to enforce the

fulfillment of deadlines. However, it is expected that at a given time, either because the patient moves away from the base station, or due to the additional traffic generated by other nodes, some deadlines will be missed. In such a case, the common procedure is to drop messages without expectations to achieve the destination and save some bandwidth for other messages that still can make it. Alternatively, the message might be transmitted in spite of missing its deadline. However, in both cases the transmission will be accounted as a failure.

4.3. Generalized timeliness notion

Following the notion presented in this paper, the procedure changes the perception of timeliness requirements both at the application as well as at the network level. In a first instance, the application does not express strict end-to-end deadlines for individual messages but rather acceptable intervals for sequences of messages. In the example, this can be expressed in the way of requesting messages in the special mode to be delivered within the interval $(1s, 3s)$ with a probability of 80%. This way, a desired end-to-end delay distribution is expressed.
The difference with respect to the classic notion is that the effects of the unpredictability of WSN are taken into account. Hence, individual messages missing their deadlines are accepted as long as the end-to-end distribution satisfies the constraint. Only when the distribution of the sequence of message exceeds the expected distribution actions are to be taken.
The generalized notion of timeliness provides hooks to apply mechanisms enhancing the network behavior. For instance, by exploring trade-offs to increase the timeliness performance of intermediate hops at the expenses of higher energy consumption (e.g. discover new routes, increase duty cycle of intermediate hops, re-adjust the radio transmission power, etc). If the network stack (e.g. routing protocol) determines that the requested timeliness performance is not feasible, it is of no sense to continue accepting messages from the application layer without the proper adaptations to the current network status. Hence, the adequate feedback channel must be established with the application. For instance, the network stack may inform the application that only 60% of the messages arrive within the required time interval $(1s, 3s)$ while 80% of them do it within $(2s, 4.5s)$. The decision whether to relax the interval bounds or accept the lower probability is left to the application.
This procedure contradicts the classic notion in which the requirements are expressed in an unilateral way, and the network stack has no option but to deal with them. Possible actions are to adjust the sampling rate, perform local processing on the data or accepting that the sequence of messages will be delivered with a worse timeliness performance than it was originally desired.

5. Conclusion

In this paper we proposed a new notion of timeliness suitable to cope with the inherent properties of WSN. We argue that the classic approach is not appropriate as it forces existing solutions to introduce restrictive assumptions in order to achieve otherwise unfeasible performance levels.
The generalized notion, permits a better expressiveness of time requirements and reflects more precisely the timeliness performance of a WSN. Moreover, it facilitates the exploitation of trade-offs and introduces enough flexibility to cope with the uncertainties of WSN.
Further work in this area is currently carried out to develop new protocols based on this concept, as well as to exploit possible timeliness trade-offs. A routing protocol to exploit at run-time the timeliness notion presented in this paper is under development.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “Wireless Sensor Networks: A survey,” *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] J. Zhao and R. Govindan, “Understanding packet delivery performance in dense wireless sensor networks,” in *SenSys ’03: Proceedings of the 1st international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2003, pp. 1–13.

[3] Y. Li, C. S. Chen, Y.-Q. Song, and Z. Wang, “Real-time QoS support in wireless sensor networks: a survey,” in *7th IFAC International Conference on Fieldbuses & Networks in Industrial & Embedded Systems - FeT’2007*, 2007.

[4] G. C. Buttazzo, *Hard Real-Time Systems: Predictable Scheduling Algorithms and Applications*, second edition ed., Springer, Ed., 2005.

[5] J.-D. Decotignie, “Keynote: Real-time and wireless sensor networks: Why do we need another view at it?” in *7th International Workshop on RealTime Networks (RTN) (abstract)*, July 2008, pp. 6–7.

[6] R. Serna Oliver and G. Fohler, “Probabilistic routing for wireless sensor networks,” in *Work-in-Progress Session, 29th IEEE Real-Time Systems Symposium*, December 2008.

[7] K. Shashi Prabh and T. Abdelzaher, “On scheduling and real-time capacity of hexagonal wireless sensor networks,” July 2007, pp. 136–145.

[8] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, “Impact of radio irregularity on wireless sensor networks,” in *ACM Conference on Mobile Systems, Applications, and Services (Mobisys)*, June 2004, pp. 125–138.

[9] O. Chipara, Z. He, G. Xing, Q. Chen, X. Wang, C. Lu, J. Stankovic, and T. Abdelzaher, “Real-time power-aware routing in sensor networks,” in *14th IEEE International Workshop on Quality of Service (IWQoS).*, June 2006.

[10] T. He, J. Stankovic, C. Lu, and T. Abdelzaher, "SPEED: A stateless protocol for real-time communication in sensor networks," in *Proceedings of ICDCS'03*, May 2003.

[11] A. Sahoo and P. Baronia, "An energy efficient MAC in wireless sensor networks to provide delay guarantee," in *15th IEEE Workshop on Local and Metropolitan Area Networks. LANMAN*, June 2007.

[12] K. Karenos and V. Kalogeraki, "Real-time traffic management in sensor networks," in *Proceedings of the 27th IEEE Real-Time Systems Symposium (RTSS)*, 2006.

[13] L. Abeni and G. Buttazzo, "Qos guarantee using probabilistic deadlines," 1999, pp. 242–249.

[14] E. Felemban, C.-G. Lee, E. Ekici, R. Boder, and S. Vural, "Probabilistic QoS guarantee in reliability and timeliness domains in wireless sensor networks," vol. 4, March 2005, pp. 2646–2657.

[15] B. Hughes, , B. Hughes, and V. Cahill, "Achieving real-time guarantees in mobile ad hoc wireless networks," in *Proc. Work-in-Progress Session 24th IEEE Real-Time Systems Symp*, 2004.

[16] S. Gobriel, R. Cleric, and D. Mosse, "Adaptations of TDMA scheduling for wireless sensor networks," in *7th International Workshop on Real-Time Networks (RTN)*, 2008.

[17] M. Caccamo, L. Zhang, L. Sha, and G. Buttazzo, "An implicit prioritized access protocol for wireless sensor networks," in *23rd IEEE Real-Time Systems Symposium (RTSS)*, 2002.

[18] T. Watteyne and I. Auge-Blum, "Proposition of a hard real-time MAC protocol for wireless sensor networks," in *13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. IEEE Computer Society, 2005.

[19] T. Abdelzaher, S. Prabh, and R. Kiran, "On real-time capacity limits of multihop wireless sensor networks," in *Proceedings of the IEEE International Real-Time Symposium (RTSS)*, 2004.

[20] A. Varga, "The omnet++ discrete event simulation system," in *Proceedings of the European Simulation Multiconference*, Prague, Czech Republic, June 2001, pp. 319–324.

[21] "OMNeT++ discrete event simulation environment," http://www.omnetpp.org.

[22] A. Koepke, "Mobility Framework: A framework to support simulations of wireless and mobile networks within OMNeT++," url: http://mobility-fw.sourceforge.net/.

[23] A. El-Hoiydi and J.-D. Decotignie, "WiseMAC: An ultra low power MAC protocol for the downlink of infrastructure wireless sensor networks," in *Ninth IEEE Symposium on Computers and Communication (ISCC)*, June 2004.

[24] C. Lokhorst, R. de Mol, H. Wells, M. Mazzu, F. Frumento, A. Corongiu, W. Savio, A. Ipema, A. Sorniotti, and L. Gomez, "Deliverable d6.2: Specified application requirements and prototype selection," Tech. Rep., September 2007.

A Vision of Cyber-Physical Internet

Anis Koubâa*‡ and Björn Andersson*

*CISTER Research Unit, Polytechnic Institute of Porto (ISEP/IPP), Portugal.
‡Al-Imam Mohamed bin Saud University, College of Computer and Information Sciences, Riyadh, Saudi Arabia.

Emails: aska@isep.ipp.pt, bandersson@dei.isep.ipp.pt.

***Abstract*—When the Internet was born, the purpose was to interconnect computers to share digital data at large-scale. On the other hand, when embedded systems were born, the objective was to control system components under real-time constraints through sensing devices, typically at small to medium scales. With the great evolution of the Information and Communication Technology (ICT), the tendency is to enable ubiquitous and pervasive computing to control everything (physical processes and physical objects) anytime and at a large-scale. This new vision gave recently rise to the paradigm of Cyber-Physical Systems (CPS). In this position paper, we provide a realistic vision to the concept of the Cyber-Physical Internet (CPI), discuss its design requirements and present the limitations of the current networking abstractions to fulfill these requirements. We also debate whether it is more productive to adopt a system integration approach or a radical design approach for building large-scale CPS. Finally, we present a sample of real-time challenges that must be considered in the design of the Cyber-Physical Internet.**

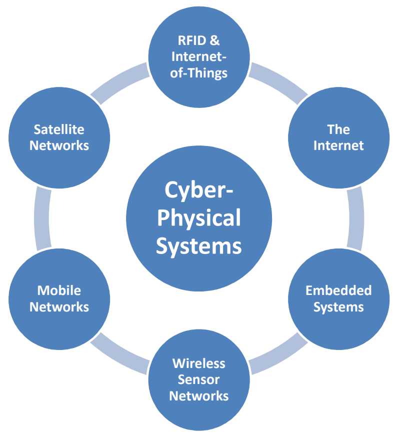


Fig. 1. Large-Scale Cyber-Physical Systems Components

I. Introduction

The vision towards *large-scale* distributed computing systems is currently evolving to a new frontier, where computation is no longer decoupled from its environment. This sight stems from the need to integrate external physical data and processes with computations for sake of pervasive and ubiquitous control of the surrounding environment. However, it is commonly known that this integration is not a new concept as it has always been the case with embedded systems. In fact, embedded computing systems are intrinsically dependent on their environment where they are deployed through sensing physical processes. As computing becomes increasingly integrated into our environment, traditional embedded systems has found their limits in satisfying the new requirements of massively networked embedded systems. On the other hand, the Internet has been providing a worldwide infrastructure for data sharing and information retrieval. However, Internet applications have been driven by the need to exchange logical information at large-scale; nevertheless, the mapping between the physical environment and the logical information has not been considered in the design of those applications. Thus, the convergence of the Internet with embedded systems is an important milestone for enabling large-scale distributed computing systems that are tightly coupled with their physical environment. On the one hand, a first step towards this convergence has been put into practice by Radio-Frequency Identification (RFID) based systems, which have enabled the concept of network of physical objects, commonly known as Internet-of-Things [1]. In November 2005, the International Telecommunication Union (ITU) produced an executive report in the World Summit on the Information Society that summarizes the basic concepts of Internet of Things, related technologies, challenges and concerns, market opportunities, etc. Clearly, RFID has been considered as the key technology for Internet-of-Things.

On the other hand, the Wireless Sensor Networks (WSNs) paradigm has emerged as another alternative to networks of physical events, which supports the control and monitoring of physical phenomena in the environment through sensing. Some other alternatives considered the use of sensor-based mobile phones for monitoring everyday items through cellular networks [2]. All of these approaches fall into the concept of Cyber-Physical Systems (CPS), which are systems deployed in large geographical areas and generally consist of a massive number of distributed computing devices tightly coupled with their physical environment. Fig. 1 presents the main components of cyber-physical systems.

The frontier between CPS and Internet-of-Things has not been clearly identified since both concepts have been driven in parallel from two independent communities (i.e. sensor networks and RFID, respectively), although they have always been closely related. The history returns itself as this situation may be thought to be similar to the design of the Internet (driven by the TCP/IP community) and Telecommunication networks (driven by International Telecommunication Union - ITU) in the early eighties. However, with the emergence of the recently released 6LoWPAN [3], the convergence between CPS and Internet-of-Things becomes a real fact as it enables to use the Internet as supportive infrastructure to sensor networks, similarly to its integration with RFID systems.

The current status of these new emerging cyber-physical systems recall to the mind the age just preceding the birth of the Internet, when networks were scattered and private mainly due to lack of standards. Similarly, CPS are currently scattered and private networks, each performs specific tasks related to the environment where it operates. The main challenge in the design of CPS is how to enable the interconnection and interoperability of all these scattered networked embedded systems into a single large-scale network that satisfies all their requirements. There are a number of handicaps that hinder the set-up of a unified network for cyber-physical systems. In addition, one important question is whether it would be better that the design of CPS follows a system integration approach, which consists in integrating heterogeneous networks together to form a CPS, or a radical design approach (as it was claimed in several papers [4]–[6]), which consists in building CPS from scratch.

In this position paper, we first present and criticize the recent vision towards the design of CPS radically from scratch and we show that realism imposes not to lose legacy. We show that the real current trends address the challenge of interoperability between existing heterogeneous systems to form a universal network interconnecting not only data but also objects and physical events. Finally, we present the networking abstraction and challenges for the design of the Cyber-Physical Internet (CPI).

II. The Design Dilemma

As of the emergence of CPS, there were many calls to rethink the computation foundations to cross a new frontier towards future cyber-physical networks [4]–[6]. This is a pretty nice statement to trigger new theoretical research challenges; however, in the practical sense, it may face serious limitations. Although it is obvious that computation paradigms must be adapted to the new requirements that arise with the emergence of the cutting-edge cyber-physical technologies, adopting a radical design approach, as it might be understood from the literature [5], [6], seems to be not pragmatic, at least in medium to short terms. During the modeling process of large-scale complex systems, it is always important that designed models ensure the best trade-off between their different requirements; however, when taking a look to history we can realize that real systems that spread out into the market does not really fulfill the objectives of the theoretically expected models. A straightforward question is: "*Will Cyber-Physical Systems face the same fate?*"

As a matter of fact, two worldwide standard technologies perfectly embed this belief: IP (Internet Protocol) and IEEE 802.11 (hereafter, WiFi). These two standard protocols are commonly known to be rather poor in terms of efficiency and Quality-of-Service (QoS). Several patches have been proposed for IP (such as Integrated Services, Differentiated Services, etc.) as well as for WiFi (e.g. IEEE 802.11e extension) to enhance their performance. In spite of efficiency and QoS shortage, these two protocols have been widely and quickly spreading since their release. On the other hand, other more sophisticated protocols such X.25, ATM or HyperLan have been designed with more care to achieve higher efficiency and better QoS, but did not gain too much space in the commercial market. It appears that there is always a gap between *how new systems are expected to operate* and *how they do operate in reality*. The reason is that the vision of the market stakeholders is different from the vision of academic researchers, as the former do not care about optimized efficiency, but rather reduce the time-to-market and cost of *real* products. Hence, it can be easily noticed that, in practical terms, the modeling paradigm is to quickly design, implement and put-into-market simple solutions that (1) just work, (2) fulfill basic requirements and (3) can be patched to plug new functionalities or to improve their behaviors. Therefore, it seems that rethinking the current computation foundations to build large-scale CPS is not pragmatic. Instead, it seems that it is more natural to take profit from the legacy infrastructure to achieve the large-scale CPS objectives. We thus believe that research efforts must focus on the system integration approach for enabling very large-scale CPS, which we refer to as the *Cyber-Physical Internet*. Consequently, interoperability is the key challenge to build large-scale heterogenous cyber-physical networks. In addition, it is necessary to take into consideration the specificities of CPS in the integration process of existing networks, in particular the nature of the manipulated data, as discussed in the next section. One question may thus arise: "*What will be the core protocol for the prospective CPI*?". Definitely, IP is the legacy protocol that will play the key role in the future Cyber-Physical Internet. IP has been thought for so long — since the birth of the sensor network paradigm — as being non compatible with the requirements of sensor-based systems. However, this thought has been recently revisited [7]–[9] and with the emergence of 6LoWPAN [3], [9] that embeds IPv6 on top of the IEEE 802.15.4 [10] as an alternative to ZigBee [11] Network Layer. The challenge has been won by the IETF 6LoWPAN Working Group and 6LoWPAN becomes a serious competitor to ZigBee as it enables to seamlessly merge the sensor network world with the Internet, which ZigBee is not able to. However, if 6LoWPAN wins in terms of high degree of interoperability with existing networks, it still needs to justify its efficiency in terms of energy-efficiency and real-time guarantees. In fact, the design objectives of 6LoWPAN are likely to put more weights on interoperability and integration with the Internet rather than on the typical requirements of sensor networks. Optimizing the trade-off between those design objectives remains a research challenge.

III. Networking Abstractions of the Cyber-Physical Internet

In this section, we define the requirements that have to considered in the specification of the networking abstractions of the Cyber-Physical Internet, which can roughly be viewed as the large-scale universal network that interconnects several heterogeneous CPS. We consider a CPS as a mixture of several and different networks that monitor physical objects and events, including WSNs, RFID-based systems, mobile phones, etc.

Future massively networked embedded systems require new standards for achieving interoperability. In fact, prospective large-scale CPS should not be foreseen as separated and dispersed systems, but as a unified system that seamlessly interconnects heterogeneous cyber-physical components. The challenge will be to build a global network that interconnects all cyber-physical devices and provide plug-and-play services to the end-users in a completely transparent way. It is therefore necessary to rethink the current networking abstractions to ensure a worldwide interoperability of cyber-physical devices. This requirement imposes the design and the development of new standardized protocols for cyber-physical systems. These protocols have to be designed while taking into account the properties of the environment, where the CPS will be deployed. The IP protocol stack model and the WSN protocol stack model [12] feature fundamental limitations that must be addressed in the design of the Cyber-Physical Internet. In fact, the current protocol layers make a total abstraction on the nature of data to be processed, thus it is not possible to design protocols tightly coupled with their external environment. It is therefore necessary to propose an extended protocol stack model for CPS that also integrates the properties of the physical environments. Fig. 2 presents the potential reference architecture for the CPI.
The protocol stack architecture for CPI must include an additional layer, the Cyber-Physical Layer (CY-PHY layer), which provides an abstract description of the properties and nature of cyber-physical data. This layer must provide the set of protocols to universally represent data in a unified and structured way. In addition, the CY-PHY layer should provide services for lower layer protocols to support an efficient cross-layer design of the underlying application and communication protocols. This means that all protocol layers have to adapt their behavior according to the information provided by the CY-PHY layer. For instance, in the context of health care monitoring, the information provided by the body sensors will have a significant impact on the behavior of the protocol suite designed for this application. In fact, depending on the type and the nature of cyber-physical data, several changes may be imposed in the protocol layers, namely:

Physical Layer: The input from the CY-PHY layer can lead to changing some properties of the physical channel such as the channel frequency band and the modulation scheme depending on the requirements of the cyber-physical data. It can be conceived that some frequency bands can be allocated for critical cyber-physical traffic using robust modulation schemes. Note that there have been great advances in radio technologies with the design of software radios and cognitive radios, which offer better flexibility to adapting radio properties to the user requirements.

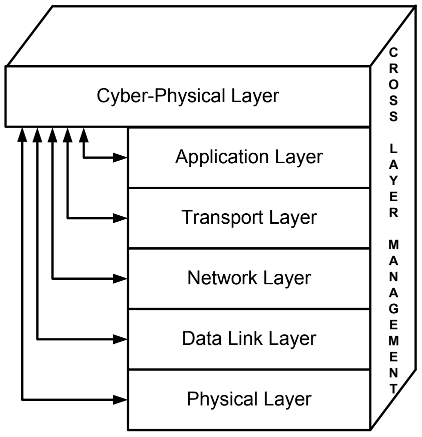


Fig. 2. Protocol Stack Architecture for CPI

Medium Access Layer: The MAC layer usually grants different operational modes and services for higher-layer protocols such as synchronized or unsynchronized modes, different levels of Quality-of-Service (QoS), energy management services (e.g. duty cycle) etc. In CPS, the dynamic behavior of the environment would significantly impact the operational behavior of the MAC layer, which must be adaptive to the CY-PHY data. For instance, the decision to switch from synchronized operational mode to unsynchronized mode (or vice-versa) or the adaptation of the duty cycle must be driven by the CY-PHY layer. This can be achieved by some existing technologies such as the IEEE 802.15.4 protocol, which offer the beacon-enabled mode (synchronized) and the non beacon-enabled mode (unsynchronized) in it MAC layer. This interaction between the MAC layer and CY-PHY layer is very important to enable a close-loop control of the QoS based integrated with the status of the monitored environment.

Network Layer: The Network Layer provides routing and data aggregation services. The cross-layer interaction between the Network Layer and the CY-PHY Layer is necessary to define the adequate routing strategies and data aggregation mechanisms. For example, the aggregation functions used for processing temperature information would be completely different from those used for accelerometer or bio-medical sensory data. In addition, the selection of the routing mechanism or the parameters affecting a given routing protocol may depend on the nature of the data and also from the status of the environment.

Transport Layer: Transport protocols have not been extensively investigated for CPS (e.g. wireless sensor networks, embedded systems), although it is of a paramount importance to specify different degrees of reliability with respect to the end-to-end delivery of data. This naturally implies three classical tasks in the transport layer including (i) reliable transport, (ii) flow control and (iii) congestion control mechanisms. The Internet already provides the connection-based TCP and connectionless UDP transport protocols for providing guaranteed and best-effort services, respectively. These heavyweight protocols are not suitable for CPS applications, which raise the need to rethink new transport protocols that cope with the requirements and properties of CPS. A real challenge with regards to transport protocols is to design reliable transport protocols without the need to send back acknowledgements to the source nodes to avoid drowning the network with increasing control traffic.

Application Layer: In CPS, the application layer is responsible of processing data and extracting useful information with respect to the application objectives. One main challenge is to provide *standard* distributed signal processing algorithms/protocols for each potential CPS applications. This will encourage the development of CPS applications and reduce the time-to-market and cost.

IV. Real-Time Challenges for CPS

Real-time usually imposes serious challenges in the design of cyber-physical systems. However, this issue must be carefully analyzed and some of the concepts must be revisited. There are several promising research directions in the real-time area. In what follows, a sample of potential directions are presented:

- *Operating systems*: It is important that operating systems supports real-time, although it induces additional design complexity. The main challenge is to achieve an optimal balance between several important features needed by a CPS operating system, including modularity, effective hardware/software split, hardware abstraction, energy efficiency, and real-time [6]. The most widely used operating system, TinyOS, represents a promising solution for CPS as it addresses most of those requirements. However, the lack of real-time support represents a serious limitation in TinyOS for developing real-time protocols and synchronization mechanisms [13]. The lack of pre-emption and prioritization in TinyOS is a main handicap for providing predictable timing behavior at node level. In many cyber-physical applications, where timing constraints must be respected, reliability and real-time are very much coupled. It is therefore fundamental to consider timing guarantees for building reliable systems. As a matter of fact, the lack of real-time in TinyOS has prevented the release of standard-conforming IEEE 802.15.4 protocol stack for both open-ZB [14] and TKN implementations [15], [16]. In [17], the authors demonstrated that the behavior of the IEEE 802.15.4 implementation has been much reliable when implemented over ERIKA [18], a promising real-time operating system for embedded devices.

- *Networking protocols*: From the networking perspective, real-time imposes several challenges still open to research. Distributed and adaptive resource allocation in synchronized multi-hop sensor networks, where resources must be adequately allocated depending of the physical/logical network changes, represents one interesting research problem. In synchronized WSNs, it is naturally more efficient to grant resources (bandwidth, memory) to active sensor nodes involved in critical tasks. The use of static allocation plans is clearly not efficient for highly dynamic and mobile systems as they do not adapt to the system changes. On the other hand, the centralized adaptive synchronization induces a significant amount of computation and communication overheads, which cannot really work in resources-constrained WSNs, due to its complexity and non-responsiveness. We need to find new approaches for adaptively managing resources for synchronized and mobile multi-hop WSNs in a *distributed*, efficient, transparent, and most importantly real-time way.

- *Timing Guarantees*: The provision of deterministic real-time guarantees in unpredictable wireless ad-hoc and sensor networks is considered as a questionable issue. In the literature, most of the papers dealing with deterministic guarantees assume that channels are error free. While this assumption might be correct for very extreme and rare cases, where wireless links are very stable and of a high quality, most of the real-world applications refute this assumption, since the majority of the wireless links are typically located in the *gray* region, where links are highly variable and unstable. The notion of real-time guarantees must be revisited, as we need to find new means to characterize deterministic performance under channel uncertainty. Claiming that a wireless network deterministically provides a delay bound would not make sense. One interesting characterization is to associate a confidence level with each guaranteed delay bound. The objective of the associated confidence level is to *quantify the uncertainty* on the guaranteed delay bound, as illustrated in Fig. 3. The main idea consists in computing the delay bounds taking into account the number of possible retransmissions, and to statistically determine the distribution of the number of retransmissions in a given channel. Fig. 3 shows the delay bounds and the cumulative distribution function (CDF) for different values of number of retransmissions due to channel errors. The CDF helps on bounding the maximum number of retransmissions with a certain probability, which defines the confidence level. In this case, it is possible to determine the delay bound that corresponds to the the maximum number of retransmissions, with an associated confidence. For instance, in Fig. 3 we can observe that the delay bound is equal to 0.35 time unit with a confidence level of 97%.

- *Performance Compositionality*: The end-to-end delay analysis in CPS is a complex and stimulating problem,

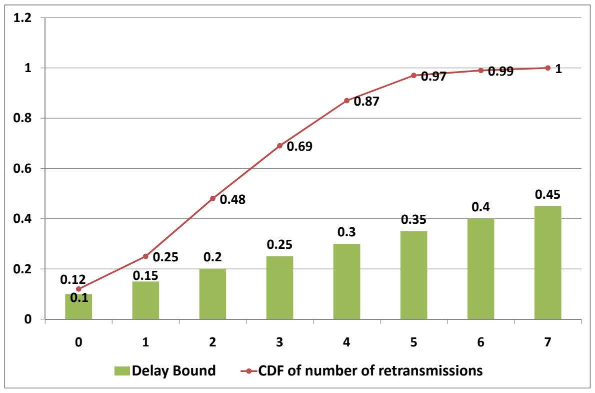


Fig. 3. Delay bound characterization under network uncertainty

in particular due to their heterogeneity. Traditional end-to-end delay analysis basically derives system-level delay from component-level delays, and this paradigm has been known to be inefficient. More sophisticated techniques use *composition theorems* to transform a multi-component system into a one-component system, thus reducing the complexity of the end-to-end analysis [19], [20]. In [19], [20] the authors presented a complete framework to systematically transform distributed real-time systems into a single system, which is used to infer the end-to-end schedulability of the original system. Although these works represent a pioneer contribution towards performance compositionality in distributed real-time systems, their traffic model — based on the schedulability analysis theory — is rather restricted to periodic/aperiodic streams and constant execution times (i.e. worst-case execution time), which is not enough generic for modeling heterogenous CPS applications. On the other hand, Network Calculus formalism, which relies on more generic traffic models (defined by their upper bound curves) [21], provides concatenation methodologies for reducing the analysis of a multi-hop system to a single-hop system by determining an equivalent service curve for the whole system. The concatenation analysis, however, has two drawbacks: (1) The equivalent service curve for a given stream depends on a parameter θ, whose optimization is quite complex [22], (2) The analysis relies on rate-latency service curves, which is not enough generic for modeling services in heterogenous CPS applications. For that reason, we need to find adequate system reduction techniques that rely on general model abstractions for representing traffic and services, and that take into account system heterogeneity.

- *Data aggregation*: As already stated in Section III, the current Internet protocol layers make a total abstraction on the nature of data to be processed. However, many CPS applications are not interested in the data itself but they are rather interested in high-level queries about the physical world. It is possible for a user to request that each sensor delivers its sensor reading and then computes the result based on all those sensor readings. Nevertheless, such an approach generates an enormous amount of data traffic something that (i) increases the time required to obtain the result of the query and (ii) wastes energy of sensor nodes. Performing information processing inside the network, for example allowing routers to also process incoming packets before forwarding, can lead to significant improvements however. This is often referred to as data aggregation, content-based network, in-network processing or data distillation network [23]. Regardless of its labeling, three important issues remain for the use of such an approach in Cyber-Physical Internet.

1) *Query language specification*. There is a need to define a language in which users can define their queries and these queries should be injected into the network. The community of wireless sensor networks is currently using slightly modified variants of SQL. However, these SQL variants are not sufficiently expressive. For example, in a scenario where we desire to detect whether a route is ice-free, we may wish that sensor nodes perform signal processing locally something that is difficult to perform efficiently with SQL. One approach could be however that a sensor allows users to install a device driver on that sensor node and this device driver acts as a virtual sensor; a virtual sensor performs a computation based on physical sensors. For example, a virtual sensor may deliver a Boolean value ”true” if there is ice close to this sensor and ”false” otherwise. Then, SQL may be used to express queries based on the virtual sensors. In fact, support for such virtual sensors are already available in a software package called Global Sensor Networks (GSN) [24] but it is used to be run on a gateway interfacing with a wireless sensor network rather than to be run on sensor nodes themselves.
2) *Query planning and optimization*. The research community of databases has produced an extensive literature on query processing of SQL queries. It typically assumes that the cost (for example time) of a query should be minimized and the query planning/optimization attempts to find a way of executing the query such that the cost is minimized. These works assume that the cost is dominated by disk accesses or CPU processing. However, for data aggregation in CPS, we expect the limited capacity of the (wireless) communication channel to be the main bottleneck and therefore query optimization should strive to minimize that cost instead. This is non-trivial because (i) knowledge about the (wire-

less) network topology and interference relationships between nodes are needed in order to exploit parallel transmissions and (ii) some operations can be performed at great efficiency (such as MIN, MAX) with a prioritized MAC protocol [25]; the query optimizer must be aware of that potential when taking decisions in how to decompose a user-query into operations.

3) *Data integrity*. When a user asks a query he wants to be sure that the sensor readings are authentic. Since data aggregation allows routing nodes to modify the data payload, normal end-to-end encryption/authentication methods do not work. Therefore, ensuring data integrity must be an integral part of the network.

V. Acknowledgements

This work was partially supported by the Portuguese Science and Technology Foundation (Fundacao para Ciencia e Tecnologia - FCT), the ArtistDesign Network of Excellence on Embedded Systems Design, funded by the European Commission under FP7 with contract number ICT-NoE- 214373 and CONET, the Cooperating Objects Network of Excellence, funded by the European Commission under FP7 with contract number FP7-2007-2-224053.

References

[1] I. T. Union, "Internet reports 2005: The internet of things," Tech. Rep., November 2005, executive Summary. [Online]. Available: http://www.itu.int/publ/S-POL-IR.IT-2005/e

[2] C. Frank, P. Bolliger, F. Mattern, and W. Kellerer, "The sensor internet at work: Locating everyday items using mobile phones," *Pervasive and Mobile Computing*, vol. 4, no. 3, pp. 421–447, Jun. 2008.

[3] G. Mulligan, "The 6LoWPAN architecture," in *EmNets '07: Proceedings of the 4th workshop on Embedded networked sensors*. New York, NY, USA: ACM, 2007, pp. 78–82.

[4] "Cyber-physical systems executive summary," Tech. Rep., March 2008. [Online]. Available: http://varma.ece.cmu.edu/Summit/

[5] E. A. Lee, "Cyber physical systems: Design challenges," in *International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC)*, May 2008, invited Paper.

[6] ——, "Cyber-physical systems - are computing foundations adequate?" in *Position Paper for NSF Workshop On Cyber-Physical Systems: Research Motivation, Techniques and Roadmap*, October 2006.

[7] A. Dunkels, "Full TCP/IP for 8-bit architectures," in *MobiSys '03: Proceedings of the 1st international conference on Mobile systems, applications and services*. New York, NY, USA: ACM, 2003, pp. 85–98.

[8] K. Mayer and W. Fritsche, "IP-enabled wireless sensor networks and their integration into the internet," in *InterSense '06: Proceedings of the first international conference on Integrated internet ad hoc and sensor networks*. New York, NY, USA: ACM, 2006, p. 5.

[9] J. W. Hui and D. E. Culler, "IP is dead, long live IP for wireless sensor networks," in *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems*. New York, NY, USA: ACM, 2008, pp. 15–28.

[10] "The IEEE 802.15.4 standard specification," Tech. Rep., 2006. [Online]. Available: http://www.ieee802.org/15/pub/TG4b.html

[11] ZigBee-Alliance, "The ZigBee standard specification," Tech. Rep., December 2006.

[12] L. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *Communications Magazine, IEEE*, vol. 40, no. 8, pp. 102–114, 2002.

[13] A. Cunha, R. Severino, N. Pereira, A. Koubâa, and M. Alves, "Zigbee over tinyos: Implementation and experimental challenges," in *8th Portuguese Conference on Automatic Control (CONTROLO2008), Invited Session on "Real-Time Communications: from theory to applications*, 2008.

[14] A. Cunha, A. Koubâa, M. Alves, and R. Severino, "open-ZB: an open-source implementation of the IEEE 802.15.4/zigbee protocol stack on TinyOS," in *The Fourth IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS07)*, October 2007.

[15] A. Kopke and J.-H.Hauer, "IEEE 802.15.4 symbol rate timer for telosb," TKN Technical Report Series TKN-08-006, Telecommunication Networks Group, Technische Universitat Berlin, Tech. Rep., May 2008.

[16] J.-H. Hauer, "TKN15.4: An IEEE 802.15.4 MAC implementation for TinyOS 2," TKN Technical Report Series TKN-08-003, Telecommunication Networks Group, Technical University Berlin, Tech. Rep., March 2009.

[17] P. Pagano, M. Chitnis, A. Romano, G. Lipari, R. Severino, M. Alves, P. Sousa, and E. Tovar, "ERIKA and OpenZB: an implementation for real-time wireless networking," in *24th ACM Symposium on Applied Computing (SAC 2009), Poster Session*, March 2009.

[18] The e.r.i.k.a. enterprise realtime operating system. [Online]. Available: http://www.evidence.eu.com/

[19] P. Jayachandran and T. Abdelzaher, "A delay composition theorem for real-time pipelines," in *ECRTS '07: Proceedings of the 19th Euromicro Conference on Real-Time Systems*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 29–38.

[20] ——, "Delay composition algebra: A reduction-based schedulability algebra for distributed real-time systems," in *RTSS '08: Proceedings of the 2008 Real-Time Systems Symposium*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 259–269.

[21] J.-Y. Le Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet (Lecture Notes in Computer Science)*, 1st ed. Springer, August 2001.

[22] L. Lenzini, E. Mingozzi, and G. Stea, "End-to-end delay bounds in FIFO-multiplexing tandems," in *ValueTools '07: Proceedings of the 2nd international conference on Performance evaluation methodologies and tools*. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 1–10.

[23] T. Abdelzaher, "CPS position paper: Towards an architecture for distributed cyber-physical systems," in *NFS workshop on Cyber-Physical Systems. Papers are available at http://varma.ece.cmu.edu/cps/*.

[24] K. Aberer, M. Hauswirth, and A. Salehi, "Infrastructure for data processing in large-scale interconnected sensor networks," in *The 8th International Conference on Mobile Data Management (MDM'07)*, 2007.

[25] B. Andersson, N. Pereira, W. Elmenreich, E. Tovar, F. Pacheco, and N. Cruz, "A scalable and efficient approach to obtain measurements in CAN-based control systems," *IEEE Transactions on Industrial Informatics*, vol. 4, 2008.

Session 3 – Keynote Talk 2

Chair: Jean-Dominique Decotignie

Rapporteur: Jean-Luc Scharburg

Real-Time Internet for Buildings?

Peter van der Stok

Philips Research Laboratories Eindhoven, The Netherlands

Abstract: Networking in buildings is characterized by as many standards as there are application domains. With the urge to save energy the need to integrate building networks increases. The most prominent integrating standard at this moment is the Internet Protocol stack. This talk discusses existing standards, requirements on the building networks (i.a. Real-time, wireless) and by how far IP satisfies these requirements. The talk concludes with a list of open topics.

Session 4

Chair: Björn Andersson

Rapporteur: Ivan Tarasov

A Real-Time and Robust Routing Protocol for Building Fire Emergency Applications Using Wireless Sensor Networks[•]

Yuanyuan Zeng, Cormac J. Sreenan

Mobile & Internet Systems Laboratory, Department of Computer Science, University College Cork, Ireland

{yz2, cjs}@cs.ucc.ie

Abstract

Fire emergency monitoring and evacuation for building environments is a novel application area for the deployment of wireless sensor networks. In this context, real-time delivery and robust routing is the essential operating criteria in order to ensure safe and timely building evacuation and application of fire fighting resources. Existing routing mechanisms for wireless sensor networks are not well suited for building fire, especially as they do not consider critical and dynamic network scenarios. In this paper, a novel real-time and robust routing protocol is presented for building fire emergency applications. The protocol adapts to handle dynamic emergency scenarios and works well with the routing hole problem. Preliminary analysis indicates that the protocol is well suited for building fire and other similar emergency scenarios.

1. Introduction

In the near future it can be expected that buildings will be equipped with a range of wireless sensors functioning as part of an overall building management system. Included in this set of sensors will be devices to monitor fire and smoke, allowing detection, localisation and tracking of fires. It is expected such information could be used for a variety of purposes, including guiding building occupants to the nearest safe exit, and helping fire fighting personnel to decide on how to best tackle the disaster. Fire/smoke sensors are expected to be programmed to report periodically and also when they detect a sensor input that exceeds a threshold. In the latter case, there is a need for real-time and robust message delivery. For example, a fire-fighter relies on timely temperature updates to remain aware of current fire conditions. In addition, as the fire spreads throughout the building it becomes likely that the sensing devices may become disconnected from the network or indeed be destroyed. So the routes have to be changed or re-discovered to adapt to these extreme conditions in order for the network to continue operating. Most existing routing protocols consider the energy efficiency and lifetime of the networks as the foremost design factor. The routing mechanisms used in general wireless sensor networks and even routing for forest fire applications are not well suited for in-building disaster situations, where timeliness and reliability is much more critical. For forest fires the focus is on tracking of fires, rather than evacuation or guidance of fire personnel. This combination of real-time requirements coupled with dynamic network topology in a critical application scenario provides motivation for our research. In this paper, we propose a real-time and robust routing mechanism (RTRR) designed especially for building fire emergency using wireless sensor networks (WSN), which provides timely and robust data reporting to the sink. To the best of our knowledge, this is the first time a real-time and robust routing mechanism suitable for building fire emergency using WSNs has been proposed.

This short workshop paper presents the core elements of our routing mechanism and a preliminary assessment of its expected behavior. Section 2 presents the related work. In Section 3, we outline the routing problem. We present the real-time and robust routing mechanism in Section 4. In Section 5, we present a preliminary analysis. Finally, Section 6 concludes this paper.

2. Background and Related Work

Most routing protocols for WSNs focus on energy efficiency and link node lifetime explicitly to its energy resources, i.e., a node is assumed to fail when the battery is depleted. Some WSN applications require real-time communication, typically for timely surveillance or tracking. For example, SPEED [2], MM-SPEED [3], RPAR [4] and RTLD [5] were designed for real-time applications with explicit delay requirements. But these routing protocols are not well suited for routing in building fire scenarios, where a critical and dynamic network scenario is also a key factor.

In this regard the work by Wenning et. al. [6] is interesting, where they propose a proactive routing method that is aware of the node's destruction threat and adapts the routes accordingly, before node failure results in broken routes, delay and power consuming route re-discovery. Fire emergency using wireless sensor networks within buildings is more challenging because of the complex physical environment and critical factors of fire hazards. Other researchers work on emergency guidance and navigation algorithms with WSNs for buildings. Tseng et. al. [7] propose a distributed 2D navigation algorithm to direct evacuees to an exit while helping them avoid hazardous

[•] The NEMBES project is funded by the Irish Higher Education Authority under the PRTLI-IV programme.

areas. Based on this, Pan et. al. [8] propose a novel 3D emergency service that aims to guide people to safe places when emergencies happen. Barnes et. al. [9] present a distributed algorithm to direct evacuees to an exit through arbitrarily complex building layouts in emergency situations. They find the safest paths for evacuees taking into account predictions of the relative movements of hazards, i.e., fires and evacuees. Tabirca et. al. [13] solve a similar problem but under conditions where hazards can change dynamically over time.

The "Routing Hole Problem" is a very important and well-studied problem, where messages get trapped in a "local minimum". Some existing "face routing" algorithms have been developed to bypass routing holes using geo-routing algorithms. GPSR [1] recovers holes by using the "right-hand rule" to route data packets along the boundary of the hole, combining greedy forwarding and perimeter routing on a planar graph. In [10], the authors propose the first practical planarization algorithm with reasonable message overhead, lazy cross-link removal (LCR). Fang et al [11] present an interesting approach, the BOUNDHOLE algorithm, which discovers the local minimum nodes and then "bounds" the contour of the routing holes. In the building fire situation, holes feature prominently and can be expected to grow in size rapidly as a fire spreads, thus demanding solutions that are robust and low complexity for quick reactions.

3. Definitions

Given a homogeneous WSN deployed in a building for fire emergency applications with N sensors and M sinks. Each sensor can adjust its maximal transmission ranges to one of the k levels: r_0, r_1, …., $r_{k-1}=r_{max}$ by using different transmission power levels from $p_0, p_1,$ till $p_{k-1}=p_{max}$. Initially, all sensors work in p_0. T_{max} is the maximum acceptable delay in reporting such a fire emergcncy event to a sink node. It is required that each sensor i will report data packets to a sink node, such that:

(1) A communication path from sensor to the sink can be found if such a path exists.

(2) The end-to-end delay of the path is no more than T_{max}.

(3) The choice of route is adaptively changed in response to failed nodes (assumed to be caused by fire damage).

(4) A suitable minimized power level ($min\{ p_0, p_1, …., p_{k-1}\}$) is selected to ensure transmission to satisfy (1),(2),(3) without unnecessary power dissipation.

Each node in the network exists in one of four states (listed in the order of health degree from best to worst):

(1) *safe*: while no fire occurs. Initially, all nodes are in this state.

(2) *lowsafe*: when a sensor is one-hop away from an "*infire*" node.

(3) *Infire:* when a sensor detects fire.

(4) *unsafe*: the sensor detects that it cannot work correctly due to a definite fire, and takes this state to indicate imminent failure.

There is a STATE message recording current node state to notify its neighbourhood nodes in a fire.

(1) STATE(INFIRE) message: If a sensor detects the fire, it enters "*infire*" by broadcasting the message to denote a new local fire source.

(2) STATE(LOWSAFE) message: The nodes in "*safe*" that receive a STATE(INFIRE) message will become "*lowsafe*" and then broadcast this message. The other nodes that hear the STATE(LOWSAFE) message only ignore it.

(3) STATE(UNSAFE) message: An "*infire*" node works until it cannot work correctly, then it enters into "*unsafe*". Any nodes that detects its residual energy is too low to work will also enter into "*unsafe*", and then broadcast a STATE(UNSAFE) message.

Thus each sensor may change its state autonomously in response to the local fire condition and messages it receives, as shown in Figure1.

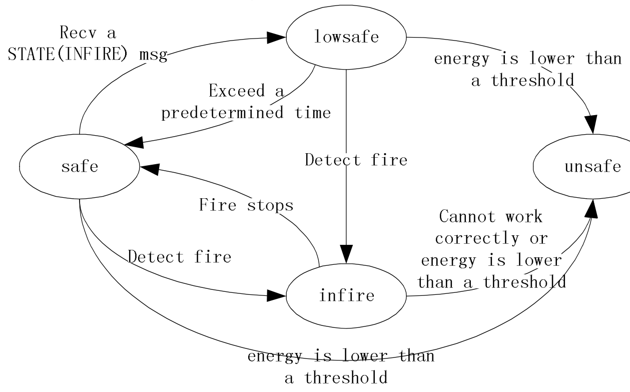

Recv a STATE(INFIRE) msg
lowsafe
energy is lower than a threshold
Exceed a predetermined time
safe
Detect fire
unsafe
Fire stops
Cannot work correctly or energy is lower than a threshold
Detect fire
infire
energy is lower than a threshold

Fig.1 State Transition diagram for each node.

4. Protocol Description

4.1 Initialised Routing Structure

A. Initialized Sink Beacon

The purpose of routing initialisation is to form initialised neighbourhood and routing structure after the sensors are deployed and connected as a WSN in the building. We assume that sinks are deployed in a relatively safe place such that they are less likely to be destroyed for example due to walls collapsing. Once the network is deployed, each sink generates a HEIGHT message using power level p_0. This serves to advertise to neighbouring nodes and includes a "*height*" parameter that represents the hop count toward the sink, and is initialised to 0. The height value is incremented by each forwarding hop. Each node records the height information in its local neighbourhood table when it receives the first HEIGHT message. The message contains a sequence number so that a node can determine if it has seen the message already, in which case it ignores it. If it is the first time that it receives a HEIGHT message, the node forwards the HEIGHT message out. As explained below this process serves to ensure that each node will know a minimal delay experience from sink toward node.

B. End-to-End HEIGHT Delay Estimation

In this HEIGHT message broadcasting process, the end-to-end delay from a node to the sink could be approximated by the cumulative delay on each hop. We use this delay estimation in our RTRR routing mechanism to make the forwarding choice.

We denote *delay*(*sink*, *i*) as the delay experienced from the sink to each node, and then we could use *delay*(*sink*, *i*) as a bound to guide a real-time delivery from node to sink. The delay in transmitting a packet is estimated as:

$$delay(\text{sink}, i) = \sum_{n=1}^{h} Avg\_delay = \sum_{n=1}^{h} (T_c + T_t + T_q) * R \quad (1)$$

where, n is the hop count from the sink to node i, T_c is the time it takes for each hop to obtain the wireless channel with carrier sense delay and backoff delay, T_t is the time to transmit the packet that determined by channel bandwidth, packet length and the adopted coding scheme, T_q is the queuing delay, which depends on the traffic load, and R is the retransmission count. Among them, we omit the propagation delay, as in a WSN this is negligible due to the use of short-range radios.

The average end-to-end delay from each node to the sink can be computed as the cumulative delay hop-by-hop, and the delay experienced on current hop is calculated and updated locally, and then recorded in HEIGHT message. Then *delay*(*sink*, *i*) is recorded into neighbourhood table of each node. We use periodic HEIGHT message update to calculate an average end-to-end delay (from multiple sink-to-node delay estimations) as reference. Since packets in WSNs always tend to be relatively small, we consider it reasonable to ignore any impact of delay differences related to packet size.

Furthermore, our delay estimation utilises Jacobson's algorithm [12] to consider both the weighted average and variation of the estimated variable and as a result provides a good estimation of the delay. It can work well when link quality and network load varies. The calculation of average end-to-end delay and variation avoids a large number of deadline misses due to high variability in communication delays.

Since the traffic from node to sink is usually heavier than the traffic from sink to node in the same channel situation according to sensor applications, so the queuing delay $T_q(sink, i) \leq T_q(i, sink)$. This is bounded by the maximum queueing delay so $T_q(i, sink) \leq T_{qmax}$. When assuming the same radio and link quality for downstream and upstream links on the counterpart route path, we can get that: $delay(i, sink) \leq delay_{qmax}(sink,i)$. $delay_{qmax}(sink,i)$ is the delay experienced from sink to i with the maximal delay on queuing. Then our delay estimation and realistic delay on the route path T satisfy that: $delay(sink, i) \leq T \leq delay_{qmax}(sink,i)$.

We can use *delay*(*sink*, *i*) as a bound to guide the real-time routing forwarding selection. If the delay and slack time meets the estimated delay time for data delivery, the packet has a high probability to arrive before deadline and thus ensures real-time communications.

C. Periodic Sink Update

With the HEIGHT message broadcast process, an initial neighbourhood is formed by each sensor for which it records neighbour ID, height, state, estimated delay, residual energy of all neighbours, as well as the transmission power that the node uses to communicate with its neighbour on the path to the sink. Each sensor records its own ID, state, and residual energy. Also, each node maintains sink ID with its minimal delay sink. In fire scenario, the sink may become disabled and the networks topology will be changed with fire. To ensure robust connectivity each sink will periodically send out a HEIGHT message to refresh the networks. The refresh rate is a protocol design parameter that trades off overhead for increased robustness to lost HEIGHT messages and path changes. In a fire situation one would expect to decrease the period, although the impact on network traffic load must also be examined.

4.2 Routing Mechanism Details

A. Forwarding Metrics

For a given application-specific T_{max}, we use *slack* to remember the time left on the path from the current node to the sink. Each node in the neighbourhood table is associated with a *forward_flag* and a *timeout*. The flag is used to identify the next hop as a best forwarding choice, i.e., when a node is chosen as the best forwarding choice, the *forward_flag* is set to 1. The *timeout* value is the valid time for the current forwarding node and used to prevent stale neighbourhood information (introduced in Section 4.3.) If the timeout of a forwarding choice is due, then its forwarding flag is set to 0 to evict the stale relay node.

To select the best forwarding choice from the local neighbourhood table:

(1) Firstly, we filter the forwarding choices by the height to choose the nodes with lower height.
(2) Secondly, choose the node with and enough slack time according to delay estimation on the path.
(3) Thirdly, we filter the remaining forwarding choices by node state in the priority from "*safe*" to "*infire*".
(4) If there is more than one node satisfied, we select the best forwarding choice with the higher residual energy. If there is still a tie, we choose the lower ID.

If we cannot find a best forwarding choice with the current transmission power, we say that a "hole" has occured (i.e., stuck in local minimum).

B. Hole Problem Handler by Adapting Power Level

If a sensor node cannot find a next hop that satisfies the real-time constraint with current power level, it means that the node is stuck in a local minimum.

The solution is to increase the transmission power gradually by levels to find another neighbour or invoke a new neighbour discovery. Otherwise, a notify message is sent to its upstream node (i.e., parent) to stop sending data packets to the current node; and then a routing re-discovery is invoked by the upstream node.

If we could find another node existing in the neighbourhood table by adapting the transmission power, then we increase the power level and name this neighbour as a forwarding choice. Otherwise, a new neighbour discovery is invoked by increasing the transmission power gradually by levels.

Figure 2 shows an example of a new neighbour discovery. *sink1* and *sink2* are two sinks, and the other nodes are sensors. Node *i* reports and routes the data to the sink. The number on each node represents the "height" of each node toward the sink. As the route path {*i*, *a*, *sink1*} with p_0 is invalid because slack does not satisfy the estimated end-to-end delay, node *i* is in the "hole". If there are no existing eligible neighbours, then *i* will increase its power to p_1 to reach node *j* and delivers the packets to another sink *sink2* by route path {*i*, *j*, *sink2*} when slack on this route is no less than delay estimation.

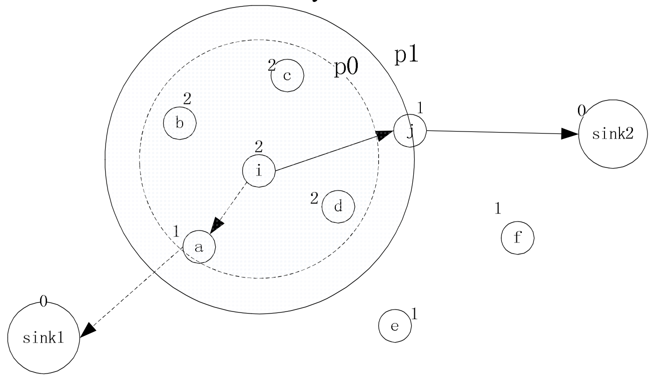


Fig.2 New neighbour discovery to solve routing "hole"

Each sensor has *k* levels of power setting: $\{p_0, p_1, p_2, \ldots p_{k-1}\}$ and could be in *k* levels of maximal transmission range as: $\{r_0, r_1, \ldots r_{k-1}\}$. We defined a function to find appropriate transmission power by increasing the power as follows:

$$p = p_{cur+\imath+1},\ \imath = 1, 2, 3 \ldots k-1 \tag{2}$$

where, *cur* is the current number of transmission range level among *k* levels, $\imath$ is the count of unsuccessful attempts. A sensor will increase its transmission power gradually in levels if it cannot find an eligible new neighbour.

A node increases its power according to formula (2) until one of the following conditions is satisfied:

(1) It finds a node as a forwarding choice in "*safe*" state according to forwarding metrics of lower height and delay estimation described in section *A*.

(2) When $p=p_{max}$; in this case, it finds the new neighbour as a forwarding choice by forwarding metrics of lower height and delay estimation in a priority from "*safe*", "*lowsafe*" to "*infire*"; otherwise, no eligible new neighbour is found.

In the new neighbour discovery process, sensor *i* will broadcast out a Routing Request (RTR) message. In the neighbour discovery, sensor *i* piggybacks *height*, *slack* and the newly adapted power *p* in RTR message. For a node *j* that hears the message, if the estimated end-to-end delay is no more than *slack* and its height is lower than *height*(*i*, *sink*), as well as its state is "*safe*", then *j* is selected as a new neighbour. If sensor *j* hears the RTR with P_{max}, and if its height is lower than *height*(*i*, *sink*), then *j* is selected a new neighbour when *j* is not in "*unsafe*" state. The new neighbour will reply to node *i* with the same power that node *i* is using, after a random backoff to avoid collisions. The forwarding choices send reply message with *power*(*i*) only as necessary for reaching node *i*, otherwise reverting to their previous power level. Upon receiving the reply, node *i* inserts the new neighbours into its neighbour table. And during the RTR and reply message exchange, we could calculate the delay between *i* and its new neighbour *j* as follows:

$$Ave\_delay(i,j) = Round\_trip\_time/2 \tag{3}$$

For meeting real-time requirement, the forwarding choices should satisfy that slack is no less than the average delay between *i* and *j* plus the delay estimation at node *j*:

$$slack(i) \geq Ave\_delay(i,j) + delay(sink, j) \tag{4}$$

If there is more than one new neighbour found, a best forwarding node is elected by the priority of state from "*safe*", "*lowsafe*" to "*infire*". If there is still a tie, the best relay is elected by the node with higher residual energy and lower ID number.

For a node that works in a larger transmission range could still be adapted to decrease the transmission power to improve energy efficiency and network capacity, while delay deadline is loose. So we define when a node detects a good connectivity with safe neighbourhood that is larger then a predefined threshold, i.e., $|Neighbour_{safe}| > N\_threshold$, power decrease process is invoked.

We defined a function to find appropriate transmission range by decreasing transmission power as follows:

$$p = p_{cur-\imath'},\ \imath = 1, 2, 3 \ldots k-1 \tag{5}$$

where, *cur* is the current number of transmission power level among *k* levels. $\imath'$ is the count of decrement.

A node is eligible for power decrease until:

(1) The minimum power has been reached.

(2) There are two consecutive power levels such that at the lower level the required delay is not met but at the higher power level the required delay is met.

(3) There are two consecutive power levels such that at the lower level the required safe neighbourhood connectivity *N_threshold* is not met but at the higher power level it is.

C. Neighbourhood Table Management

The neighbourhood table records information including transmission power for reaching the neighbour nodes, and is updated by periodic HEIGHT messages from sinks. For power adaptation and new neighbour discovery, the neighbourhood table will be updated with the new neighbours and new transmission power. The node also updates its neighbourhood with the neighbours and new states as they change. If it receives a STATE(UNSAFE) message, the unsafe neighbour is removed from the table.

4.3 Routing Reconfiguration

In building fire emergencies, robust data routing is crucial due to the impact of quickly moving fire on node liveness. In this section we explain how we reconfigure to deal with failures. We assume that: (1) the minimal time interval between "*infire*" and "*unsafe*" state of a node is chosen as a parameter known beforehand and denoted as t_{unsafe}. (2) We use necessary transmission range for connectivity between nodes (according to selected power level) to approximate the minimal fire spreading time between two nodes. In practice there are well-known guidelines for estimating the rate of fire spread, taking into account building materials, etc. It's also the case that obstacles, such as walls, that mitigate radio propagation also have the effect of slowing fire spread.

When a forwarding choice is used for routing, we add a *timeout* to avoid the use of stale and unsafe paths, i.e., every node on the path from source *s* to destination *d* has a *timeout* to record the valid time of each link on this route. The *timeout* is updated when node state changes occur among the neighbourhood. The forwarding choice that exceeds the *timeout* value is considered invalid and then evicted.

We assign an initialised large constant value to represent the estimated valid time for the node in "*safe*" state.

When the neighbour node of node *i* is caught in fire, a STATE(IN-FIRE) message is broadcast. If node *i* is in safe and receives a STATE(IN-FIRE) message from its neighbours, node *i* will enters into the "*lowsafe*" state. The timeout of node *i* is updated, i.e., the valid time of node *i* is updated, as the minimal time that this node may be caught in fire and until it is out of function:

$$timeout(i)=\min(spread\_time(i, fireneigh(i)))+t_{unsafe} \quad (6)$$

Then the *timeout* value of both downstream and upstream links that are adjacent to node *i* are also updated accordingly.

If node *i* becomes "*infire*", the *timeout* of adjacent links are updated as t_{unsafe}, i.e., $timeout(i)= t_{unsafe}$.

Otherwise, if node *i* becomes "*unsafe*" by local sensed data and threshold, then *timeout*(*i*) is updated as 0 and the *timeout* of the adjacent links are also updated to 0.

The link *timeout* value is updated as the state of the node adjacent to the link changes. When a node state is changed in the fire, the upstream and downstream links that are adjacent to this node will update the timeout on both links. For path link (*i*, *j*) on each route path, the *timeout* value for this link is calculated as:

$$timeout(path(i, j))=\min(timeout(i), timeout(j)) \quad (7)$$

where *timeout*(*i*) and *timeout*(*j*) represents the valid time for each node *i*, *j* of the route in fire.

In a building fire, node failures because of fire damage will trigger routing tree reconfiguration. In case of a path link *timeout* value that is lower than a threshold (i.e., the route path will be invalid very soon), a route reconfiguration is invoked to find another available route path before the current one becomes invalid. The reconfiguration is only invoked by an upstream node *i* of the path link (*i*, *j*) whose valid time is no less than the timeout of the link, i.e., $timeout(i)\geq timeout(path(i, j))$. The routing reconfiguration of the node is invoked as a routing re-discovery by broadcasting a RTR message to set up a new route path search. The search of the forwarding choice is invoked in its neighbourhood table to find if one of the existing neighbours is eligible to act as a relay or not by adapting the power to the setting recorded in local neighbourhood. Otherwise, we will start a new neighbour discovery process by increasing its power gradually.

The re-discovery stops when it finds another forwarding choice with a valid route path cached toward one of the sinks (may be a different sink from current one).

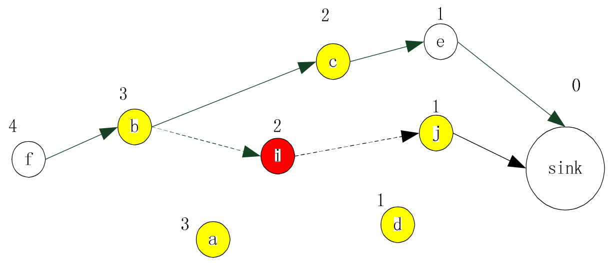


Fig.3 Timeout update in fire and route reconfiguration

Figure 3 shows an example for timeout update in fire. For sensor *f*, it reports to sink by route {*f*, *b*, *i*, *j*, *sink*}. After working for a while, sensor *i* (colored red) senses the fire occurrence. Then sensor *i* broadcasts a STATE(IN-FIRE) message to notify its communication neighbours(colored yellow): *a*, *b*, *d*, *j*, and *c*. These nodes when they receive the message will enter into the "*lowsafe*" state. For the state change of sensor *i*, then *timeout*(*i*) is updated as t_{unsafe}. Accordingly, sensor *i* will update the timeout of its upstream and downstream link: (*b*, *i*) and (*i*, *j*). As our designed condition for reconfiguration, when *timeout*(*path*(*b*, *i*)) and *timeout*(*path*(*i*, *j*)) is lower than a predetermined threshold, the routing reconfiguration is invoked by the upstream node whose timeout is no less than the link timeout. Then sensor *b* will broadcast a RTR message to find a new relay to the sink, i.e., route path {*f*, *b*, *c*, *e*, *sink*}. When it comes to path link (*i*, *j*), sensor *i* is the upstream node of the link with the lower valid time, then it will still work on this path (to forward data from sensor *i* to sink if needed) until sensor *i* becomes unsafe.

It is assumed that data packet acknowledgements are sent at the link layer (not end-to-end). When a node does not receive an acknowledgement after a certain time, we assume the downstream link is invalid and reconfigure routing.

5. Analysis

Lemma 1. The routing graph of the sensor network is loop-free.

Proof. We suppose that there exists a loop A→B→C→D→E→⋯→A in the routing graph. Each node selects its next node which is has less height (hop count) towards the sink. When each node is stuck in local minimum, the node could increase its transmission range to find another node that has less height towards the sink if it exists. So we can get:

height(A)<⋯<*height*(E)<*height*(D)<*height*(C)<*height*(B)< *height*(A). This is a contradiction, so we conclude that the routing graph of our routing is loop-free.

Theorem1. If a route from sensor nodes to the sink exists, RTRR can find the route to the sink.

Proof: From Lemma 1, we know that there is no loop in the routing graph. So the next hop of each node is routed towards the sink node. Since the number and height of sensor nodes is limited, so the routes will lead to the sink eventually as long as the real-time route exists.

Theorem2. If the real-time route from node to sink exists, RTRR can find such a route path.

Proof. We denote *delay*(*sink*, *i*) as the delay estimation that is the minimal delay experienced from sink to node, while delay(*i*, *sink*) as the delay from node to sink on the counterpart route path. We denote *T*(*i*, *sink*) as the realistic delay experienced from a node to the sink.

In RTRR, we use *delay*(*sink*, *i*) as estimation of delay time form node to sink in routing discovery to find a route that meets the lower delay threshold, i.e., using *delay*(*sink*, *i*) to estimate *T*(*i*, *sink*).

Since we measure average delay with HEIGHT message using power p_0, we get the maximal delay estimation time *delay*(*sink*, *i*) on the minimum delay experienced from sink to node within different power levels. In RTRR, we find a relay node *i* that delay *T* from *i* to the sink with this route path should satisfy that it is no larger than the delay estimation on the route path, where $T_{slack} = T_{max} - T(s, i)$:

$$delay(sink, i) \leq T_{slack}$$

So, $T(s, sink) = T(s, i) + T(i, sink) = T(s, i) + delay(sink, i) \leq T(s, i) + T_{slack} \leq T(s, i) + T_{max} - T(s, i) \leq T_{max}$.

Otherwise, we increase the power level to find another forwarding choice *j*, and such a node *j* (with increasing power) exists by satisfying:

$$delay(sink, j) + Ave\_delay(i, j) \leq T_{slack}$$

where $T_{slack} = T_{max} - T(s, i)$.

So, the end-to-end delay time *T* satisfies: $T(s, sink) = T(s, i) + T(i, sink) \leq T(s, i) + T(i, j) + T(j, sink) \leq T(s, i) + Ave\_delay(i, j) + delay(sink, j) \leq T(s, i) + T_{slack} \leq T_{max}$.

So we find a route from node *s* to sink that satisfies: $T(s, sink) \leq T_{max}$.

From the above situations, if a real-time route exists, RTRR can find a route satisfying that the end-to-end delay from s to sink is within the delay requirement T_{max}.

6. Conclusion and Future Work

We present a novel real-time and robust routing mechanism that is designed especially for building fire emergencies. The probability of end-to-end real-time communication is achieved by maintaining a desired delay based on estimation and power level adaptation. The design takes into account realistic application characteristics including fire expanding, shrinking and diminishing. Our routing mechanism is designed as a localized protocol that makes decisions based solely on one-hop neighbourhood information.

We are currently developing our protocol for evaluation in the ns-2 simulator, which will also allow comparison with other related real-time protocols in WSNs. Future work will include implementation on a testbed we have deployed at our university.

7. References

[1] B. Karp, and H. T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *ACM MobiCom*, 2000.

[2] T. He, J. Stankovic, C. Lu, and T. Abdelzaher, "SPEED: A Stateless Protocol for Real-time Communication in Sensor Networks," *ICDCS'03*, 2003.

[3] E. Felemban, C. –G. Lee, E. Ekici, R. Boder, and S. Vural, "Probabilistic QoS Guarantee in Reliability and Timeliness Domains in Wireless Sensor Networks," *InfoCom'05*, 2005.

[4] O. Chipara, Z. He, G. Xing, Q. Chen, et. al., "Real-time power-aware routing in sensor networks," *14th IEEE International Workshop on Quality of Service*, 2006.

[5]A. Ahmed, N. Fisal, "A real-time routing protocol with load distribution in wireless sensor networks," *Computer Communications*, 31(14), pp.3190-3203, 2008.

[6] B.-L. Wenning, D. Pesch, A. Timm-Giel, C. Gorg, "Environmental monitoring aware routing: making environmental sensor networks more robust," *Telecommunication Systems*, 2009.

[7] Y.-C. Tseng, M.-S. Pan, and Y.-Y. Tsai, "Wireless sensor networks for emergency navigation," *IEEE Computer*, 39(7), pp. 55-62, 2006.

[8] M.-S. Pan, C.-H. Tsai, and Y.-C. Tseng, "emergency guiding and monitoring applications in indoor 3D environments by wireless sensor networks," *Int. J. of Sensor Networks,* 1(2), pp. 2-10, 2006.

[9] M. Barnes, H. Leather, D. K. Arvind, "Emergency evacuation using wireless sensor networks,"*32nd IEEE Conference on Local Computer Networks*, pp. 851-857, 2007.

[10] Y-J. Kim, R. Govindan, B. Karp, S. Shenker, "Lazy cross-link removal for geographic routing," *ACM SenSys'06*, 2006.

[11] Q. Fang, J. Gao, L. Guibas, "Locating and bypassing holes in sensor networks," *Mobile Networks and Applications*, 11(2), pp. 187-200, 2006.

[12]V. Jacobson, "Congestion avoidance and control," *SIGCOMM'88*, 1988.

[13] T. Tabirca, K. Brown, C.J. Sreenan. "A Dynamic Model for Fire Emergency Evacuation Based on Wireless Sensor Networks," *8th International Symposium on Parallel and Distributed Computing (ISPDC),* July 2009, to appear.

Assessing the Clusters Formation in the HCT-MAC Protocol

Marcelo Maia Sobral
Dept of Automation and Systems
Federal University of Santa Catarina
Florianopolis, SC - Brazil
Email: sobral@das.ufsc.br

Leandro Buss Becker
Dept of Automation and Systems
Federal University of Santa Catarina
Florianopolis, SC - Brazil
Email: lbecker@das.ufsc.br

***Abstract*—A real-time MAC protocol for applications like mobile wireless ad hoc networks (MANET) and wireless sensor networks must transmit frames in bounded time, and adapt quickly to topology changes. Our recently proposed HCT MAC protocol, targeted to such applications, presents a solution that self-organizes the network in nodes agglomerations called clusters, and provides a TDMA-like medium access within them. In this paper we present a performance evaluation of HCT-MAC to assess the resulting cluster formation in some key scenarios, analyzing the clustering delays and the number of clusterized nodes. We also highlight the advantages of HCT-MAC in comparison to a naive CSMA/CA, in respect to both medium utilization and transmission delays.**

I. Introduction

The emerging of a new generation of applications will require communication capacity in environments without any infrastructure. To make the problem more complex, such applications might be composed by mobile nodes, which rely on wireless links to achieve communicability. The literature recently proposed the term MANET (*Mobile Ad-Hoc Networks*) to represent such application domain. Some MANET applications require time guarantees with respect to the communication medium. An example of such applications include vehicle-to-vehicle (V2V) systems [1], like platooning, that help ease of traffic congestion and safe driving. New areas of research in the space community have similar communication requirements, as for example in distributed satellite systems (DSS) [2] where multiple spacecraft in varying configurations are used to achieve a mission's goals collaboratively.

Recently we proposed a hybrid medium access control mechanism named HCT (Hybrid Contention/TDMA-based) MAC [3], which aims to provide a deterministic medium access by means of resource reservation while also supporting reconfiguration and mobility by using a contention-based approach. Therefore, it assumes that the mobile nodes are self-organized in clusters, used exclusively to support the allocation of time-slots within the corresponding member nodes. Ideally, cluster should be defined in a way to allow as many nodes as possible to operate in resource-reservation mode. In [4] we adressed the problems related to the dynamic self-organization of clusters. More specifically, we presented an approach to establish the clusters in a distributed and autonomous way, taking into account the quality of communication among the member-nodes.

In this paper we present a performance evaluation of the HCT-MAC to assess the resulting cluster formation in some key scenarios. To be more specific, we measured the clustering delays and the rate of clusterized nodes, since this has a direct influence in the predictable mode of operation of HCT (the TDMA phase). The clustering delays should be as small as possible to allow a fast cluster formation and adaptation to topology changes. Also, the more nodes that become cluster members the better, allowing them to transmit messages in resource-reservation mode. Our evaluation also includes a comparison of HCT-MAC with a naive CSMA/CA, in order to show that besides the related overhead from HCT-MAC it has advantage over CSMA/CA. Currently we explore static scenarios in order to obtain the values (delay and utilization bounds) that are originated from the protocol itself. In order words, it means that we want to discover the minimum expected delay for a cluster to be formed.

The remainder of the paper is structured as follows: section II gives a brief overview of the HCT and its clustering strategy. Section III presents preliminary results obtained through simulation, that shows the resulting clusters for small networks with random topologies, including the clustering delays. This section includes a comparison between the HCT and a naive CSMA/CA regarding the network utilization. Finally, section IV concludes the paper and points out some future directions for our work.

II. HCT-MAC Protocol

A. Overview

In [3] we presented the Hybrid Contention/TDMA-based (HCT) MAC, which aims to provide a time bounded medium access control to mobile nodes that communicate through an ad-hoc wireless network. A key issue in this protocol is to self-organize nodes in clusters (i.e: set of neighbor nodes). Using clusters it is possible to solve the problem of timely transmission of messages. Our protocol assumes as basic requirements a periodic message model, where the assignment of slots must be done in a local manner. A competition strategy is adopted, without the need of a global coordinator nor

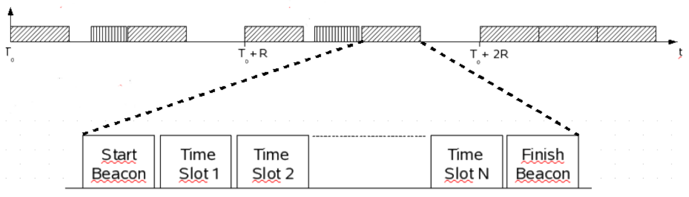


Fig. 1. Timing in the HCT: cycles of length R divided in superframes

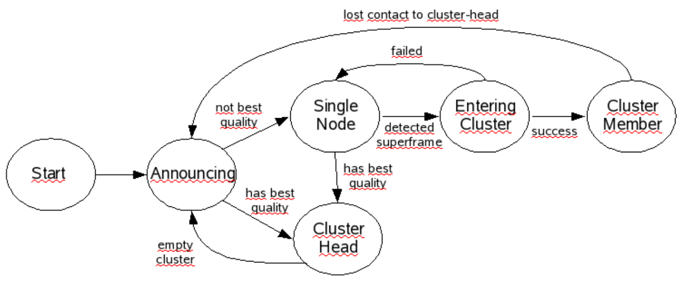


Fig. 2. Overview of clustering in the HCT protocol

scheduler, in such a way that time-slots are iteratively allocated by the nodes.

The HCT-MAC is a hybrid protocol because it has both contention-based and resource-reservation characteristics. A TDMA-based MAC protocol divides time in so-called time-slots, being responsible to assign one or more time-slots to each node. To solve this problem, the HCT-MAC uses a contention-based approach to allocate slots: if a node knows which slots are idle, it can try to use some of them, chosen randomly, and then verifies if any collision has occurred. For each chosen slot, if there was no collision, the node may assume that the slot is allocated. For the remaining slots, it can repeat the procedure until all the needed slots are allocated or, in the worst-case, no slots are available anymore. This protocol assigns the time-slots iteratively, until the allocation stabilizes, i.e. the nodes allocate all needed slots. The already allocated time-slots can be used just like in a TDMA protocol, revealing the resource-reservation aspect of the HCT-MAC.

The timing in the HCT has a periodic and hierarchical structure, as ilustrated in figure 1. A *cycle* is the base period for transmissions, thus it works like a time unit for the protocol. It is an interval of time that is common to all clusters, and is divided in *superframes*, which are assigned to the clusters. Superframes are all equal in size, what means that they contain the same number of *time-slots*, plus two control frames called *beacons*. Therefore, clusters need to allocate superframes, and cluster members allocate time-slots within those superframes.

The TDMA component of the HCT, described in [4], depends on the clustering of the nodes, which must be obtained in a self-organized manner. This is due to the fact that the HCT protocol is designed to be used in mobile ad-hoc networks, where the nodes are not previously aware of the topology, neither of their neighborhoods. The chosen approach relies on initial contention-based access, that shifts gradually to time-based as clusters are formed and become stable. That means, as nodes self-organize in clusters, they can reserve bandwidth and transmit messages in a timely manner. This implies procedures for the neighborhood discovery, the collection of information to support the independent choice of the best candidate nodes to start clusters, and the announcement of new clusters and ingress of interested nodes.

B. The Clustering Approach

Clusters are simply sets of nodes that agree to share a superframe, which represents a portion of the network bandwidth. A key element in the cluster topology is the cluster-head, a special node responsible to start cluster transmissions, to account for idle and used time-slots, and to report successful transmissions. Ideally, the cluster-head should be the node with the best neighborhood quality within the region to be covered by the cluster, in order to minimize the probability of errors in transmissions inside the cluster. In our proposed HCT it is not possible to determine exactly the nodes with best neighborhood quality, since no node has a global view of the network, and no global information is maintained by the protocol. But the information about the neighborhoods, estimated based on the received frames, can be collected and shared locally among the nodes to help them to decide to become or not cluster-heads. Thereby, the cluster-heads can be self-elected according to the information they are able to obtain about the nodes around them.

The rule for establishing clusters is guided by the fact that the cluster-head should be the node with the best neighborhood quality. The quality of a neighborhood is defined in this work as the sum of the qualities of links towards a node. Instead of an average, the sum expresses both quantity of existing links and their qualities. For instance, a node with four good links is a better cluster-head candidate than other with two links of comparable quality. A high quality means few or no transmission errors (missed or corrupted frames) or, in other words, a high frame reception rate. Therefore clusters have higher probability to be stable, because more frame losses would force member nodes to try to bind to other clusters. According to the signal model described in [5], the frame reception rate can be estimated using the RSSI (Received Signal Strength Indication), an information commonly given by radio transceivers.

As can be seen in the diagram in figure 2, the clustering in the proposed protocol has a number of phases and works continuously. The protocol starts with a discovering phase, represented by state *Announcing*. In this state each node transmits randomly an announcement frame, at most once per cycle, during a predefined number of cycles. The announcement frame includes a value for the neighborhood quality, which is computed using the RSSI of the received frames during the last cycle, as detailed in the next subsection. Meanwhile, it also listens to the medium and all received frames have their RSSI recorded.

The next step of the protocol occurs after the announcement cycles. In this moment each node decides whether to become a cluster-head if the following conditions hold:

- the node is not alone;
- its neighborhood has a better quality than reported by its neighbors;
- there exists an idle superframe;

A node experiences a delay to become a cluster-head, and this depends on network characteristics like network size and density, and also on HCT parameters like cycle length and superframe size. Such delay is called clustering delay, and is considered the expected time to establish a cluster. After a cluster is established, it can accept other nodes as cluster members. Cluster members also are subject to delays until they become fully integrated to the clusters, as demonstrated in [3]. The clustering delay is an important result of the protocol, because it imposes a limit on the frequency of major topology changes - i.e. changes that destroy an existing cluster and cause the formation of a new one. Ideally it should be limited to a previously known number of cycles, however this is still under investigation in the HCT.

It is expected that a small percentage of the nodes will have these conditions satisfied and will behave as potential cluster-heads (state *Cluster Head* in figure 2). Those nodes that are not included in this group will then behave as single nodes. In this case, they remain listening to the medium, waiting either for transmissions of superframes or for announcement frames. Thereby, each of these nodes can update continuously its neighborhood quality. If a superframe is detected, the node tries to allocate a time-slot and, if succeeded, it becomes a cluster member. Once it becames a cluster member, a node only exits the cluster in case it loses contact with the cluster-head. In this case, the node returns to the discovery phase, restarting the clustering part of the protocol. In figure 2, these sequences are represented by states *Single Node*, *Entering Cluster* and *Cluster Member*, with the corresponding transitions.

III. Simulation Analysis

The HCT protocol was designed to be self-organizing and time-bounded, and to allow a higher medium utilization compared to a pure contention-based MAC. However, this implies some relevant questions: i) which clusters appear in a network; ii) which is the delay until the clusters are established; and iii) how does the HCT compares to a CSMA/CA protocol with respect to network utilization and transmission delays. The resulting clusters in a given network depend on the current network topology and on the tuning of some HCT parameters, such as the cycle length and the superframe size. There exists an optimal solution for each network topology, where each cluster-head is the node with the best possible neighborhood quality. Thereby, the clustering obtained by the HCT should be as close as possible to this optimal solution. Besides that, the clusters should be established within an acceptable time, otherwise the HCT could not adapt properly to topology changes. and therefore it would be better to transmit in a contention-based manner.

To assess the correct behavior of the HCT, and to obtain lower limits for the adaptation delays, only static scenarios were considered. The present simulation analysis focus on evaluating the delays for cluster formation plus the network utilization compared to a CSMA/CA protocol, while analyzing the optimality of the solution is left for future work.

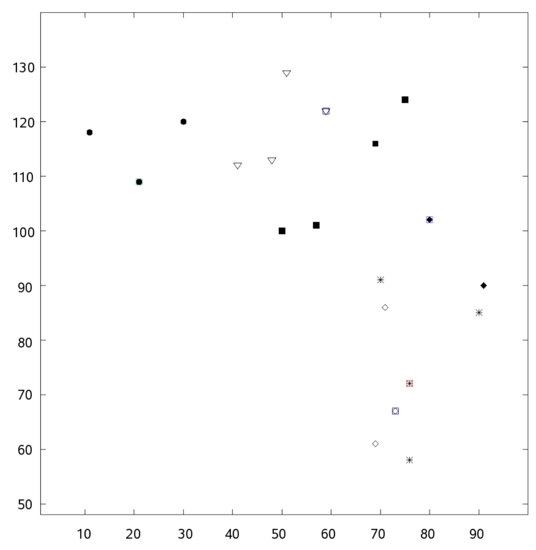
The CSMA/CA was chosen initially due to its contention-based approach for the medium access, where the nodes do not share any common information and do not perform any negotiation. This makes the CSMA/CA highly adaptable to topology changes, at the expense of an unpredictable timing behavior. The HCT uses similar procedures to access the medium in its contention-based phase, until it succeeds to change to resource-reservation mode of operation. There exists an intrinsec overhead in the HCT until the nodes self-organize in clusters, because it takes one or more cycles to establish clusters and there is the risk of transmission errors. Moreover, if the clustering becomes unstable the nodes can loose many cycles to reconfigure themselves, decreasing the expected performance of the protocol. Although it is expected that the resource-reservation mode of the HCT compensates its overhead, it is not obvious if it performs better than a simple CSMA/CA.

To perform the simulations we used the Omnet++ simulation framework [6], and the physical layer model included in the Castalia framework developed by the National ICT at the University of Australia [7]. The Castalia framework is based on the signal model described in [5], which defines the probability of frame reception as a function of the distance between nodes. It also implements an additive collision model, which compares the sum of interfering transmissions to the SNR (signal to noise ratio) at the receiver to decide if there was a collision. The physical layer gives a 250 kbps IEEE 802.15.4 compatible channel, and relies on a simulated CC2420 transceiver. It was configured with a transmission power of -15 dBm and noise floor of -100 dBm; the wireless channel was configured with a path loss of 55 dBm at 30 m and path loss exponent of 2.4. Over this physical layer the HCT clustering was implemented as described in [4].

To investigate the proposed clustering in the HCT protocol, network scenarios composed by random topology networks with 20 up to 100 nodes, increasing 10 nodes at each step, were simulated. For each network size, 10 different networks were generated. The cycle was defined with 8 superframes containing 10 time-slots with $1ms$ each (including the beacons, which are control messages to start and finish a superframe), giving a cycle length of $80ms$; these values were assumed to be sufficient to clusterize these networks. For each network a series of simulations of 60 seconds (simulated time) were executed, to record the resulting clusters and the delay to establish each cluster.

A. Simulation results

The first experiments simulated the HCT in random topology networks with sizes ranging from 20 nodes up to 100 nodes. The figure 3 shows two examples of the obtained clusters; all the other simulations gave similar results. As it can be observed, clusters could be established automatically by


(a) Network with 20 nodes

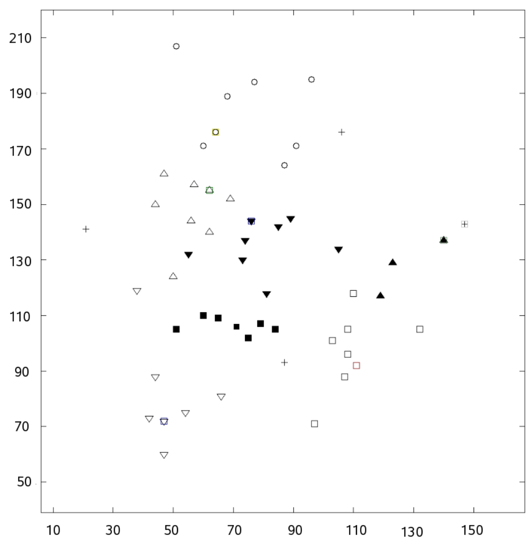
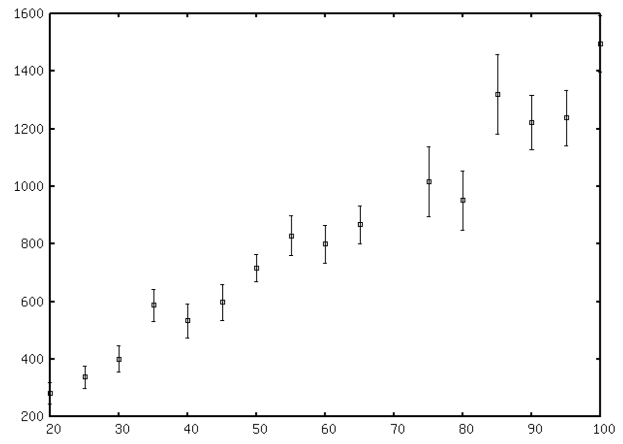

(b) Network with 100 nodes

Fig. 3. Resulting clusters for two example networks; coordinates in X and Y axis given in meters


(a) Clustering delays (in milliseconds)

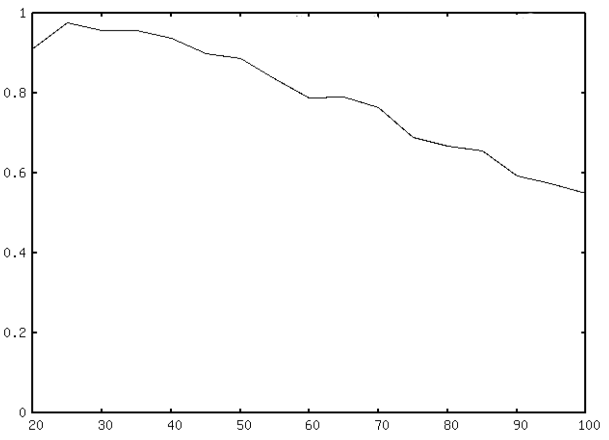

(b) Rate of clusterized nodes

Fig. 4. Clustering results as function of the network size in random networks

the HCT. They were formed with delays ranging from 0.28 seconds (network with 20 nodes) to 1.49 seconds (network with 100 nodes), as shown in figure 4(a). They were obtained averaging the clustering delays given by each simulation run; in all simulated scenarios, the standard deviation of the clustering delays (shown as error lines) were at most 10% of the mean value. It shows that these delays increase almost linearly with the network size, but at a low rate.

The clustering delays are important results to evaluate how long does it take to establish a cluster. A cluster is considered established when its cluster-head succeeds in allocating a superframe and its cluster has at least one member node. It is expected that the clustering delay is not very sensitive to the network size. In fact, we consider that the exchange of the neighborhood quality information allows the selection of good cluster-heads, which messages can be received with low error probability by their cluster members and also their neighbors, and vice-versa. But there is a visible linear relation between the clustering delays and the network size in these simulation experiments. Since the number of cycles a node waits before deciding to become a cluster-head was predefined, this evidences that, as the number of nodes increased, there were more failed clustering attempts.

Other important result is the rate of clusterized nodes. Specially in networks with sizes above 45 nodes, not all nodes became cluster members, as can be seen in figure 4(b). Since there are 64 available time-slots per cycle (there are 8 superframes, and a superframe has 10 time-slots, but 2 time-slots are used to transmit control frames), this limited the number of clusters in these small networks. This is due to the signal range and how HCT avoids interfering clusters, as described in [4].

To compare the HCT with the CSMA/CA, the random networks were also simulated using the CSMA/CA as the MAC protocol. To obtain results which can be compared to those previously derived for the HCT, it was defined that each node periodically broadcasts frames with period $80ms$,

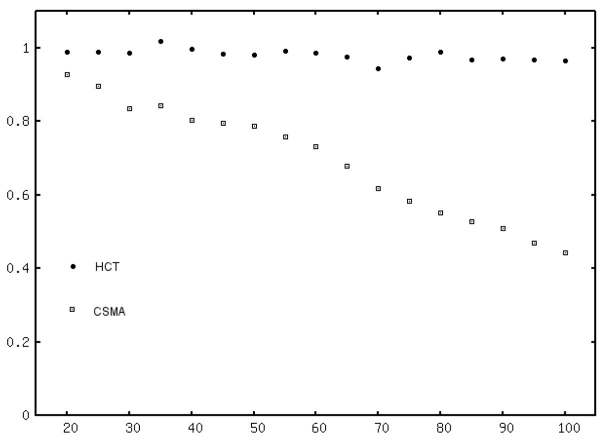
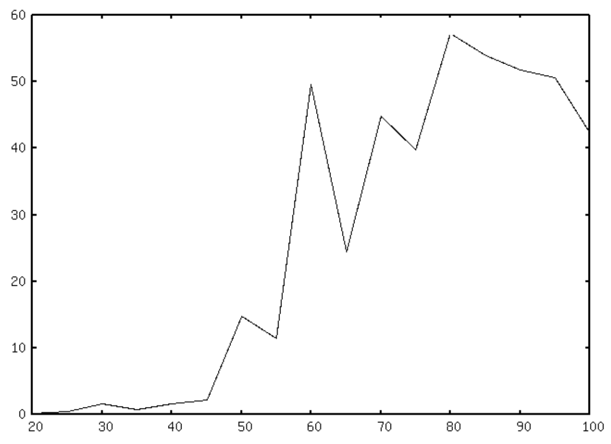


Fig. 5. Medium utilization for HCT and a CSMA/CA protocol as function of the network size

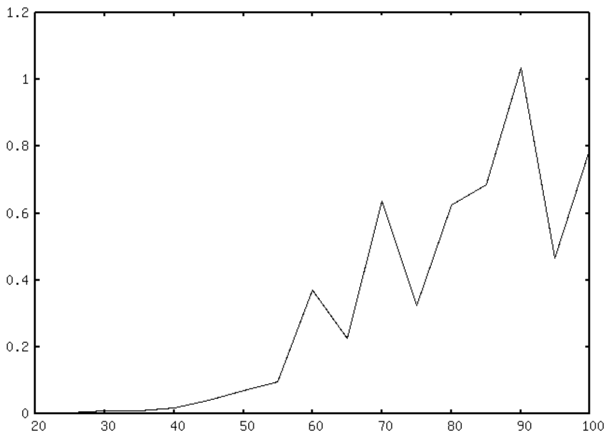
and with a random phase (time of the first transmission). Since the simulation time was $60 seconds$, each node could transmit at most 750 frames during the simulation. Each frame includes a timestamp corresponding to its transmission time. At the receiving side, each node records both the maximum and average transmission delay for the received frames, and the number of received frames. Using these values one can identify the network utilization and the transmission delays in the simulated networks.

The network utilization using both HCT and CSMA/CA is shown in figure 5. According to the performed simulations, the network utilization using the CSMA/CA decreases significantly, dropping from 700 average received frames, for networks with 20 nodes, to about 300 received frames, for networks with 100 nodes. For the same networks, the HCT was able to keep the number of received frames always above 700 frames, and there was no noticeable decrease in its network utilization when the network size increased. This result demonstrates the resource-reservation operation of the HCT, which avoids collisions of transmissions of clusterized nodes, and thus minimizes transmission errors. The CSMA/CA protocol, as a probabilistic MAC, relies only on carrier sense and random delays to try to avoid collisions, but this does not perform well under higher network loads or bigger networks.

The transmission delays presented by the CSMA/CA protocol were also collected and are shown in figure 6. The delays for the HCT are defined by the cycle length, since a clusterized node must wait at most one cycle to transmit a frame. Therefore, the transmission delay in the HCT is expected to be at most $80ms$ in the simulated scenarios. The CSMA/CA, however, presents a higher variability in the transmission delays, which increase significantly with the network size. This is as expected, because since more nodes compete for the medium, is more probable that they repeatedly find the medium busy, and then impose random delays before trying again. Because this happens more often, the frames take longer to be transmitted, increasing the overall transmission delay. However, the HCT does not suffer the same problem,



(a) Maximum transmission delays



(b) Average transmission delays

Fig. 6. Transmission delays (in seconds) for the CSMA/CA protocol as function of the network size

imposing always the same delay for the transmission of the clusterized nodes.

IV. Conclusions and Future Work

This paper presented a performance evaluation for the clustering in the HCT MAC, which is a hybrid MAC protocol for wireless ad-hoc networks with real-time capabilities previously proposed by the authors. Differently from other wireless networks protocols that use clustering for routing purposes, HCT relies on clustering to allow their nodes to operate in resource-reservation mode. It has a TDMA component that coordinate member nodes to transmit according to a dynamic schedule. As mobility can cause the network topology to change, and the nodes are not aware of each other, the clustering solution must be adaptive and self-organized. To investigate its adaptability, a range of small networks with random topologies were simulated to obtain corresponding clustering delays and number of clusterized nodes.

In the several performed simulations, suitable clustering was obtained for small networks with less than one hundred nodes. In these cases, clusters were established within less than two seconds, and contained nodes with higher link quality related to the cluster-head. As consequence, clusters could be established within a reasonable time, allowing clusterized nodes to operate in resource-reservation mode and better utilize the network. The clusterized nodes were bound to the number of available time-slots (64 time-slots), and in networks with more than 50 nodes a substantial number of nodes remained outside the clusters. This happened due to the density of the simulated networks, which did not allow cluster to share superframes - i.e. clusters that do not interfere at all were not formed.

To obtain better results for the clustering delays and clusterized nodes, further improvements in the protocol must be investigated, together with the tuning of its parameters according to the characteristics of each network. However, the established clusters allow the communications in resource-reservation mode, and therefore it is expected that this provides high medium utilization and bounded transmission delays. To investigate this issue it was performed a comparison between HCT and a simplified CSMA/CA protocol as provided by the Castalia framework. This CSMA/CA protocol operates similarly to the unslotted CSMA/CA in IEEE 802.15.4. This comparison has proof the stability of the HCT, while with the CSMA/CA the network utilization decreased significantly with the network size. Besides that, the transmission delays with the CSMA/CA highly increased with the network size, but these delays in the HCT are limited to the cycle length.

There exist a number of further questions regarding the efficiency of the proposed clustering approach. Firstly, it must be further investigated the scalability of the HCT protocol, related to the network size. Ideally, clusters must be formed quickly for any network size, in case the parameters *superframe size* and *cycle length* are appropriately defined. The choice of these parameters is another issue of the protocol, because it depends on the network topology. Therefore, a desired result is a generic definition of these parameters for networks with density below a known value.

V. ACKNOWLEDGMENTS

The authors would like to thanks the Brazilian Research agencies FAPESC, CAPES, and, in special, CNPq (Brazilian National Counsel of Technological and Scientific Development) for the grant 486250/2007-5.

REFERENCES

[1] J. Voelcker, “Cars get street smart,” *IEEE Spectrum*, vol. 44, no. 10, pp. 16–18, Oct. 2007.

[2] C. P. Bridges and T. Vladimirova, “Agent computing applications in distributed satellite systems,” in *9th International Symposium on Autonomous Decentralized Systems (ISADS 2009)*, Athens, Greece, 2009.

[3] M. M. Sobral and L. B. Becker, “A wireless hybrid contention/tdma-based mac for real-time mobile applications,” in *ACM Symposium on Applied Computing 2008, Real-Time Systems Track*, Fortaleza, Brazil, March 2008.

[4] ——, “Towards a clustering approach to support real-time communication in ad-hoc wireless networks,” in *Brazilian Workshop on Real-Time Systems 2009 (WTR 2009)*, Recife, Brazil, May 2009.

[5] M. Zuniga and B. Krishnamachari, “Analyzing the transitional region in low power wireless links,” in *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, 2004, pp. 517–526. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1381954

[6] A. Varga, “The omnet++ discrete event simulation system,” in *Proceedings of the European Simulation Multiconference*. Prague, Czech Republic: SCS – European Publishing House, June 2001, pp. 319–324.

[7] H. N. Pham, D. Pediaditakis, and A. Boulis, “From simulation to real deployments in wsn and back,” in *IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007*, June 2007, pp. 1–6.

Session 5

Chair: Zdenek Hanzalek

Rapporteur: Ramon Serna Oliver

Design and Development of a Reliable Ethernet-based Real-Time Communication Protocol

Greg Bollella
Sun Microsystems
Greg.Bollella@sun.com

Mike Duigou
Sun Microsystems
Mike.Duigou@sun.com

Ivan Tarasov
Sun Microsystems
Ivan.Tarasov@gmail.com

Abstract—**Ethernet-based networks are very popular for providing high throughput connections at a low cost. There's abundance of literature describing the ways to use Ethernet for real-time applications, including customization of the hardware, however the problems imposed by computer architecture and operating systems used to implement these real-time systems are often only glossed upon. This paper concentrates on challenges of design and development of a high performance Ethernet-based protocol with hard real-time constraints for an off-the-shelf x86 hardware running general-purpose operating system with real-time scheduling support.**

I. Introduction

The research done on the topic of Ethernet-based Real-Time communications is often concerned with the problems of dealing with collisions (by suppressing them, reducing their number, or resolving them deterministically), implementation of QoS routing in presence of switches, and scheduling real-time and best-effort traffic [1]. The questions of the design and implementation of the software stack on the end-points are often discussed only in terms of the protocol implementation.

In [2] an overview of the requirements on the general-purpose OS for real-time networking is provided, together with notable examples of general purpose operating systems which provide some real-time support. Still, the question of interactions of operating systems with hardware and effects of additional devices (such as extra network cards doing best-effort communication) on the predictibility of the system are not discussed.

This paper provides a case study of design and development of a hard real-time system using the off-the-shelf hardware and software, and the problems related to the hardware/OS choice. The designed system used point-to-point Ethernet connection which eliminated most of the "medium" problems (delays and packet loss at switches, effects of non-real-time traffic on the predictability).

Typically communication protocols are implemented as part of a kernel, often as a separate kernel module. The applications using the protocol are usually only provided with a simple interface that allows them to send and receive the data, most of the communication logic is hidden in kernel space. General-purpose protocols, such as TCP or UDP are often inadequate for specific requirements of the real-time systems, especially in cases where the protocol stacks are optimized for throughput rather than predictability.

The work described in this paper has been done as a part of building a prototype for a hard real-time control system. It was supposed to replace an old PLC-based control system with the one based on x86 systems running Solaris 10 with control and communication algorithms implemented in Java language using Real-Time for Java Specification [3] conforming implementation.

One of the reasons for replacing the PLC-based solution was its low flexibility, and high cost. Moreover, a computer-based solution can provide additional features, such as observability, configuration, and control over the network, which can further decrease the total cost of ownership.

The approach chosen in this project was to implement the smallest possible subset of operations in the kernel module and move the bulk of the protocol implementation into a user space library written in Real-Time Java, which can be used from an application. Main benefits of such an approach are decreased development time (due to the language choice, availability of data-structures implementations in the standard library, and ability to rely on real-time garbage collection), quicker development/deployment/testing cycle, and better system stability (since the kernel module is kept lean). It also allowed to improve the predictability (compared to TCP implementations), since most of the throughput-optimized kernel code paths causing predictability problems were bypassed.

II. Problem description

The control system consists of two machines connected to a single PROFIBUS [4] link which is used for observation and control of some device. One machine is a master which sends the actual control commands over the PROFIBUS link, while another one is a redundancy slave and is ready to take over the control should a problem on the master occur. Master and slave are interconnected using two direct 1000BASE-T Ethernet links, over which master needs to send up to 2 megabytes of data reliably every 100 ms, with a constraint that the transmission must not take more than 50 ms.

Reliability constraint means that even if a packet of data at the physical layer is lost, for example due to EMI-caused bit corruption, or network card malfunction, the protocol is able to correct that situation and ensure that all data is transferred.

Additional requirements include the ability to have some low-priority low-bandwidth network communications on the two other Ethernet NICs which are available on master and slave, e.g. for health monitoring and logging.

The solution was supposed to work on a specific hardware and software stack, which can be regarded as yet another important set of constraints.

A. Hardware and software specification

The following hardware was selected for master and slave:

- *CPU*: Pentium Celeron M, 1.4 GHz
- *Chipset*: Intel 855GME
- *Memory*: 2GB (DDR333/400)
- *Networking*: Four 1 Gbps Ethernet NICs on a single PCI bus

The system runs Solaris 10 update 4, with control program implemented in Sun Java Real-Time System 2.1. Solaris 10 has real-time scheduling priority group which is higher than the priority of kernel worker threads; that allows critical threads to preempt the system. Sun JRTS 2.1 is a Real-Time for Java Specification implementation which supports the necessary primitives for writing real-time applications, and also allows to use real-time garbage collection, which simplifies the protocol implementation code.

III. Feasibility analysis

Theoretical throughput of 1000BASE-T Ethernet link is $s = 10^9$ bits/second. Let m be MTU, and o be estimated protocol overhead in bytes (including the MAC header), then $e = m - o$ is effective payload in bytes. Ethernet frame overhead is $f = 8 + 4 + 12 = 24$ bytes (preamble, CRC32, interframe gap). Thus, the lower bound for transfer time of $d = 2 \cdot 2^{20}$ bytes of data can be estimated using the following formula:

$$t_{ideal}(m) = \frac{8 \cdot (m + f)\frac{d}{m-o}}{s}.$$

Higher MTU values correspond to lower transmission overhead, so usage of jumbo frames is beneficial. Substituting the values and taking the protocol overhead $o = 32$, MTU of $m = 512$ (the worst reasonable packet size) and $m = 16128$ (the largest packets supported by the NICs in use), we get

$$t_{ideal}(512) = \frac{8 \cdot (512 + 24)\frac{2 \cdot 2^{20}}{512-32}}{10^9} \approx 18.73456 \cdot 10^{-3}\, s.$$

$$t_{ideal}(16128) = \frac{8 \cdot (512 + 24)\frac{2 \cdot 2^{20}}{512-32}}{10^9} \approx 16.83559 \cdot 10^{-3}\, s.$$

These values are well within the requirements limits, the difference is quite substantial (almost 2 ms), however as we'll see below there are other more important factors compared to the overhead which affect performance because of the MTU changes.

IV. Solution approaches

Several approaches where tried for the solution prototype. Each solution was extensively tested with various kinds of loads on the system (lots of low-priority periodic tasks, memory allocation and garbage collection stress tests, network communications on the two NICs not used for real-time communications) running with a test program sending 2 Mb of data every 100 ms. The acceptance condition was the ability to run at least 120 hours with transmission times always less than 50 ms. Transmission times also included the time necessary to fill up the 2 Mb buffer with various kinds of data (doubles and integers).

KB	t_{ideal} (ms)	t_{min} (ms)	t_{avg} (ms)	t_{max} (ms)
50	0.41	1.1	1.2	1.6
100	0.83	1.9	2.1	3.4
300	2.48	4.9	5.6	6.8
500	4.12	8	9.3	10.2
1024	8.45	16.4	18.7	19.7
1536	12.67	25.1	28	29.4
2048	16.89	32.6	37	39

TABLE I
TCP test results for different transmitted data amounts

A. TCP-based solution

The simplest approach was to use TCP to send all the data. Without tuning the TCP configuration the average time was above 40 ms, and the maximum time was up to 900 ms. The problem appeared to be in priority inversion in the kernel:

The TCP stack in Solaris uses low-priority worker threads to do some clean-up, and in the case when it runs out of resources, it blocks until the worker threads do the clean-up. This works fine for general-purpose applications, because system threads have higher priorities than user space application threads, but in the case of a real-time application system threads may be preempted for a long time by real-time threads, which have higher priority. In our case the garbage collecting threads could preempt the worker threads for an extensive amount of time, which caused a priority inversion since the critical application threads which had priority higher than that of the garbage collector.

We were able to find these worker threads and increase their priority, so that non-critical threads couldn't preempt them, which fixed the outlier problem, however since the code-path from the NIC interrupt handler to the user application was too long, we couldn't guarantee that there are no more threads involved which could possibly cause another priority inversion.

Another problem with the TCP solution was that that the average and minimum transfer times were significantly higher than t_{ideal}. The results of the tests with different amounts of data being transmitted are shown in Table I

B. Ethernet-based custom protocol solutions

Another approach is implementation of a custom network protocol on top of Ethernet. There are several important things which were considered:

- the protocol should mitigate packet loss;
- bit-error rate of 1000BASE-T is very low (experiments showed loss due to corruption of $3.4 \cdot 10^{-8}$ of 8 KB packets, or $4.1 \cdot 10^{-11}$ BER);

- number of synchronous roundtrips (request-reply) should be minimized, since every roundtrip adds up to 1 ms to the transmission time.

Solaris allows user space applications to use DLPI (Data Link Provider Interface) to send and receive raw Ethernet packets. Initial tests showed that naive solution using DLPI which ignored the possibility of packet loss had mean transfer time of 2 megabytes of data around 18.5 ms with a very low variance.

However, the following problems were identified with DLPI solution:

- packets were dropped without packet buffering enabled in the kernel;
- buffering code caused priority inversion, because it used low-priority threads to handle timer events;
- priority inversion which prevented NIC interrupts from being handled for a long time.

DLPI can drop packets if the application can't keep up with the transmission rate. Without packet buffering in the kernel one system call resulted in a single packet, which caused problems when non-jumbo packets were used. To solve this problem, *bufmod* kernel module was used, which buffers the packets and returns multiple packets to user space in a single system call.

Bufmod uses timer facility of Solaris kernel which is based on the low-priority system threads running the callbacks. This causes a priority inversion which was solved by modifying the *bufmod* code to use real-time priorities for timer event handlers.

Another priority inversion which was uncovered was likely present in the TCP solution as well: DLPI is based on top of STREAMS implementation, and due to Solaris STREAMS design the interrupt handler can be occasionally blocked on a mutex in STREAMS code. The mutex may be held by some low priority worker threads, likely the ones transferring the packets between the packet queues.

Unfortunately it is hard to identify all the system threads which can possibly cause problems without an extensive knowledge of the Solaris kernel internals.

V. Custom kernel module-based solution

Both TCP and DLPI solutions have a common trait: the packets are processed by big parts of the Solaris kernel before they reach the user space application. However there is no real need for that: the only important code in the kernel is that which handles the interrupt and retrieves the data from the hardware, the code which resides in the NIC driver.

Whenever a packet (or a sequence of packets) is received by a NIC, the interrupt handler retrieves the packets and puts the data into preallocated data buffers, which it then passes to the MAC layer, calling one of the MAC layer functions. This function in turn calls all the registered handlers for the associated MAC device. In theory these handlers must return quickly, without blocking, because they are executed in the interrupt context and thus prevent all other interrupts at this interrupt priority level from being handled. This was not true for TCP and DLPI, however a custom kernel module could be designed which was guaranteed not to block in the packet handling code.

One way of implementing such a module is to have a ring-buffer to hold the received packets which have not been transferred into user space yet. Whenever the ring-buffer gets full, the packet receive handler must drop the packet instead of waiting for the user space process to receive some packets and thus free some space in the ring-buffer. In our case little more than 256 packets were usually sent during a single transmission, and the ring-buffer of 1024 elements was used. Realistically a much smaller ring-buffer (30-50 elements) could suffice.

To receive the packets from the ring-buffer, the application calls an *ioctl* on an opened virtual device file created by the kernel module, passing a buffer to receive multiple packets. The kernel module copies the available packets into the buffer and removes them from ring-buffer. If there are no packets in the ring-buffer, the *ioctl* returns immediately, blocks until a packet arrives, or blocks with timeout, depending on the argument passed by the application.

To send the packets, a different *ioctl* call is used to transfer the raw Ethernet packet data into kernel space and handle it off to the driver; no blocking is involved in this part.

A. Design

While it's possible to implement all the protocol logic inside the kernel module, it is not necessarily the best approach: development process of the code supposed to run in kernel context is much more complicated than that for the code running in user space. Since the user application was to be implemented in Java using Sun Java Real-Time System, it made sense to offload most of the protocol logic into a Java library which could interface with the driver to send and receive the data. The library in turn could provide a simple interface for the specific application, moreover it made possible the development of libraries which implement different protocols just using the *send* and *receive* ioctl calls described above.

A simplified protocol diagram is shown on Fig. 1.

One of the cases not covered by the protocol diagram is when an RUOK ("are you ok?") or ACK/NAK ("acknowledgement"/"negative acknowledgement") packet is lost. To mitigate that problem, before sending the first data packet the master sets a timer which first fires at the time when it expects to receive an ACK or NAK reply from the slave, and then fires periodically with a small period (for example 1 ms) until it is turned off. When the timer fires, its event handler resends the RUOK packet. When a NAK is received, the amount of data to be sent is estimated and the timer is restarted based on that estimate. When an ACK is received, the timer is stopped and the transfer is assumed completed.

Once the ACK packet is sent, the transfer is assumed to be completed on the slave and the received data becomes available for processing. If the ACK packet is lost, the master will try to resend an RUOK packet, but the slave has already

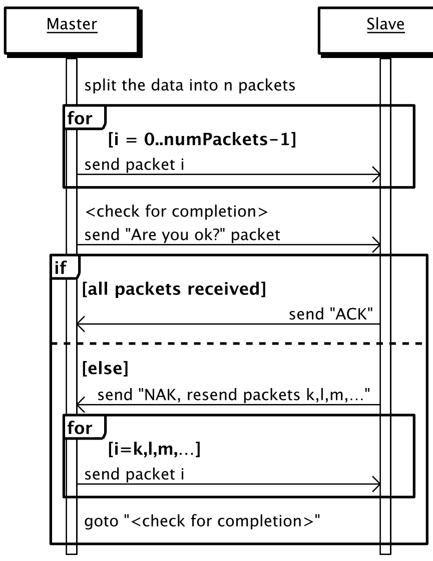


Fig. 1. Simplified protocol diagram

completed its receive procedure. That means that there is need to keep a thread on the slave which will answer all the RUOK packets during the time after the transfer completes and before the next one starts.

There are two distinct types of the packets: data packets (used to transfer the actual data) and control packets (RUOK, ACK, and NAK used for signalling). The amount of data packets is much greater than the amount of control packets, and the more packets there are, the more kernel space to user space context switches occur.

One of the ways to optimize this is to accumulate the data packets in a contiguous chunk of the kernel memory, and allow the application access this memory using memory mapping. This means that the driver becomes aware of the protocol and whenever it receives a data packet, it places the data into the right place in the buffer. Although the memory used by the buffer is shared between the kernel module and the application, the interrupt handler can't get blocked. The kernel module also keeps track of the sequence numbers of the packets it has received so far, so when the application on the slave receives an RUOK packet, it can check if the kernel module has received all the data packets it was supposed to receive. If it hasn't, a NAK packet is sent with a list of sequence numbers of the packets it hasn't received and the master sends these packets again.

An important part of that design is that control packets still may be added to the system without the need to add code to the kernel module.

B. Experienced problems

1) PCI bus contention: One of the problems which was noticed once we started running the tests with TCP network load on the other NICs was that although the threads producing the network load had low priority and were active only while

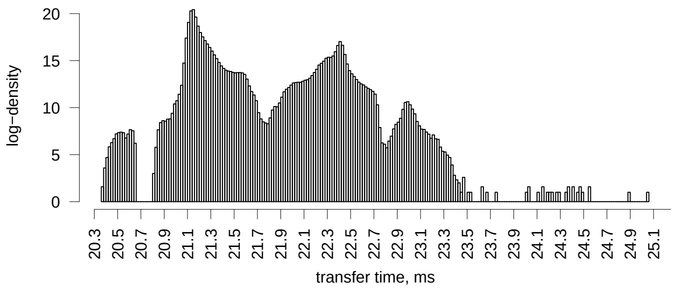


Fig. 2. Log-histogram of transfer time of 2 MB of data (in ms). The data was collected during a 180-hour run, at a 10 transfers per second rate.

the higher priority threads were not running, the transmission time went up by 8 ms. Thread switching diagram produced from the DTrace logs of the transmission didn't show any anomalies or evident priority inversions, so the PCI bus was suspected to be a bottleneck. And indeed, PCI bus throughput is 1067 Mbit/s, which is barely higher than 1 Gbit/s of a single network card. This means that the PCI bus could allow at most one NIC to be saturated and when the other NICs became involved, the bus arbiter started to share the bandwidth, thus increasing the critical transmission time significantly.

Since we didn't expect a lot of traffic on the other links, but also had to ensure that, we switched the two NICs connected to the external network into 10 Mbit/s mode, thus restricting the amount of extra data which could be transferred over the PCI bus during the time-critical phase. This proved to be effective and didn't produce any significant increase in transfer time compared to the transfer without the network load.

A better way could be to instruct the PCI bus arbiter to favor the specific NIC transfer requests, instead of using fair sharing. Unfortunately we were not able to find if that is possible.

2) Interrupt-caused priority inversion: Another problem related to the TCP network load was the priority inversion caused by the low-priority network load threads. These threads do not share any locks with the high-priority threads doing the real-time transfer, however the interrupts of all the NICs by default have the same interrupt priority level (IPL) and in Solaris a single thread per IPL is used to handle the interrupts. That means that the interrupt can't be handled until the previous interrupt handler from the same IPL has finished.

An old problem of an interrupt handler being blocked on a mutex which is held by a low-priority thread prevented the interrupts of the NIC doing the real-time transfer from being handled, which caused an unbounded priority inversion when any potentially long real-time task (like garbage collection) was running.

A solution to this problem was to explicitly set a higher IPL for the NIC doing the real-time transfer. This caused the interrupt to be handled by a different thread of a higher priority.

VI. Results and future work

Figure 2 shows the log-histogram of the transfer times using the described solution. Min/avg/max transfer times are

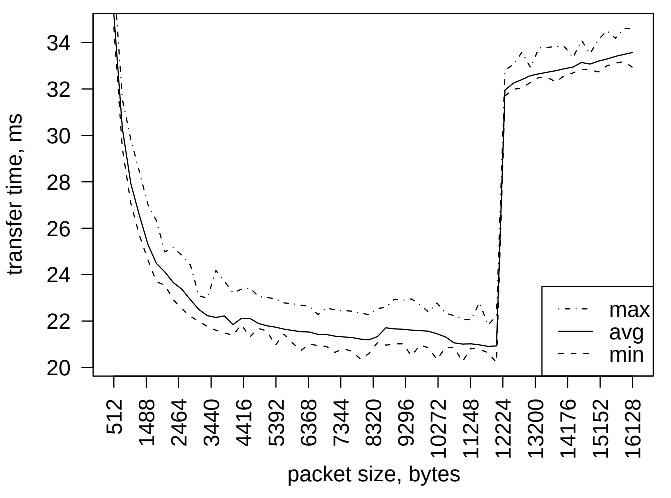


Fig. 3. Min/avg/max transmission times for different packet size values.

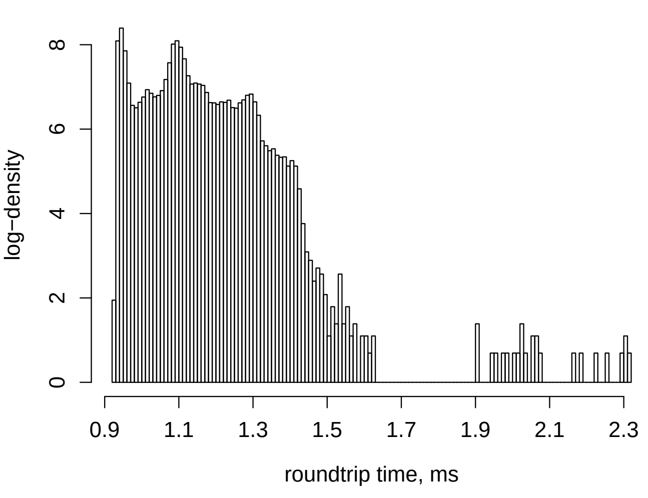


Fig. 4. Log-histogram of roundtrip times of small packets (ping test).

20.37/21.34/25.05 ms for the 2 MB block of data, with extra network load on the two NICs and with threads producing CPU and garbage collector load.

In this project two of four NICs are dedicated for the real-time data transfer, however only the protocol using a single link has been developed and tested. A protocol which can handle intermittent network failures on both links and complete failure of a single link is being designed and developed now. One interesting feature of such a protocol would be an improved transmission time when transferring the data on two links (if a bus with a better throughput is available, e.g. PCI Express).

Since the kernel module provides buffered receive and transmit capabilities for raw Ethernet packets, it is general enough to allow implementations of different protocols in a user space library. Applicability of that approach to protocols development where sub-millisecond latency is required should be investigated.

A. Optimal MTU selection

Figure 3 shows the relationship between the packet sizes used to transmit the data and the respective min/avg/max transmission times. As expected, the transmission time goes down with increase of the packet size, however at some point (at $MTU = 12244$) there is a sharp increase. The likely explanation for such behaviour is that the operating system has a pool of preallocated buffers for network packets, and there are different number of such buffers available for different sizes. The buffers of bigger size may be used for smaller packets, however if there are not enough big enough preallocated buffers for bigger packets, the packets must be split into smaller parts (Solaris represents network packets as a linked list of segements of data), which hurts the performance.

B. Minimal roundtrip time measurement

Another interesting measurement is the minimal roundtrip time for the smallest possible packet. Figure 4 shows the log-histogram of roundtrip times produced from about 55000 samples. The code sending the ping messages and replying to them was implemented in the user-space Java application. Before the measurement was done we expected to see the results distributed around a peak somewhere in hundreds of microseconds, not above 1 millisecond. This difference suggested that the delay may be introduced by the wait on the condition variable in the receive ioctl in the driver: the system was configured with a 1 millisecond timer “tick” resolution, which caused delays close to 500 microseconds during receive on both machines.

A better solution could use a semaphore to make the receiving thread block while waiting for packets, however this solution was not feasible on Solaris, because the semaphore$V$ function cannot be called from the interrupt context (thus it is impossible to increase the semaphore counter of available packets in the queue from the interrupt).

VII. Conclusion

Development of the solution for the requirements appeared to be much more complex than it appeared originally. Many of the difficulties were caused by the operating system, which wasn’t designed from the start to support real-time applications; nevertheless, it was possible to resolve the difficulties, once enough insight was obtained into the inner workings of the operating system. Hardware also proved to be a bottleneck in the project, however it was possible to work around the particular bottlenecks. Interestingly, Sun Java Real-Time System was never a part of any significant problem in the development.

One important conclusion which should be made is that all the parts of the real-time system should be analyzed on the lowest possible level, including software, operating system, computer hardware and the means of communication between the parts of the system. Without such an analysis and enough experience a solution may wind up being infeasible, or take more time to develop.

Real-time Java implementation was used previously for industrial robot control using Ethernet-based EtherCAT protocol [5], however the questions of predictability under high memory and network load were not analyzed. If it is possible to allocate a separate processor for the kernel worker threads, or otherwise guarantee that the kernel threads won't cause a priority inversion, it is possible to use simpler approaches, e.g. use DLPI to implement the protocol without the need for a custom kernel module, or use TCP/IP if it satisfies the timing requirements.

ACKNOWLEDGMENT

The work on this project is a team effort at Sun Microsystems. We would like to thank Frederic Parain, Roland Westerlin, Evgeny Yavits, Leonid Mesnik and Gail Yamanaka for their work on the project.

REFERENCES

[1] J.-D. Decotignie, "Ethernet-based real-time and industrial communications," in *Proceedings of the IEEE*, vol. 93, June 2005, pp. 1102–1117.

[2] K. Gopalan, "Real-time support in general purpose operating systems," 2001.

[3] G. Bollella, B. Brosgol, J. Gosling, P. Dibble, S. Furr, and M. Turnbull, "The real-time specification for java," 2000.

[4] *General Purpose Field Communication System*, vol. 2/3 (Profibus). CENELEC Std. EN 50170, 1996.

[5] S. G. Robertz, R. Henriksson, K. Nilsson, A. Blomdell, and I. Tarasov, "Using real-time java for industrial robot control," in *JTRES '07: Proceedings of the 5th international workshop on Java technologies for real-time and embedded systems*. New York, NY, USA: ACM, 2007, pp. 104–110.

Improving information throughput in CAN networks: Implementing the dual-speed approach

Sheikh Imran[1] **and Michael Short**[1]

[1]Joint lead authors: Embedded Systems Laboratory, University of Leicester,
University Road, Leicester, LE1 7RH.
si52@le.ac.uk, mjs61@le.ac.uk

Keywords: High-Speed CAN, Overclocking, Soft-Core.

Abstract

The CAN protocol was originally introduced for distributed automotive applications in the 1980s, and is still widely used by the industry today. CAN provides the basis for many cost-effective distributed systems, and since its first inception has proved to be a continuous area of research. Despite the fact that CAN possess many appealing features, the protocol also suffers from several significant drawbacks. One major drawback that can severely limit the applicability of CAN in data-intensive real-time applications is related to the wired-AND nature of the physical layer; this can act to severely limit the maximum transmission speed and bus length of a given CAN network. This paper is concerned with overclocking a CAN network to improve the information throughput, whilst simultaneously maintaining the priority-driven arbitration mechanism. The paper will describe the implementation of a modified CAN-controller, the basic design of which is based on a soft-core controller previously developed by the authors. The effectiveness of this prototype controller is illustrated in a series of experiments, in which the data transmission rate is dynamically increased to 2 Mbps during transmission of the data and CRC fields; this allows an almost doubling of information throughput with no penalty in transmission time. The paper is then concluded by suggesting possible areas of further research and improvement.

1. Introduction

Controller Area Network [1] is one of the most widely employed protocols for creating distributed embedded systems, with applications as far ranging as vehicle electronics, process control and many other important industrial applications. Some of the key features of CAN that have led to its widespread use include low overheads, non-destructive message arbitration, low message latency and good error detecting abilities; all features which are required for control applications running on embedded processors.

Traditionally, the use of CAN was mostly restricted to non-critical, event-triggered and soft real time applications [2], for example vehicle body electronics and industrial applications in which allocation of message priorities and bandwidth is a trivial requirement. This is perhaps due to the fact that the CAN protocol suffers from several significant drawbacks w.r.t. hard real-time systems. These drawbacks include redundancy issues, atomic broadcast problems, lack of protection from babbling idiot failures and information throughput restrictions; although most of these issues have been previously dealt with in the literature (e.g. see [3][4][5]).

Previous research has also focused on adapting CPU scheduling and feasibility analysis techniques to prove that timely delivery of messages will take place (e.g. see [6][7][8]). In conjunction, these techniques would seem to indicate that in many situations, CAN may be considered a viable choice in critical hard real-time applications such as drive-by-wire and aircraft engine control. However, with the evolution of mechatronic technology, in many situations the increased complexity and distribution of the control system requirements will dictate a need for high-bandwidth networks, capable of delivering information throughput at a level that is not achievable with CAN [2][3].

The CAN protocol employs a unique non-destructive priority-based arbitration scheme; when multiple nodes attempt to transmit messages simultaneously, a priority-based non-destructive mechanism is employed to ensure the highest priority message gains bus access. The wired-AND nature of the physical layer that is used to achieve this arbitration requires that all nodes in the network achieve a logical consensus on the instantaneous bit-patterns appearing on the bus lines; it is this requirement of the protocol that it acts to severely limit both the maximum transmission speed and bus length of a given CAN network. The maximum transmission rate is inversely proportional to the length of the bus, and has an upper limit of 1 MBit/s; due to its design a CAN frame may carry up to a maximum of 8 data bytes.

This paper is concerned with improving the performance and throughput of CAN networks by overclocking, whilst additionally increasing the available maximum data payload. The technique that will be implemented in this paper operates in such a way that the arbitration mechanism remains operational; this pilot implementation allows the transmission of frames with 15-byte data payloads without any additional increase in message transmission times. The modified protocol is implemented using a soft-core FPGA controller, based on a standard CAN controller that has previously been developed by the authors, and shown to be fully conformant to the CAN protocol [9][10]. The remainder of the paper is organised as follows. Section 2 describes previous work in the area of overclocking CAN networks, and describes the specific technique to be employed in this paper. The design of the modified physical layer, along with several practical considerations, is described in Section 3. Section 4 describes several initial experiments that have been performed to evaluate the operation and efficiency of the controller. The paper is concluded in Section 5, which also suggests areas of future work.

2. Overclocking in CAN Networks

Overclocking in CAN networks is a technique that may be used to artificially decrease the bit-time, and hence the transmission time, of a message. As illustrated in figure 1, each CAN controller in a given network is driven by a local clock source (typically a crystal oscillator circuit). This clock signal is typically stepped down by an internal prescaler register, before being used to drive - via the bit timing registers - the protocol controller and bit stream processor. As such, overclocking most often refers to the deliberate use of a faster oscillator - in combination with appropriate settings in the prescaler and bit timing registers - to increase the transmission rate.

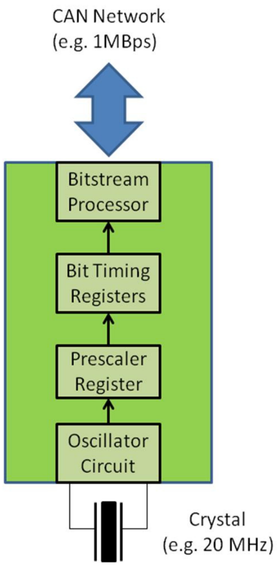


Figure 1 – Typical setup of a CAN controller oscillator.

In a time-triggered implementation of a CAN-based system, the overclocking problem does not pose many serious problems, save perhaps for certification issues. This is because when an appropriate higher-level protocol is employed, it is possible to statically design a message schedule such that no run-time message collisions – and hence message arbitrations – actually take place. An example of such a higher-level protocol is the shared-clock family of algorithms, described by Ayavoo et al [11]; with appropriate modifications, such algorithms are also suitable for use in redundant networks and may even be implemented in hardware [3][12].

Since no run-time collisions take place, the arbitration mechanism becomes redundant and there is no need for system-wide consensus as only a single node is in control of the bus; if an adequate technique is employed to deal with the ACK slot (e.g. by using local message acknowledgment) the CAN controllers may be overclocked by a significant factor, resulting in a proportional increase in data throughput. For event-driven and hybrid systems, however, the arbitration mechanism must be heavily relied upon, and simply overclocking the CAN controllers is not guaranteed to work; the requirement for node-wide temporal consensus of bit values is violated by signal propagation delays. One possible way around this problem has previously been suggested by Cena & Valenzano [13], who suggest using a dual-channel approach with one 'forward' channel and one 'reverse' channel. With appropriate modifications to the physical and datalink layers, they have suggested that such an approach would theoretically allow a bit-rate of up to 16 MBit/s.

However, this paper is concerned with another (simpler) technique that was first outlined by Cena & Valenzano in a short communication [14]. The technique is based upon the observation that a CAN frame may be divided into different zones, as depicted in figure 2. The M-zones of the frame are the portions in which multiple writers are allowed (i.e. during arbitration, acknowledgement, EOF sequence, and inter-message space). The S-zones of the frame are those portions in which only a single-writer is allowed (i.e. transmitting the DLC, payload and CRC). During the S-zone, only a single node has 'won' the arbitration and - save for error signalling - is in total control of the bus. As such, the CAN network may be operated at its normal transmission rate during the M-zones, ensuring temporal bit-wise consensus across the network; when the S-zone is entered, the CAN network may be overclocked to decrease the transmission time of the payload.

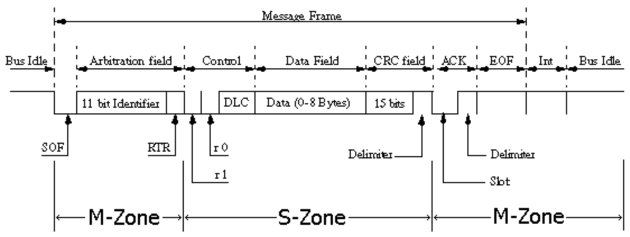


Figure 2 – CAN frame format showing S and M zones.

To the knowledge of the current authors, this technique - despite its simplicity - does not seem to have been experimentally evaluated in any previous works. This may be because it requires changes at the physical layer of the protocol; such changes have traditionally been difficult to implement in silicone. However, with recent advances in the development of programmable logic devices (PLD's) such as FPGA's, it is now possible to implement the required changes with (relative) ease. To this end, this paper seeks to implement - and also begin to experimentally evaluate - the dual rate approach. In this initial implementation, the M-zones are clocked at a speed of 1Mbps, whilst the S-zone is clocked at 2 Mbps; additionally, the available payload size is increased to 15-bytes, almost doubling the available information throughput rate for a given message transmission time when compared with standard CAN.

3. Modified CAN Controller Development

The authors of the current paper have previously developed a soft core CAN controller, primarily to allow extensions and modifications to the basic CAN protocol to be explored [12]. A test bed has been created around this controller, the general structure of which is shown in figure 3. For clarity the figure shows a system employing two CAN nodes, but the test bed may be extended as required to encompass an arbitrary amount of nodes. In the current study, the system was primarily based around four CAN nodes running the modified CAN protocol.

The Xilinx FPGA contains the implementation of the CAN controller, with modified features to work on dual rate. The soft core is implemented in Verilog [16], and has previously been verified as fully CAN-conformant [9][10]. The next subsections discuss some of the issues that were encountered when configuring the controller to support the dual-rate approach, and illustrate the basic transmission and reception of CAN frames. The figures that are displayed in this (and subsequent) Sections were taken with the use of the Chipscope tool, which can capture a large number of internal FPGA signals in real-time; thus allowing verification of the state of the core [17]. An ARM7 processor (LPC2129 from NXP) is used as the host controller for the CAN interface, and configures the CAN controller before use and subsequently handles transmission and reception processing of filtered message.

3.1. Issues setting up the test bench

A number of features were required to be added to the controller during the setting up of the test bench to support the dual rate approach; these are now described in detail, along with the issues that were encountered.

1. In order to implement dual-rate CAN, the modified controller has an additional Dynamic Bit Timing Register (DBTR) added into its structure. This register contains the settings to be used when adapting to the higher data rate (for this study the rate employed was 2 Mbps) during transmission of S-zones.
2. In order to enable effective switching, a state machine was used to generate a signal which is asserted only for the duration of the S-zone. During this time the DBTR setting are employed instead of the standard BTR.
3. In order to handle error signalling during the S-zone (e.g. detected bit or stuff errors), the state machine was set to de-assert the switching signal in case of a detected error or error frame; the only acceptable behaviour in the network is for the controllers to dynamically resume to the lower data rate, as the entire message – including M-zones – is required to be retransmitted in this case.
4. The foremost issue that was encountered while developing this modified CAN controller was whether the existing CAN transceiver interface can reliably support data rates higher than 1 Mbps; experimentally, messages were successfully transmitted and received in the 4-node network with a DBTR configuration of 2 Mbps, using commercially-available transceivers [19].

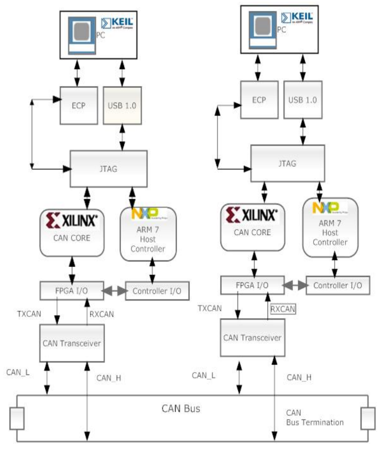
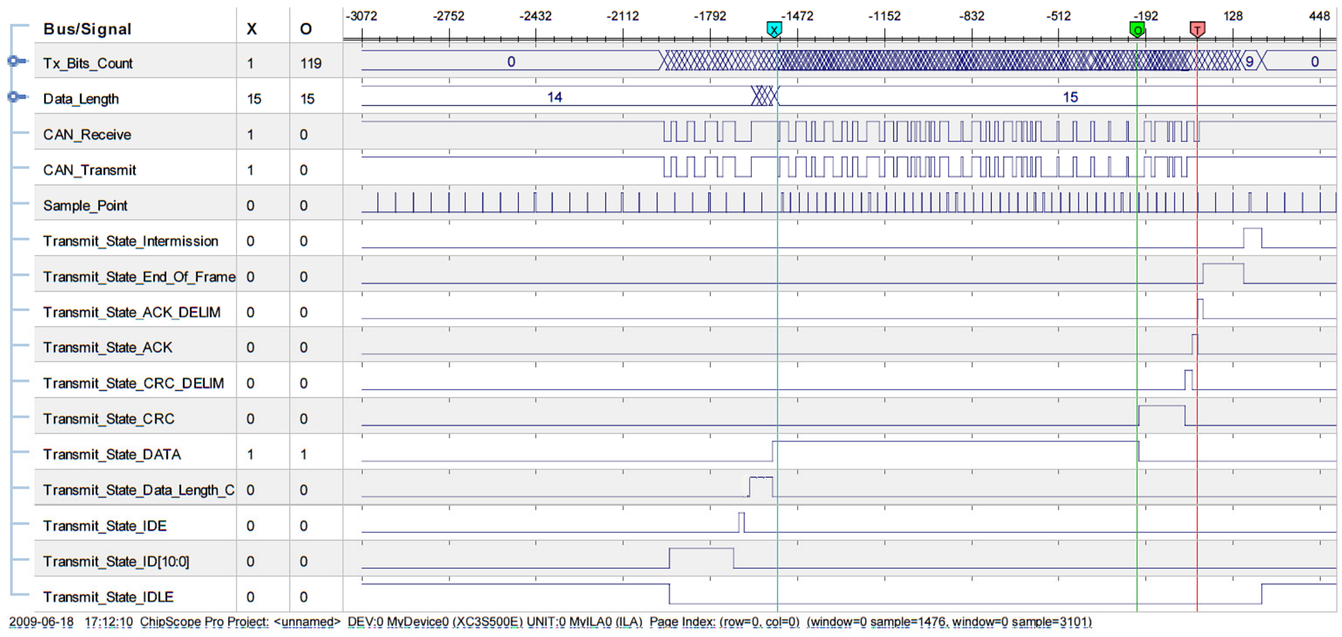


Figure 3: Overview of the test bed.

3.2. Basic transmission and reception of messages

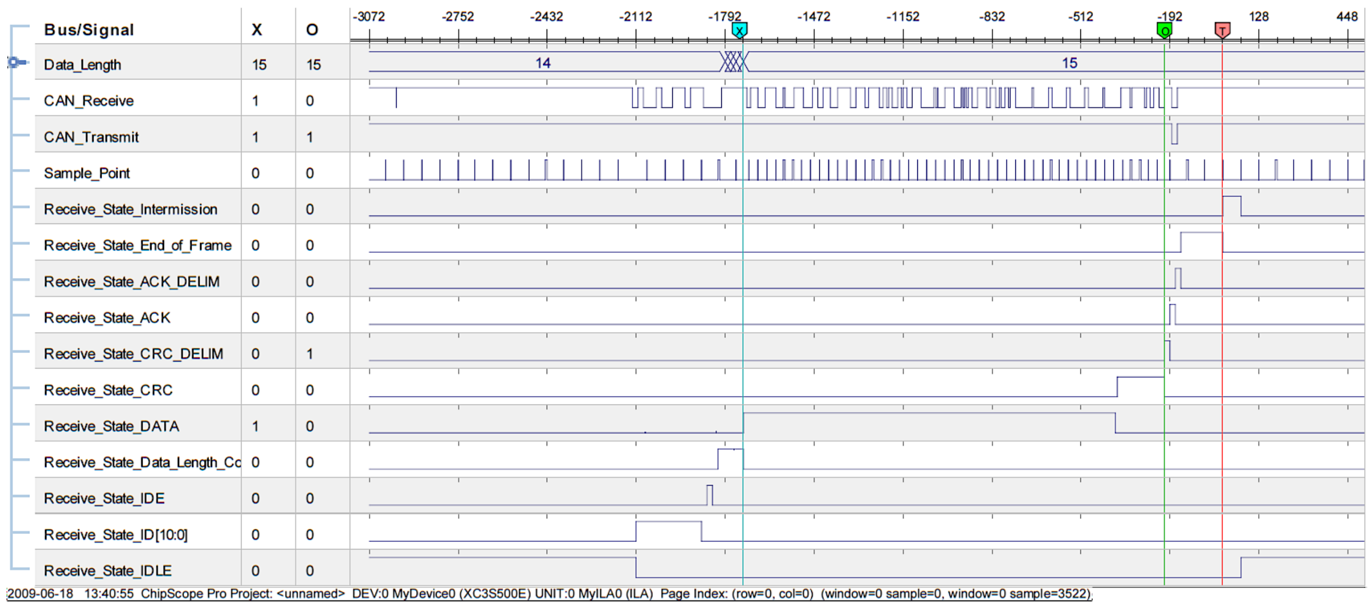
This Section will briefly describe the behaviour of the transmitter and receiver nodes using Chipscope snapshots; as mentioned, for this experiment the M-zone data rate is 1 Mbps and the higher data rate S-zone was 2 Mbps. Figure 4 depicts the transmitter behaviour during frame transmission. From the figure, the message transmission on the CAN bus can be observed via the CAN_Receive and CAN_Transmit signals showing the bits being sent and received simultaneously. Additionally, for clarity the figure illustrates several important transmit states which correspond to fields of the CAN data layer [1]. The sample point signal is critical when switching between data rates; figure 4 shows the assertion of the switching signal (Transmit_State_DATA) once the S-zone field has been entered, and the sample point can be seen occurring with twice the frequently during the Transmission_State_DATA and Transmission_State_CRC. It can also be observed that the field Tx_bits_Count between Marker ‘X’ and ‘O’ increases from 1 to 119, and the Data_Length field shows a 15 byte data payload in the message.

This behaviour can also be observed in figure 5, which is a snapshot taken on one of the participating receiver nodes. When the S-zone is exited, the switching signal is de-asserted and the sampling frequency reverts back to the lower rate. The successful transmission and reception of the message is demonstrated by the various transmit and receive states illustrated in the figures; in figure 5, an acknowledgement signal is sent by the receiver node as demonstrated by Receive_Ack_Delimiter field, and the CAN_Transmit signal becomes dominant.



2009-06-18 17:12:10 ChipScope Pro Project: <unnamed> DEV:0 MyDevice0 (XC3S500E) UNIT:0 MyILA0 (ILA) Page Index: (row=0, col=0) (window=0 sample=1476, window=0 sample=3101)

Figure 4: Dual rate implementation at transmitter node.



2009-06-18 13:40:55 ChipScope Pro Project: <unnamed> DEV:0 MyDevice0 (XC3S500E) UNIT:0 MyILA0 (ILA) Page Index: (row=0, col=0) (window=0 sample=0, window=0 sample=3522)

Figure 5: Dual rate implementation at receiver node.

4. Case Studies

To further verify the behaviour of this modified CAN controller, two test cases were specifically created to illustrate the operation of two of the main features of the protocol; the arbitration mechanism and error signalling. These tests are discussed and documented in the following sub-sections.

4.1. Arbitration

Figure 6 shows a snapshot of a node competing for the access of the CAN bus, whilst another node has a higher-priority pending message. At this point, the arbitration scheme should ensure that the higher-priority message wins bus access, and be transmitted first. Of particular note are the following observations:

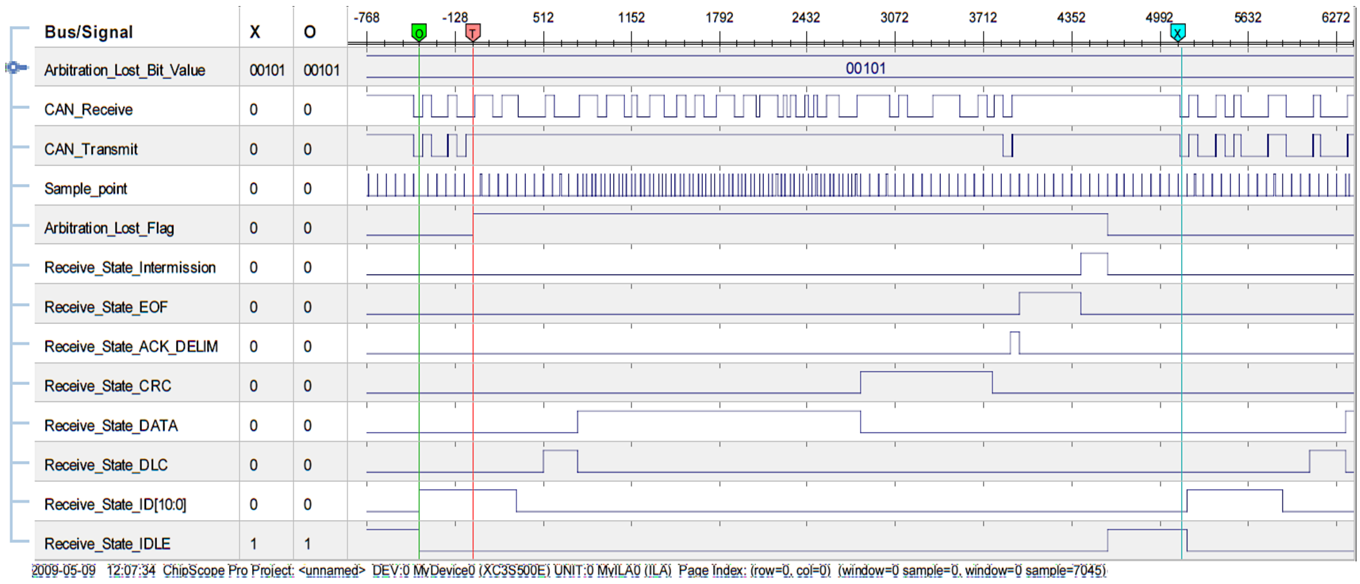

Bus/Signal
X
O
-768
-128
512
1152
1792
2432
3072
3712
4352
4992
5632
6272
Arbitration_Lost_Bit_Value 00101 00101
00101
CAN_Receive 0 0
CAN_Transmit 0 0
Sample_point 0 0
Arbitration_Lost_Flag 0 0
Receive_State_Intermission 0 0
Receive_State_EOF 0 0
Receive_State_ACK_DELIM 0 0
Receive_State_CRC 0 0
Receive_State_DATA 0 0
Receive_State_DLC 0 0
Receive_State_ID[10:0] 0 0
Receive_State_IDLE 1 1
2009-05-09 12:07:34 ChipScope Pro Project: <unnamed> DEV:0 MyDevice0 (XC3S500E) UNIT:0 MyILA0 (ILA) Page Index: (row=0, col=0) (window=0 sample=0, window=0 sample=7045)

Figure 6: Arbitration with switch back to normal receiver mode.

1. Marker ‘O’ shows the Start of Frame point, while the bit values between marker ‘O’ and ‘T’ demonstrate the transmission of identifier field; also shown by high state of signal Receive_State_ID [10:0].
2. Marker ‘T’ shows the CAN_Transmit signal changing to recessive state for the rest of message fields showing that the node has lost arbitration and has turned into a receiver.
3. The value on the Arbitration_Lost_Bit_value shows a value’00101’ which means that the arbitration round was lost on the 5th bit of the identifier.
4. The switch over during S-zone can be seen, as with the successful reception of the message; CAN_Transmit is pulled dominant at Receive_State_ACK_Delimiter field.
5. After Marker ‘X’, the pending message previously losing arbitration is now transmitted, as demonstrated by the CAN transmit/receive values and state registers.

4.2. Error Signalling

The second test case is related to error signalling, and demonstrates that in case of an error while data is being transmitted on the higher data speed, the controller will dynamically switch back to its normal speed. A bit error was generated on the CAN bus by corrupting a dominant bit to a recessive bit, to deliberately induce an error frame (the error was injected using an additional ARM7 development board connected via transceivers with the CAN bus [9]). Figure 7 shows a transmitter snapshot, and the following observations can be made:

1. Marker ‘X’ demonstrates the start of a normal transmission, illustrating the different CAN fields during the initial phase of transmission.
2. At marker ‘T’ the CAN_Transmit signal is dominant, whilst the CAN_Receive signal is made recessive by the injection of the bit error.
3. Before marker ‘T’ on the Transmit_State_Data, the increased sample point frequency can be seen; as soon as the bit error occurs, and the start of the error signal begins, the sample rate switches to its normal rate.
4. The Error_Frame_Flag field (immediately after marker ‘T’) shows the transmission of the error frame at the lower M-zone data rate. Also observe that the transmit error counter has increased from ‘143’ to ‘151’ as per the protocol specification [1].

These two test cases demonstrate that the modified CAN controller, with enhanced functionality, can maintain conformance with the arbitration scheme and error handling requirements mandated by the CAN protocol standard. It should also be noted at this point that many additional tests have also been successfully performed with the modified CAN controller, to cover all corner cases; the full documentation of these tests is beyond the scope of the current paper.

5. Discussion and Conclusion

This paper has considered an implementation of a dual-transmission rate CAN controller. The modified controller has been shown capable of transmitting CAN frames with an increased data payload size, and at a higher data rate than the protocol normally allows. Although initial experiments with the technique indicate its effectiveness, it does have several drawbacks; it seems unlikely that such a scheme can be made backwardly-compatible with existing CAN implementations. When employed in a mixed system with one or more standard CAN controllers, numerous errors (most notably bit stuffing and CRC errors) are observed and signalled.

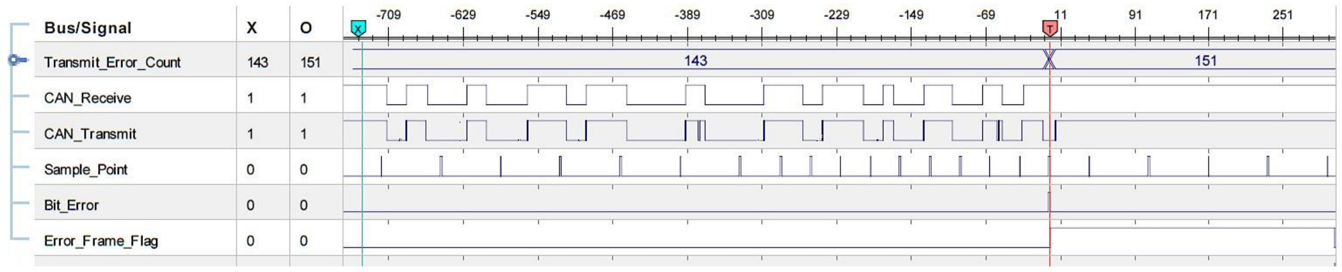


Figure 7: Error signalling in dual rate CAN.

As such, it would seem that this scheme is perhaps best suited to new CAN implementations which require a high rate of information throughput. An alternate application may be in situations in which the message set characteristics are found to be infeasible - from a scheduling perspective - when using a standard CAN network (using techniques described in [6],[7] and [8] for example). In these situations, overclocking may be applied to one or more of the messages to decrease transmission times, and hence improve worst-case response times to an appropriate level. The use of the method in such a way is an area of future investigation. Other potential areas of future work include examining the impact of the dual-rate technique on bit-error susceptibility in noisy environments, investigating the maximum achievable bit rate when using standard CAN transceivers, and exploring the possibility that nodes may dynamically request a particular data rate for S-zone transmission as part of a message itself.

In conclusion, with appropriate care in the implementation stages, this modification to the CAN protocol seems to indeed be feasible, and may have the capability to further increase the applicability of this popular protocol; albeit only in several specialised applications or situations.

Acknowledgements

The work in this paper is supported by the award of a PhD scholarship to Sheikh Imran from the "NWFP University of Engineering & Technology, Peshawar, Pakistan through the Higher Education Commission of Pakistan".

References

[1] R. Bosch, "CAN Specification 2.0", Postfach, Stuttgart, Germany: Robert Bosch GmbH, 1991.

[2] H. Kopetz, "A Comparison of CAN and TTP", *Annual Reviews in Control*, Vol. 24, pp. 177–188, 2000.

[3] M. Short and M.J. Pont, "Fault-Tolerant Time-Triggered Communication Using CAN", *IEEE Transactions on Industrial Informatics*, Vol. 3, No. 2, 2007.

[4] I. Broster, and A. Burns, "An analyzable bus-guardian for event triggered communication", in *Proc. of the 24th IEEE Real-Time Systems Symposium*, pp. 410-419, December 2003.

[5] J. Rufino, P. Verissimo, G. Arroz, C. Almeida and L. Rodrigues, "Fault-Tolerant Broadcasts in CAN", in *Proc. of the 28th Fault-Tolerant Computing Symposium (FTCS)*, pp. 150–159, 1998.

[6] I. Broster and A. Burns, "Timely use of the CAN protocol in critical hard real-time systems with faults", In Proceedings of the *13th Euromicro Conference on Real-time Systems (ECRTS)*, Delft, The Netherlands, June 2001.

[7] P. Pedreiras and L. Almeida. "EDF message scheduling on controller area network", *Computing and Control Engineering Journal*, pp. 163-170, August 2002.

[8] K. Tindell, A. Burns and A.J. Wellings, "Calculating controller area network (CAN) message response times", *Control Engineering Practice*, Vol. 3, No. 8, pp. 1163-1169, 1995.

[9] I. Sheikh, and M. Short, "A low-cost and flexible approach to CAN conformance testing", Paper to be presented at ICINCO 2009, Milan, Italy, July 2009.

[10] I. Sheikh, M. short, and K.F. Athaide, "Using Virtual I/O for CAN Bit timing Conformance Tests", Paper to be presented at ICIE 2009, London, UK, July 2009.

[11] D. Ayavoo, M.J. Pont, M. Short, and S. Parker, "Two novel shared-clock scheduling algorithms for use with CAN-based distributed systems", *Microprocessors and Microsystems*, Vol. 31, No. 5, pp. 326-334, 2007.

[12] S. Imran, M. Short and M.J. Pont, "Hardware implementation of a shared-clock scheduling protocol for CAN: A pilot study", in *Proceedings of the 4th UK Embedded Forum*, Southampton, UK, September 2008.

[13] G. Cena and A. Velanzano, "FastCAN: A High-Performance Enhanced CAN-Like Network", *IEEE Transactions on Industrial Electronics*, Vol. 47 No. 4, pp. 951-963, 2000.

[14] G. Cena and A. Valenzano, "Overclocking of controller area networks", *Electronics Letters*, Vol. 35, No. 22, pp. 1923-1925, October 1999.

[15] L.M. Pinho and F. Vasques, "Reliable real-time communication in CAN Networks", *IEEE Transactions on Computers*, Vol. 52, No. 12, pp. 594–1607, 2003.

[16] IEEE, Standard for Verilog Hardware Description Language, *IEEE standard 1364*, 2001.

[17] Chipscope Pro. Web link: http://www.xilinx.com/ise/optional_prod/cspro.htm

[18] SJA1000 standalone CAN Controller from NXP semiconductors. Web link: http://www.nxp.com/acrobat_download/datasheets/SJA1000_3.pdf

[19] PCA82c250 CAN Controller Network interface, from NXP Semiconductors. Web link: http://www.nxp.com/acrobat/datasheets/PCA82C250_5.pdf