Optimization of TTEthernet Networks to Support Best-Effort Traffic

Tamas-Selicean, Domitian; Pop, Paul

Published in:
Proceedings of the 19th IEEE International conference on Emerging Technology and Factory Automation (ETFA 2014)

Link to article, DOI:
10.1109/ETFA.2014.7005256

Publication date:
2014

Optimization of TTEthernet Networks to Support Best-Effort Traffic

Domiţian Tămaş–Selicean and Paul Pop
DTU Compute Technical University of Denmark
Kongens Lyngby, 2800 Denmark
Email: dota@dtu.dk, paupo@dtu.dk

***Abstract*—This paper focuses on the optimization of the TTEthernet communication protocol, which offers three traffic classes: time-triggered (TT), sent according to static schedules, rate-constrained (RC) that has bounded end-to-end latency, and best-effort (BE), the classic Ethernet traffic, with no timing guarantees. In our earlier work we have proposed an optimization approach named DOTTS that performs the routing, scheduling and packing / fragmenting of TT and RC messages, such that the TT and RC traffic is schedulable. Although backwards compatibility with classic Ethernet networks is one of TTEthernet's strong points, there is little research on this topic. However, in this paper, we extend our DOTTS optimization approach to optimize TTEthernet networks, such that not only the TT and RC messages are schedulable, but we also maximize the available bandwidth for BE messages. The proposed optimization has been evaluated on a space application case study.**

I. Introduction

The increase in functionality that is implemented as real-time embedded applications, most often on distributed architectures, has resulted also in increased bandwidth requirements. A large number of communication protocols have been proposed for embedded systems. Although Ethernet [6] is not suitable for real-time and safety-critical applications [3], many Ethernet-based communication solutions have been proposed in recent years, due to the low costs and high speeds of Ethernet (up to 10 Gbps). Some examples are ARINC 664 Specification Part 7 (ARINC 664p7, for short) [1], TTEthernet [9], EtherCAT [4] and IEEE Audio Video Bridging (AVB). [10] compares several proposed Ethernet-based real-time communication protocols.

TTEthernet [9] is a deterministic, synchronized and congestion-free network protocol based on the IEEE 802.3 Ethernet [6] standard and compliant with the ARINC 664p7specification [1]. ARINC 664p7 is a full-duplex Ethernet network, which emulates point-to-point connectivity over the network by defining *virtual links*, tree structures with one sender and one or several receivers (see Section II). TTEthernet supports applications with mixed-criticality requirements in the temporal domain, providing three types of traffic: static time-triggered (TT) traffic and dynamic traffic, which is further subdivided into Rate Constrained (RC) traffic that has bounded end-to-end latencies, and Best-Effort (BE) traffic, for which no timing guarantees are provided. TT messages are transmitted based on static schedule tables and have the highest priority. RC messages are transmitted if there are no TT messages in transmission, and BE traffic has the lowest priority. TTEthernet is highly suitable for applications of different safety criticality levels, as it offers spatial separation for mixed-criticality messages through the concept of virtual links. TTEthernet is suitable for automotive [12], avionics [15] and space [5] applications.

Several researchers have started to address the analysis and optimization of TTEthernet. For full-duplex switched Ethernet networks with priority operations, Schneider et al. [11] have proposed a compositional timing analysis based on real-time calculus [2]. For ARINC 664p7 systems, researchers [7] have proposed a new real-time switching algorithm that guarantees an upper bound on the switching period. Having such an upper bound simplifies the worst-case delay analysis. For TTEthernet, Steiner [13] proposes an approach for the synthesis of static TT schedules, where he ignored the RC traffic and used a Satisfiability Modulo Theory (SMT)-solver to find a solution which satisfies an imposed set of constraints. The same author has proposed an SMT-solver approach to introduce periodic evenly-spaced slots into the static schedules to help reduce RC delays in [14]. Suethanuwong [16] proposes a scheduling approach of the TT traffic, ignoring RC traffic, that introduces equally distributed available time slots for BE traffic.

In this paper we are interested in safety-critical real-time applications implemented on heterogeneous processing elements interconnected using TTEthernet. In [17] we have proposed an approach that focuses only on deriving the static schedules for the TT messages. In [18] we have proposed the DOTTS ("Design Optimization of TTEthernet-based Systems") strategy, a Tabu Search-based metaheuristic that optimizes the TTEthernet network implementation, such that the TT and RC messages are schedulable, and the end-to-end delay of RC messages is minimized. DOTTS optimizes: the routing of virtual links (VLs), the packing and fragmenting of messages into frames, the assignment of frames to VLs, the bandwidth of the RC VLs and the schedules for the TT messages.

DOTTS focuses only on TT and RC traffic. However, one of the strong points of TTEthernet is the backwards compatibility with Ethernet (the BE traffic). In practice, the BE traffic is used for non-critical applications, without timing or safety constraints, and for legacy components, which do not support TT or RC communications. Hence, it is imperative to accomodate the BE traffic in TTEthernet networks. There is no research, that we are aware of, that supports the integration of BE traffic together with TT and RC traffic. Hence, in this paper we propose an extension to our TT and RC optimization approach DOTTS [18] to consider also the BE traffic, such that the TT and RC messages are schedulable and the bandwidth available for BE messages is maximized.

In Section V we show how DOTTS can be modified to take into account also BE (regular Ethernet) traffic, by updating its cost function and adding new design transformations. We refer to this updated version of DOTTS as DOTTS+. We have evaluated DOTTS+ using a real-life test case.

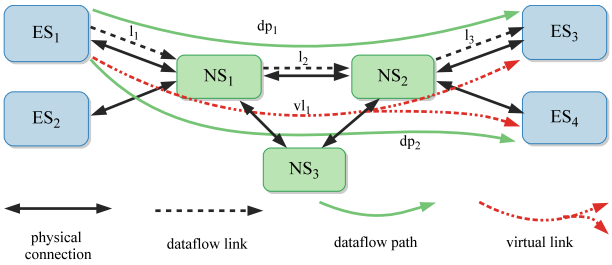


Fig. 1: TTEthernet cluster example

II. System Model

A TTEthernet network is composed of a set of clusters. Each cluster consists of End Systems (ESes) interconnected by links and Network Switches (NSes). The links are full duplex, allowing thus communication in both directions, and the networks can be multi-hop. An example cluster is presented in Fig. 1, where we have 4 ESes, ES_1 to ES_4, and 3 NSes, NS_1 to NS_3. The design problems addressed in this paper are performed at the cluster-level. Each ES consists of a processing element containing a CPU, RAM and non-volatile memory, and a network interface card (NIC).

We model a TTEthernet cluster as an undirected graph $\mathcal{G}_C(\mathcal{V}_C, \mathcal{E}_C)$, where $\mathcal{V}_C = \mathcal{ES} \cup \mathcal{NS}$ is the set of end systems ($\mathcal{ES}$) and network switches ($\mathcal{NS}$) and $\mathcal{E}$ is the set of physical links. For Fig. 1, $\mathcal{V}_C = \mathcal{ES} \cup \mathcal{NS} = \{ES_1, ES_2, ES_3, ES_4\} \cup \{NS_1, NS_2, NS_3\}$, and the physical links $\mathcal{E}$ are depicted with thick, black, double arrows.

A *dataflow path* $dp_i \in \mathcal{DP}$ is an ordered sequence of dataflow links connecting one sender to one receiver. For example, in Fig. 1, dp_1 connects ES_1 to ES_3, while dp_2 connects ES_1 to ES_4 (the dataflow paths are depicted with green arrows). A *dataflow link* $l_i = [\nu_j, \nu_k] \in \mathcal{L}$, where $\mathcal{L}$ is the set of dataflow links in a cluster, is a directed communication connection from ν_j to ν_k, where ν_j and $\nu_k \in \mathcal{V}$ can be ESes or NSes. Using this notation, a dataflow path such as dp_1 in Fig. 1 can be denoted as $[[ES_1, NS_1], [NS_1, NS_2], [NS_2, ES_3]]$.

The space partitioning between messages of different criticality transmitted over physical links and network switches is achieved through the concept of *virtual link*. Virtual links are defined by ARINC 664p7 [1], which is implemented by the TTEthernet protocol, as a "logical unidirectional connection from one source ES to one or more destination ESes".

We denote the set of virtual links in a cluster with $\mathcal{VL}$. A virtual link $vl_i \in \mathcal{VL}$ is a directed tree, with the sender as the root and the receivers as leafs. For example, vl_1, depicted in Fig. 1 using dot-dash red arrows, is a tree with the root ES_1 and the leafs ES_3 and ES_4. Each virtual link is composed of a set of dataflow paths, one such dataflow path for each root-leaf connection. More formally, we denote with $\mathcal{R}_{VL}(vl_i) = \{\forall dp_j \in \mathcal{DP} | dp_j \in vl_i\}$ the routing of virtual link vl_i. For example, in Fig. 1, $\mathcal{R}_{VL}(vl_1) = \{dp_1, dp_2\}$.

TTEthernet transmits data using *frames*. The TTEthernet frame format fully complies with the ARINC 664p7 specification [1]. Messages are transmitted in the payload of frames. A bit pattern specified in the frame header identifies the traffic class of each frame (TT, RC or BE). The size $m_i.size$ for each message $m_i \in \mathcal{M}$ is given, where $\mathcal{M}$ is the set of all messages. We assume that the designer has decided the traffic classes for each message. We define the sets $\mathcal{M}^{TT}$, $\mathcal{M}^{RC}$ and $\mathcal{M}^{BE}$, respectively, with $\mathcal{M} = \mathcal{M}^{TT} \cup \mathcal{M}^{RC} \cup \mathcal{M}^{BE}$. In addition, for the TT and RC messages we know their periods / rate and deadlines, $m_i.period$ or $m_i.rate$, and $m_i.deadline$, respectively. RC messages are not necessarily periodic, but have a minimum inter-arrival time. We define the rate of an RC message m_i as $m_i.rate = 1/m_i.period$. We model BE messages as aperiodic or sporadic, with a minimum inter-arrival time (denoted by $m_i.period$ for each $m_i \in \mathcal{M}^{BE}$). In TTEthernet networks, BE traffic can be either statically routed, or be routed via address learning in the switches. In this paper, we assume that the BE traffic is statically routed.

We compute the necessary bandwidth for a message m_i as:

$$BW_{Req}(m_i) = \frac{m_i.size}{m_i.period} \tag{1}$$

In case of the aperiodic BE messages, we substitute in Eq. 1 $m_i.period$ with T_{cycle}, defined as the least common multiple of the periods of the TT messages.

We have shown in [17] how the TTEthernet protocol works for TT and RC frames. Next, we will show how BE messages are transmitted. For the sake of simplicity, we consider the same topology as in Fig. 1. We consider that ES_1 transmits TT messsage TT_1 to ES_3, and RC message RC_1 to both ES_3 and ES_4, while ES_2 sends RC message RC_2 to ES_3, and the BE messages BE_1 to BE_4 to ES_4. Fig. 2a depicts, using a Gantt diagram, the transmission of frames on dataflow links $[ES_1, NS_1]$, $[ES_2, NS_1]$ and $[NS_1, NS_2]$.

TTEthernet has three traffic integration policies: shuffling, preemption and timely block (described in [17]). In this paper we consider the *timely block* policy: a low priority frame cannot be transmitted if it interferes with the transmission of a scheduled TT frame. Such a timely block interval is shown in Fig. 2a with a hatching pattern on $[NS_1, NS_2]$ between BE_1 and TT_1. We denote with $BW_{blocked}(l_i)$ the bandwidth taken by the timely blocked intervals on l_i. TT frames have the highest priority, while BE messages have no timing guarantees, and have the lowest priority. Although BE frames BE_2 and BE_3 arrive at NS_1 before RC frames RC_1 and RC_2, the BE frames have the lowest priority, so they can be transmitted only when the link is not occupied by TT or RC frames. Thus, BE_2 is transmitted at time unit 19, while BE_3 and BE_4 will be held at NS_1 and will be eventually dropped, since there is not sufficient available bandwidth on $[NS_1, NS_2]$. Thus, in Fig. 2a $BW_{blocked}([NS_1, NS_2])$ = 150kbps, corresponding to the timely blocked interval around time unit 11. On the other hand, the *shuffling* traffic integration policy allows lower criticality messages to be sent even if this will result in delaying the transmission of a scheduled TT frame. Thus, shuffling improves the bandwidth utilization for BE and RC traffic, while increasing the jitter for the TT frames.

We define the required bandwidth by BE traffic on a dataflow link l_i $BW^{BE}_{Req}(l_i)$ as the sum of the required bandwidth by all the BE messages transmitted over l_i.

$$BW^{BE}_{Req}(l_i) = \sum_{m_i \in l_i} BW_{Req}(m_i) \tag{2}$$

Similarly, $BW^{TT}_{Req}(l_i)$ and $BW^{RC}_{Req}(l_i)$ define the necessary bandwidth for TT and RC frames on l_i. Thus, we define the available bandwidth for BE traffic on a dataflow link l_i as

$$BW_{Avail}(l_i) = BW^{\circ} - BW^{TT}_{Req}(l_i) - BW^{RC}_{Req}(l_i) - BW_{blocked}(l_i) \tag{3}$$

where BW° is the initial bandwidth of the dataflow link.

III. Problem formulation

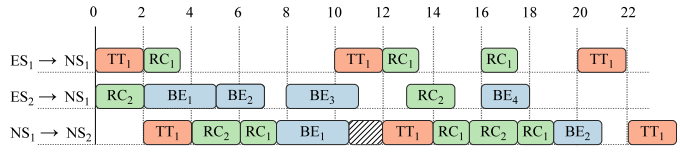
Our problem can be formulates as follows: given the network topology $\mathcal{G}$, the set of messages $\mathcal{M}$ (including the size and period/rate for all messages, and the deadline for the TT and RC messages), we are interested to optimize the network implementation Υ such that the TT and RC frames are schedulable, and the bandwidth for the BE frames is maximized.

We exemplify our problem using the topology presented in Fig. 1. The solution presented in Fig. 2a was obtained with DOTTS, which optimizes the network implementation considering only TT and RC messages, and ignoring the BE traffic. Moreover, in DOTTS we consider that the BE frames are routed along the shortest route. Thus, in this configuration dataflow link [NS_1, NS_2] does not have sufficient bandwidth for all the BE frames. However, taking into account the BE traffic when optimizing the network implementation leads to solutions which provide more bandwidth for the BE traffic, while at the same time guaranteeing the timing constraints of the TT and RC traffic, see Fig. 2b. The solution in Fig. 2b was obtained by rerouting the RC frame RC_2 and the BE frames BE_1 and BE_3 from the virtual link [[ES_2, NS_1], [NS_1, NS_2], [NS_2, ES_4]] considered in Fig. 2a, through the virtual link composed of dataflow links [[ES_2, NS_1], [NS_1, NS_3], [NS_3, NS_2], [NS_2, ES_4]].

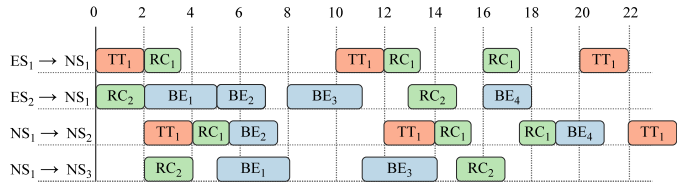
The increase in bandwidth available for BE messages can also be obtained, in general, by rescheduling TT frames, rerouting TT and RC frames or by packing / fragmenting.

IV. Design Optimization of TTEthernet-based Systems

Next, we will summarize the DOTTS strategy, presented in detail in [18]. DOTTS takes as input the topology of the network $\mathcal{G}$ and the set of TT and RC messages $\mathcal{M}^{TT} \cup \mathcal{M}^{RC}$ (including the size, period/rate and deadline), and returns (i) the fragmenting of messages Φ_m and packing in frames $\mathcal{K}$, (ii) the assignment of frames to virtual links $\mathcal{M}_F$, (iii) the routing $\mathcal{R}_{VL}$ of virtual links, (iv) the bandwidth $\mathcal{B}$ for each RC virtual link and (v) the schedules $\mathcal{S}$ for the TT frames. We denote with Υ the tuple $< \Phi_m, \mathcal{K}, \mathcal{M}_F, \mathcal{R}_{VL}, \mathcal{B}, \mathcal{S} >$ obtained by running the DOTTS algorithm.



(a) DOTTS does not consider the BE traffic. Frames BE_3 and BE_4 are dropped since they do not have enough bandwidth for transmission.



(b) Rerouting RC_2, BE_1 and BE_3 via NS_3 ensures enough bandwidth for all BE traffic.

Fig. 2: Motivational example

DOTTS is a Tabu Search-based metaheuristic, which searches for that solution that minimizes the *cost function*:

$$Cost = w_{TT} \times \delta_{TT} + w_{RC} \times \delta_{RC} \tag{4}$$

where δ_{TT} and δ_{RC} are the degree of schedulability for the TT and RC frames, respectively. These are summed together into a single value using the weights w_{TT} and w_{RC}, given by the designer. In case a frame is not schedulable, its corresponding weight is a very big number, i.e., a "penalty" value. This allows us to explore unfeasible solutions (which correspond to unschedulable frames) in the hope of driving the search towards a feasible region. Once the TT frames are schedulable we set the weight w_{TT} to zero, since we are interested to minimize the end-to-end delays for the RC frames. The degree of schedulability is calculated as:

$$\delta_{TT/RC} = \begin{cases} c_1 = \sum_i \max(0, R_{f_i} - f_i.deadline) & \text{if} c_1 > 0 \\ c_2 = \sum_i (R_{f_i} - f_i.deadline) & \text{if} c_1 = 0 \end{cases} \tag{5}$$

If at least one frame is not schedulable, there exists one R_{f_i}, i.e., the worst-case end-to-end delay (WCD) of f_i, greater than the deadline $f_i.deadline$, and therefore the term c_1 will be positive. We have discussed in [17] how the WCD of a frame is calculated. However if all the frames are schedulable, this means that each R_{f_i} is smaller than $f_i.deadline$, and the term $c_1 = 0$. In this case, we use c_2 as the degree of schedulability, since it can distinguish between two schedulable solutions.

Tabu Search explores the design space by using design transformations (or "moves") applied to the current solution in order to generate neighboring solutions. DOTTS has three classes of moves: (1) *routing* moves applied to virtual links, (2) *packing* moves applied to messages and (3) *scheduling* moves applied to the TT frames.

V. Optimization for BE-traffic

DOTTS is a flexible framework that can be used to optimize different aspects of a TTEthernet network. We will show next how to modify DOTTS to obtain schedulable implementations that maximize the available bandwidth for the BE traffic. The first step is to modify the cost function used by DOTTS from Eq. 4 to the following:

$$Cost = \begin{cases} c_1 = \sum_i \max(0, R_{f_i} - f_i.deadline) & c_1 > 0, f_i \in \mathcal{M}_{TT} \cup \mathcal{M}_{RC} \\ c_2 = \sum_j \max(0, BW^{BE}_{Req}(l_j) - BW_{Avail}(l_j)) & c_1 = 0 \text{ and } c_2 > 0 \\ c_3 = \sum_j (BW^{BE}_{Req}(l_j) - BW_{Avail}(l_j)) & c_1 = 0 \text{ and } c_2 = 0 \end{cases} \tag{6}$$

Once all the TT and RC frames are schedulable, i.e., each R_{f_i} is smaller than the deadline $f_i.deadline$, and as such, $c_1 = 0$, DOTTS will search for a solution that satisfies the bandwidth requirements of the BE frames. If at least one dataflow link $l_j \in \mathcal{DL}$ exists, where the $BW^{BE}_{Req}(l_j)$ (Eq. 2) bandwidth required by the BE frames routed via l_j is larger than the available bandwidth $BW_{Avail}(l_j)$ (Eq. 3), the BE traffic has insufficient bandwidth, i.e., $c_2 > 0$, and DOTTS will use c_2 as the value of the cost function. In case all the dataflow links have sufficient available bandwidth to satisfy the BE traffic, c_2 is 0, and the value of the cost function will be c_3. This cost function will drive DOTTS towards solutions that satisfy the schedulability of TT and RC traffic and that maximize the available bandwidth for BE frames.

The second extension of DOTTS is concerned with new moves. In DOTTS, the reroute move focuses only on TT and RC frames, and the choice of new VLs was made randomly.

We introduce a *reroute* move for BE traffic, which selects BE frames from the most "congested" (i.e., highest bandwidth utilization) dataflow link and reroutes them via the least used links. We also introduce a *packing* move for BE messages, which selects the small BE messages routed on the same VL, and packs them together into a frame, thus reducing the incurred protocol overhead. We denote with DOTTS+ the application approach presented in this section.

VI. Experimental Evaluation

We evaluated the new DOTTS+ strategy using the real-life case study Orion Crew Exploration Vehicle (CEV), derived from [8]. The topology for the case study is shown in Fig. 3. The Orion CEV case study has 31 ESes and 13 NSes, connected by dataflow link transmitting at 100 Mbps, and 187 TT and RC messages, with parameters generated based on the messages presented in [8]. At the network level, the TT and RC messages require a little over 50 Mbps. For BE traffic, we added 123 messages with bandwidth requirements that vary between 96 kbps (the average internet radio stream) and 10 Mbps (the read/write speed of DVD at 1x speed). The BE messages require 351 Mbps of bandwidth at the network level. Overall, these messages resulted in 52,815 frame instances transmitted over the network. DOTTS is not able to accommodate BE traffic in the network resulting in dataflow links which have zero bandwidth for BE frames. However, we ran our DOTTS+ optimization for 90 minutes, we managed to obtain a solution which satisfies the deadlines of the TT and RC messages, and provides 73.47% of the bandwidth required by BE messages. The BE frames have enough bandwidth for transmission on 88 out of the 94 dataflow links.

We were also interested to determine the impact of the timely blocked integration policy, compared to shuffling, on the BE messages. Thus, the unusable bandwidth on a dataflow link due to the timely block intervals ranges from 0 Mbps (no TT frames transmitted on that link) to 7.36 Mbps. At the whole network level, the timely block intervals take up 198 Mbps of bandwidth. On the 6 dataflow links the BE frames do not have enough bandwidth for transmission, 33.65 Mbps of bandwidth is taken by the timely block intervals. In case the time constraints for TT messages are less stringent, and the TT traffic can tolerate a higher jitter, the timely blocked intervals would be eliminated by implementing the shuffling policy, thus satisfying 83.06 % of the required bandwidth for BE traffic.

VII. Conclusions and Future Work

In this paper we have proposed a design optimization strategy to take into account BE traffic in TTEthernet-based systems. The proposed strategy, named DOTTS+, optimizes the network implementation such that the TT and RC messages are schedulable, and the available bandwidth for BE traffic is maximized. We have evaluated the DOTTS+ strategy using the Orion CEV real-life test case, and we have also shown the impact of timely traffic integration policy on the BE traffic.

DOTTS+ is based on a Tabu Search meta-heuristic. As future work, we will implement a heuristic-based approach, which would considerably reduce the necessary time to obtain a solution. Moreover, in this paper we modeled BE messages as aperiodic, with a minimum inter-arrival time. However, for many non real-time applications it is not possible to derive such parameters. Thus, another challenge is to improve the BE traffic model. Finally, we will evaluate DOTTS+ on several real-life and synthetic benchmarks.

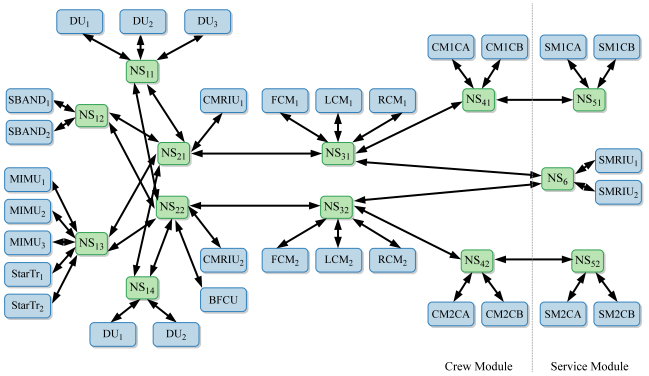


Fig. 3: Network topology of the Orion CEV, derived from [8]

References

[1] ARINC. *ARINC 664P7: Aircraft Data Network, Part 7, Avionics Full-Duplex Switched Ethernet Network*. ARINC, 2009.

[2] S. Chakraborty, S. Kunzli, and L. Thiele. A general framework for analysing system properties in platform-based embedded system designs. In *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, pages 190–195, 2003.

[3] J. D. Decotignie. Ethernet-based real-time and industrial communications. *Proceedings of the IEEE*, 93(6):1102–1117, 2005.

[4] ETG. *ETG.1000.1 EtherCAT Specification*. EtherCAT Technology Group, 2013.

[5] M. Fletcher. Progression of an open architecture: from Orion to Altair and LSS. White paper S65-5000-20-0, Honeywell, International, 2009.

[6] IEEE. *IEEE 802.3 - IEEE Standard for Ethernet*. IEEE, 2012.

[7] M.-Y. Nam, J. Lee, K.-J. Park, L. Sha, and K. Kang. Guaranteeing the End-to-End Latency of an IMA System with an Increasing Workload. *IEEE Transactions on Computers*, 99(PP):1, 2013.

[8] M. Paulitsch, E. Schmidt, B. Gstöttenbauer, C. Scherrer, and H. Kantz. Time-triggered communication (industrial applications). In *Time-Triggered Communication*, pages 121–152. CRC Press, 2011.

[9] SAE. *AS6802: Time-Triggered Ethernet*. SAE International, 2011.

[10] S. Schneele and F. Geyer. Comparison of IEEE AVB and AFDX. In *Proceedings of the Digital Avionics Systems Conference*, pages 7A1–1–7A1–9, 2012.

[11] R. Schneider, L. Zhang, D. Goswami, A. Masrur, and S. Chakraborty. Compositional analysis of switched ethernet topologies. In *Proceedings of the Design, Automation Test in Europe Conference Exhibition*, pages 1099–1104, 2013.

[12] T. Steinbach, H.-T. Lim, F. Korf, T. C. Schmidt, D. Herrscher, and A. Wolisz. Tomorrow's In-Car Interconnect? A Competitive Evaluation of IEEE 802.1 AVB and Time-Triggered Ethernet (AS6802). In *IEEE Vehicular Technology Conference*, pages 1–5, September 2012.

[13] W. Steiner. An Evaluation of SMT-based Schedule Synthesis For Time-Triggered Multi-Hop Networks. In *Proceedings of the Real-Time Systems Symposium*, pages 375–384, 2010.

[14] W. Steiner. Synthesis of Static Communication Schedules for Mixed-Criticality Systems. In *Proceedings of the International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, pages 11–18, 2011.

[15] J. Suen, R. Kegley, and J. Preston. Affordable avionic networks with Gigabit Ethernet assessing the suitability of commercial components for airborne use. In *Proceedings of SoutheastCon*, pages 1–6, 2013.

[16] E. Suethanuwong. Scheduling time-triggered traffic in TTEthernet systems. In *Emerging Technologies Factory Automation*, pages 1–4, 2012.

[17] D. Tămaş-Selicean, P. Pop, and W. Steiner. Synthesis of Communication Schedules for TTEthernet-based Mixed-Criticality Systems. In *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis*, pages 473–482, 2012.

[18] D. Tămaş-Selicean, P. Pop, and W. Steiner. Design Optimization of TTEthernet-based Distributed Real-Time Systems. *submitted to Real-Time Systems Journal*, 2014.