# Using Timing Analysis for the Design of Future Switched Based Ethernet Automotive Networks

Jonas Rox and Rolf Ernst
Institute of Computer and Network Engineering
Technische Universität Braunschweig
Hans-Sommer-Str. 66, 38106 Braunschweig, Germany
Email: {rox|ernst}@ida.ing.tu-bs.de

Paolo Giusto
General Motors R & D
422 Portage Avenue, Palo Alto, CA 94306
Email: paolo.giusto@gm.com

***Abstract*—In this paper, we focus on modeling and analyzing multi-cast and broadcast traffic latencies on switch-level within an Ethernet- based communication network for automotive applications. The analysis is performed adapting existing worst/best case schedulability analysis concepts, techniques, and methods. Under our modeling assumptions, we obtain safe bounds for both the minimum (lower bound) and maximum (upper bound) latencies. The formal analysis results are validated via simulation to determine the probability distribution of the latencies (including the worst/best case ones). We also show that the bounds can be tightened under some assumptions and we sketch opportunities for future work in this area. Finally, we show how formal analysis can be used to quickly explore tradeoffs in the system configuration which delivers the required performance. All results in this work are obtained on a moderately complex yet meaningful automotive example.**

## I. Introduction

The increasing bandwidth, rate, time determinism, low jitter, and low cost requirements for infotainment, autonomous driving, and diagnostics have led automotive OEMs to consider Ethernet as a possible alternative to the existing network communication protocols. Ethernet-based protocol specifications are either under development or have been completed. Ethernet-based full-duplex data transmissions are promising as they could provide the required high bandwidth/high rate, standard routing mechanisms that will enable flexible communication across multiple networks (intra-vehicle and V2X), with relatively low jitter and high determinism. In fact, although Ethernet-based protocols might experience switch congestion, they are inherently collision free because of their full-duplex communication architecture. In addition, the standardized nature of Ethernet protocols could provide cost-efficient solutions.

A foreseeable realistic use case is the adoption of Ethernet for the backbone network of automotive E/E architectures or at minimum for domains (e.g., chassis). As illustrated in Fig. 1 the backbone network interconnects the different domains. When designing such an Ethernet-based backbone network, there are several design options leading to very different system performances which may be very challenging to predict. To enable system architects and network designers to fully explore the trade-offs of Ethernet-based E/E architectures and eventually create a close-to-optimal design solution, methods and tools that support the exploration, analysis, and

978-3-9810801-8-6/DATE12/ © 2012 EDAA

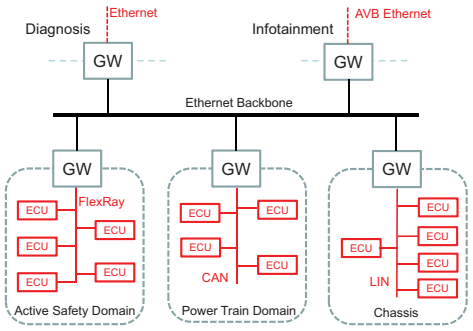


Fig. 1: Assumed system architecture

optimization steps of the development process are required. In this paper, our focus is on the modeling, analysis, and simulation of multi-cast and broadcast traffic at the switch level. The novelty of our approach resides in the analysis aspect as it is performed using worst/best case schedulability analysis concepts, techniques, and methods. In addition to provide best and worst case figures for latencies from a sending node to a receiver node, we also perform simulations to determine the distribution of the latencies and therefore the probability that the worst and best case figures will occur.

The rest of this paper is organized as following. In Sec. II, we provide a short overview of the effects of a switched based standard Ethernet based architecture relevant for the communication timing. In Sec. III, we first give a brief introduction into the adopted formal analysis approach and then we discuss the modeling artifacts that we use for the worst/best case timing analysis of the system. In Sec.IV we present how we modeled the system for simulation. Next, in Sec. V we conduct experiments and compare the results of the analysis and the simulation. Finally, we draw our conclusions.

## II. Timing in Ethernet Based Networks

The goal of our performance evaluation is to obtain the frame end-to-end latency, from the instant when it starts traversing the backbone network until it arrives in the receiving buffer of the receiving gateway. The backbone network is realized as a switch-based Ethernet network with full duplex links between all participating nodes. Full-duplex operation is an optional MAC capability that allows simultaneous two-way transmission over point-to-point links. Full duplex transmission is functionally much simpler than half-duplex transmis-

sion because it involves no media contention, no collisions, and no need to schedule retransmissions. The result is not only more time available for transmission, but also doubling of the link bandwidth because each link can now support full-rate, simultaneous, two-way transmission. Transmission can usually begin as soon as frames are ready to send. The only restriction is the minimum-length inter-frame gap (96 bits) between successive frames. Hence, there is no interference on the physical links between traffic sent in different directions. Therefore, the transmission delay incurred by a packet is equal to its size in bytes (including the overhead bytes in addition to the data payload) divided by the bit rate.

While the time a specific Ethernet frame occupies a link is easy to determine, the time a Frame takes to cross a switch is calculated considering different factors:

1) Store and forward mechanism,
2) switch fabric processing,
3) wire line transmission and
4) frame queuing.

*Store and forward latency:* Store and forward refers to the basic operating principle of an Ethernet switch. The received data is stored in the switch memory until the complete frame is received. The frame is then transmitted out the ports leading to the frames destination. The introduced latency is proportional to the size of the transmitted frame and inversely proportional to the bit rate.

*Switch fabric latency:* The switch fabric consists of sophisticated silicon that implements the store and forward engine and MAC address table, among other functions. The execution of the logic which implements these functions creates a certain delay. While the execution of the internal logic of a switch is rather fast, the exact execution time may vary, depending on the amount of incoming and outgoing traffic.

*Wire line latency:* Bits transmitted on a fiber optic link travel at about $2/3$ of the speed of light ($3 \cdot 10^8$ m/s). This latency becomes significant only for very long wires. Considering distances involved in local area networks, this delay becomes trivial compared with the other contributions to latency.

*Queuing latency:* Ethernet switches supporting IEEE 802.1q (VLAN) use priority sorted queues in conjunction with a scheduler to eliminate the problem of frame collisions that used to exist on broadcast Ethernet networks. The latency factor introduced by queuing can often be very difficult to predict, since all traffic patterns on a network have to be considered. Additionally, the effects of the used scheduling policy have to be taken into account.

In the following section we will first briefly introduce the compositional analysis approach implemented by the tool SymTA/S [1]. Then, we will present how the different parts of the system we want to analyse can be modeled using the underlying concepts of the analysis approach. This finally enables us to apply formal analysis to determine worst/best case latency bounds for the modeled system. Similar, in Section IV we shortly present how we model relevant system aspects for simulation.

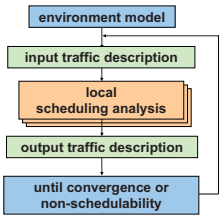


Fig. 2: System Level Performance Analysis Loop

## III. Compositional System Level Analysis

Compositional performance analysis [2], [3], [4] exhibits great flexibility and scalability for timing and performance analysis of complex distributed embedded real-time systems. The basic idea of compositional approaches (see Fig. 2) is to break down the analysis complexity of complete systems into separate local component analyses and to integrate existing local performance analysis techniques, e.g. uniprocessor scheduling analysis known from real-time research, into system level analyses. The composition is achieved by connecting the component's inputs and outputs by stream representations of their communication behavior using event models. The basic elements used to model the performance characteristic of the system under analysis are *event streams*, *resources* and *tasks*.

A *task* is the smallest unit modeling performance characteristics at the application level. Tasks are mapped and executed on a set of processing (CPUs) and communication (Buses) elements, representing the system architecture. Each task is characterized by its core execution time interval (CET interval), defined as the minimum and maximum times the task requires for a complete execution on the corresponding resource, assuming that no blocking or preemption occur during execution. A task graph describes the functional and timing dependencies between tasks. Tasks are allowed to have more than one immediate successor and predecessor.

Tasks are activated by events which are modeled as *event streams* rather than as sequences of individual events. Events can be both from an external source, such as the expiration of a timer, or from another task. Generally, event streams are described using the functions $\delta^+(n)$ and $\delta^-(n)$ that represent the maximum and minimum distance between any $n$ ($n \geq 2$) events in the stream. Essentially, these functions state the worst and best case scenario that the designer specifies. For computational efficiency, event models can be represented with various parameters. For example, the standard event models capturing key properties of event streams use three parameters, the activation period $T$, the event jitter $J$, and the minimum distance between any two events $d$.

*Resources* represent the system architecture. A task must be mapped onto a resource to execute. When multiple tasks share the same resource, then two or more tasks may request the resource at the same time. In order to arbitrate request conflicts, a resource is associated with a scheduler which selects a task to be executed from the set of active tasks according to some scheduling policy. In the analysis, the tasks are not executed. The scheduler is a function that transforms the event streams into new event streams. To apply the analysis

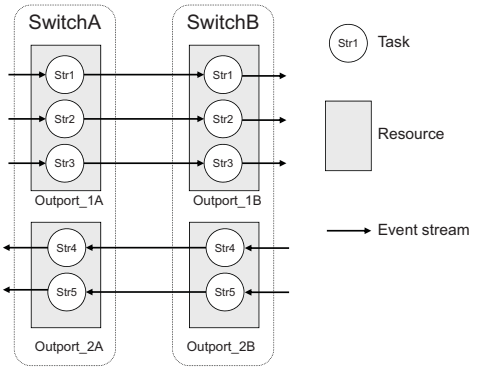


Fig. 3: Model of two interconnected switches

to our Ethernet architecture, we provide a reference design which utilizes the modelling artifacts described earlier.

### A. Switch Model

To model the Ethernet switches, we use the general model introduced in [5]. In the following, we summarize the main concepts. For each of switch **output ports**, we use a resource. We do not model the switch fabric - we assume *switch fabric latencies* as a best and worst case delays of 0 and 2 micro seconds for analysis, and a uniform distribution of delays within the same range for simulation. Fig. 3 depicts the model of two interconnected switches.

For each different traffic stream going though an output port, a task is mapped onto the corresponding resource. The minimum and maximum execution times of the tasks are given by the minimum and maximum transmission times of the Ethernet frames of the corresponding traffic stream. Thus, the response times of the tasks model the sum of the *queuing latency* and the *store and forward latency*. Essentially, the store and forward latency is modeled by the task execution time, while the queuing latency is modeled by task blocking time (due to another task occupying the resource which models a link being currently used by a packet) and interference (which is due to other tasks with higher priority in the queue which models higher priority packets waiting to be transmitted). Since the compositional analysis approaches we adopt [2] allows tasks with multiple output event streams, multi-cast communication can easily modeled by connecting one task with several output event streams to different tasks mapped on different resources modeling the output ports which the multi-cast communication traverses.

Assuming static priority **non-preemptive** scheduling which is appropriate for Ethernet links modeled as tasks executing on shared resources, the worst case response time $R_i^{max}$ of the task $\tau_i$ with maximum execution time $C_i$ can be calculated using existing worst case response time analysis [6])

Table I summarizes the previously presented mapping of network primitives to modeling artifacts used for our analysis.

## IV. Simulation

Our simulation model for a Standard Ethernet based communication architecture is created using the tool VisualSim by Mirabilis Design [7]. For the purpose of modeling and simulating latencies, we have used an abstract data structure that encapsulates the real Ethernet frames. During simulation, data structure objects are created by the traffic generators at the specified rate. Processing blocks instantiated in the model can access and modify the data structure object field. The simulator API automatically updates other fields such as the trace, that is the switches that have been traversed by the frame from source to destination, and the actual simulated time the object was generated. This information is very useful for both debugging the model, and to determine the source to destination frame latency. In the following, we briefly describe some of the basic modeling artifacts we have used for the simulation.

### A. Priority Sorted Queue

A *Priority Sorted Queue* is used to model the queuing latency of the packets due to the potential congestion at the switch. Each packet arriving at the switch is delayed by at least its store and forward latency and at most by a delay which is a function of the longest packet currently being transmitted from the switch to the destinations (using the outgoing links) and by the higher priority packets arriving at the queue while the link is busy. Since the packets are scheduled for transmission using a priority based non-preemptive policy, even if a lower priority packet is still being transmitted while an higher priority packet has arrived at the queue, the higher priority packet is obviously blocked.

### B. Switch

The Switch primitive block is used to define a Node on the network and can be connected to other blocks to construct a complete network with multiple layers of the protocol stack. It can be used to define clients, servers, routers and switches to form a full system. In our system, we used this block to create a linear topology which we will explain in more detail in the next section. Other topologies could also easily be created.

Finally, a switch is configured by a routing table. The routing table is used to specify the logical connections between the switches. Other important parameters for the switch configuration, are the routing algorithm used during the simulation to find a path from source to destination, and the parameters used by the routing algorithm (e.g., number of hops) to decide the route.

### C. Multi Cast Block

While the switch is used to specify the logical connections between the switches, essentially the logical topology, the multi cast block is used to specify the destination of broad cast transmissions from one source to the destinations and the two-way unicast transmission (request,acknowledge) protocol for diagnostic. As, in our experimental set up (which is presented in the following section), every source sends UDP packets to all the other destinations in the architecture, the multi cast is essentially a broadcast. The diagnostic protocol is instead realized as 6 unicast TCP/IP based transmissions from the diagnostic block to all other blocks. No other sources send diagnostic requests. Upon the reception of a diagnostic request each destination sends an acknowledgment.

TABLE I: Mapping of network primitives to modeling artifacts

| **Network primitives** | **Modeling artifacts in SymTA/S** |
|---|---|
| Output port with scheduler | **CPU** with corresponding **scheduling policy** |
| Forwarding of an Ethernet frame | Execution of a **task** |
| Frames routed through the network | Event triggered **task chains** |
| Application behavior | **Event streams** bounding the number of frames sent |
| Store and forward latency of a frame | **Execution time** of the corresponding task |
| Queuing latency of a frame | **Interference** and **blocking** due to other tasks |
| Queuing latency + store and forward latency of a frame | **Response time** of the corresponding task |

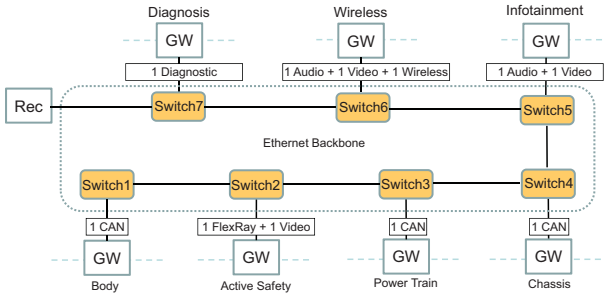


Fig. 4: Line Topology

### D. Traffic Generator

The generator can be used to generate periodic bursts of events. Both the period, the maximum number of events per period, and the minimum distance between consecutive events is configurable via parameters. In our simulations, we have used both a configuration where 10 events are generated at any given period, where their minimum distance is equal to zero (traffic with no shaping) and more periodic streams where every (outer) period, a sequence of activation events are generated with a minimum distance (inner period).

## V. Experiments

In this section we apply the formal analysis and the simulation to determine the end-to-end latencies of different traffic streams in a system example which utilizes a full duplex switched based Ethernet backbone network for the inter-domain communication.

### A. Experimental Set Up

In the system, we assume that different domains, e.g. body, power train, communicate over an switch-based Ethernet backbone. The considered topology is illustrated in Fig. 4. The different domains are interconnected by a set of switches which are arranged in a linear topology so that each packet that is broadcast from one source to the destinations traverses the network via a multi-hop path. As can be seen in Fig. 4, we have modeled a receiver `Rec` connected to `Switch7`, which is used as measurement end point for determining the maximum latencies.

Here, we focused on modelling, analyzing and simulating only the timing behavior of the switched Ethernet network. Thus, we do not model the communication behavior within the domains, but instead we model the system as if the gateways are the sources for the traffic generated by the domains. There are the following 6 types of applications, which generate different traffic streams. Note that the packet sizes given in Table II only specify the application data.

TABLE II: The applications

| Application | Priority | Req. Bandwidth | Traffic Pattern |
|---|---|---|---|
| FlexRay | 3(highest) | 2.048 Mb/s | 10 x 256-byte/10ms |
| CAN | 3(highest) | 256 Kb/s | 10 x 32-byte/10ms |
| Diagnostic | 2 | 30.72Kb/s | 6 x 2 x 32-byte/100ms |
| Audio | 1 | 1.178 Mb/s | 1 x 1472-byte/10ms |
| Video | 1 | 4.71 Mb/s | 16 x 1472-byte/40ms |
| Wireless | 0(lowest) | 23.55 Mb/s | 2000 x 1472-byte/10 s /w $d_min$ of 500us |

TABLE III: Applications running within the different domains

| Domain | Running Applications |
|---|---|
| Body | 1 CAN |
| Active Safety | 1 FlexRays + 1 Video |
| Powertrain | 1 CAN |
| Infotainment | 1 Audio + 1 Video |
| Wireless | 1 Audio + 1 Video + 1 Wireless |
| Diagnostic | 1 Diagnostics |

Thus, the size of the Ethernet frames transmitted over the backbone network is 12 (gap) + 8 (preamble) + 18 (header with 802.1q) + 4 (trailer) + 20 (IPv4 header) + 8 (UDP header) = 70 bytes larger than the application packet size (we assume that each application packet is encapsulated into one Ethernet frame) for the applications which use UDP, i.e. all of the applications except the Diagnostic application which uses TCP. The Ethernet frame size for the diagnostic traffic is 12 (gap) + 8 (preamble) + 18 (header with 802.1q) + 4 (trailer) + 20 (IPv4 header) + 20 (TCP header) = 82 bytes larger than the application packet size given in Table II. The Ethernet frame sizes and the resulting transmission times are given in Table IV.

The assumed mapping of Applications to domains (respectively traffic sources to gateways) is summarized in Table III. All packets sent by the applications except for the diagnostic packets are broadcast, as they are sent to every other gateway. The diagnostic packets however are different. The diagnostic gateway sends a diagnostic request to other gateways every 100 ms. Thus, it sends 6 packets, where each packet is sent to one domain gateway and the other gateways respond with a corresponding diagnostic response as soon as the diagnostic request is received (the processing time of the diagnostic request is assumed to be zero). The diagnostic response has the same priority as the diagnostic request. We further assume that no transmission errors occur and thus there is no packet loss on the network which would result in retransmissions.

As described in Sec. III the output ports of the switches are modeled as resources and on every such resource, one task for each traffic stream traversing the output port is mapped. The scheduling policy used is static priority non-preemptive, using FIFO scheduling for tasks with the same priority. The execution times of the tasks are given by the frame size divided

TABLE IV: Task parameters

| Task Name | Transmitted Bytes | Execution Time | Priority |
|---|---|---|---|
| CAN (UDP) | 102 | 8.16 $\mu$s | 3 (highest) |
| FlexRay (UDP) | 326 | 26.08 $\mu$s | 3(highest) |
| Diagnostic (TCP) | 114 | 9.12 $\mu$s | 2 |
| Audio (UDP) | 1542 | 123.36$\mu$s | 1 |
| Video (UDP) | 1542 | 123.36$\mu$s | 1 |
| Wireless(UDP) | 1542 | 123.36$\mu$s | 0 (lowest) |

TABLE V: Results for the Linear Topology with shaping

| | Latency [min, max] in ms | |
|---|---|---|
| Traffic | Formal Analysis | Simulation |
| CAN | [0.033, 2.335] | [0.033, 1.021] |
| FlexRay | [0.156, 2.269] | [0.156, 0.940] |
| Audio | [0.247, 6.434] | [0.247, 3.278] |
| Video | [0.247, 10.015] | [0.247, 6.979] |
| Wireless | [0.247, 13.653] | [0.247, 6.924] |
| Diagnostic | [0.128, 5.863] | [0.128, 1.067] |

through the total bandwidth of the network, which is assumed to be 100 Mbit/s. Table IV summarizes the parameters of the different tasks

For each of the switches a packet forward time (from input port to correct output port) is assumed to be 2us in the best case, and zero in the worst case. In the simulation, we use a uniform distribution with range 0-2us. Using the same parameters and the concepts introduced in Sec. IV we compare the minimum and maximum latencies for the different application streams obtained with formal analysis with the results obtained via simulation. The results are presented in the following subsection.

### *B. Results*

For our comparison, we will focus on the maximum latencies for the different traffic streams from the instant a corresponding packet was generated by its generating source, until it is received by the receiver node `Rec` (which is connected to `Switch7`, see Fig. 4). The results are summarized in Table V. Except for the diagnostic traffic, the minimum (maximum) latencies given in Table V are the minimum (maximum) of all minimum (maximum) path latencies from any source sending the specified traffic to the receiver node `Rec`. E.g. the maximum latency given for CAN is the maximum latency of a frame from `Body` $\rightarrow$ `Rec` and the minimum latency for CAN is the minimum latency of a CAN frame sent from `Chassis` $\rightarrow$ `Rec`. The minimum latency for the diagnostic traffic is the minimum path latency of the path (`Diagnostic` $\rightarrow$ `Wireless` $\rightarrow$ `Diagnostic`) and the maximum latency for the diagnostic traffic is the maximum path latency of the path (`Diagnostic` $\rightarrow$ `Body` $\rightarrow$ `Diagnostic`). I.e. the time from sending the request until the reception of the corresponding acknowledgement. The latencies obtained by formal analysis and simulation for the diagnostic traffic assume that the TCP connection had already been established a priori, i.e. the additional traffic and delay due to the 3-way handshake mechanism was not considered. Finally, all the best case numbers include a best case zero switch fabric delay for each hop that is part of the end to end path.

We also evaluate the effect of using traffic shaping by enforcing a specific minimum separation between consecutive Ethernet frames sent by the gateways. Initially we use the specific shaping parameters (i.e. the enforced minimum separation between frames) - 1ms for CAN and FlexRay and 1.125ms for the Video traffic.

TABLE VI: Rresults for the Linear Topology with shaping

| | Latency [min, max] in ms | |
|---|---|---|
| Traffic | Formal Analysis | Simulation |
| CAN | [0.033, 9.993] | [0.033, 9.795] |
| FlexRay | [0.156, 9.974] | [0.156, 9.786] |
| Audio | [0.247, 1.530] | [0.247, 0.840] |
| Video | [0.247, 18.990] | [0.247, 15.972] |
| Wireless | [0.247, 2.956] | [0.247, 0.918] |
| Diagnostic | [0.128, 2.223] | [0.128, 0.735] |

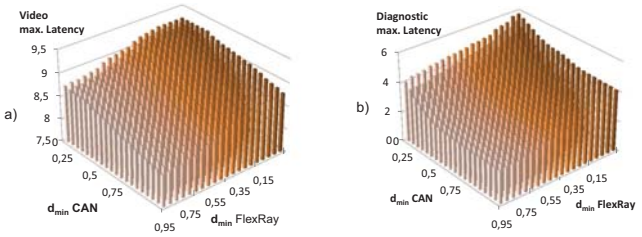


Fig. 5: a) Max. latencies for the video traffic and b) max. latencies for the Diagnostic traffic for different shaping parameters for the CAN and FlexRay traffic obtained with the formal analysis. All values are in ms.

The obtained latencies with these shaping parameters are summarized in Table VI. Applying traffic shaping drastically decreases the calculated worst case latency for the lower priority traffic which is not shaped (e.g., the audio, the wireless and the diagnostic traffic). On the other hand, the latencies of the shaped traffic streams drastically increase, because of latencies introduced by the shaper for the shaped stream. Thus, with the chosen shaping parameters, the negative impact of the traffic shaping on CAN, FlexRay and Video traffic is larger than the positive impact on the other sources. In the presence of real-time constraints it would not be obvious, which parameters to choose for the different traffic shaper to fulfill all requirements.As has been shown in [8] and [9], the used formal analysis is well suited for design space exploration. This allows us to evaluate different parameter configurations for the system to find a suitable configuration fulfilling all constraints within a feasible time frame.

To illustrate this, we performed an exploration of different for the shaping parameter values. To do so, we gradually increase the enforced minimum distance between consecutive frames for the CAN and for the FlexRay traffic and for each such configuration we determine the maximum end-to-end latencies for the different streams. For these experiments we assume no traffic shaping for the video traffic.

As shown in Fig. 5, increasing the enforced minimum distances, greatly reduces the maximum latencies for the lower priority traffic, i.e. the diagnostic and the video traffic. As shown in Fig. 6a, the impact of increasing $d_{min}$-FlexRay, i.e. the enforced minimum distance between consecutive FlexRay frames, does only marginally increase the performance of the CAN traffic. But it has a significant negative influence on the performance of the FlexRay traffic itself. Similarly, increasing $d_{min}$-CAN only slightly decreases the maximum latencies of the FlexRay traffic, but it drastically increases the maximum latency of the CAN traffic itself. Depending on the

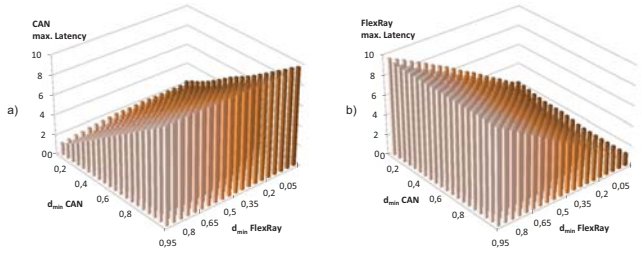


Fig. 6: a) Max. latencies for the CAN traffic and b) max. latencies for the FlexRay traffic for different shaping parameters for the CAN and FlexRay traffic obtained with the formal analysis. All values are in ms.

constraints of the different traffic streams, such an analysis is well suited to help the designer to find a good solution. Using the formal analysis, 400 different system configurations could be analyzed within a few minutes using a standard desktop PC. In the same fashion also different topologies could be evaluated and compared to each other.

### C. Discussion of Results

The key property of compositional analysis is that each component is analyzed individually. This makes it necessary to consider the individual worst case on every component to be able to guarantee conservative results. Thus, by adding the worst case response times of every task along a specific path to obtain the path end-to-end latency, it is guaranteed (under the assumption that all input data was conservative) that the end-to-end latency bound is conservative.

But for most systems (including the example system described in this paper) it is obvious that the considered scenario, i.e. a packet suffering the worst case response time on every component, is very unlikely to happen in reality. So, even after (hypothetically) performing an exhaustive simulation of the system, the observed latencies would always be lower than the bound given by the formal analysis. In fact, there is no general way to determine the scenario, i.e. a combination and phasing of stimuli (instants the packets are generated by the applications), which leads to the worst case end-to-end latency for a specific path. Essentially, simulating switch congestion at every switch is almost an impossible task.

From the results shown one can see that for the traffic with higher priority, the worst case latency observed in the simulation is quite close to the guaranteed upper bound determined with formal analysis. The lower the priority of the traffic, the larger the difference between simulation and analysis results become. This is due to the individual analyses of the components in compositional performance analysis. The main contribution to the pessimism of the response time of a task comes from the interference of other tasks (with higher priority), which by itself is pessimistic. Thus, the response times of lower priority tasks are more pessimistic compared to the response times of higher priorities tasks, because they suffer interference from more tasks.

To obtain better results in the formal analysis, i.e. less pessimistic upper bounds for the end-to-end latencies, different options exist. By assuming completely independent tasks, (which in fact is done by using the analysis in [6]) the obtained results may become very pessimistic for a larger number of tasks. Thus, an important aspect would be to investigate methods to exploit certain dependencies between the task activation timings, e.g. similar to the approach presented in [10]. Unfortunately, the method presented in [10] is only useful to improve the WCRTs of preemptively scheduled tasks, but the extension of the method to non-preemptively scheduled tasks is currently researched. Also, the techniques presented in [11] and [12] may improve the component analysis results by exploiting dependencies between task activation timing (i.e. excluding the typical critical instant). But these techniques would also have to be adapted to non-preemptive scheduling and thus, they are not directly applicable here.

## VI. Conclusion

In this paper, we have described a modeling methodology for analysis of Standard Ethernet-based E/E architectures. The methodology provides a mapping of modeling artifacts (tasks, resources, and schedulers) into Ethernet elements constituting the network. Under some assumptions made, we assess that our analysis is a useful tool for early conceptual design of Ethernet based architecture topologies. We have validated the analysis results using simulation. We have modeled only some high level aspects of the Standard-Ethernet protocol, but especially in early design phases, this may well be sufficient, since details about the concrete implementation are often not available anyway. However, we plan to extend our model to include aspects that can reduce the pessimism of the analysis.

## References

[1] http://www.symtavision.com, "Symtavision gmbh," Internet.

[2] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, "System Level Performance Analysis - The SymTA/S Approach," *IEE Proceedings Computers and Digital Techniques*, vol. 152, no. 2, pp. 148–166, March 2005.

[3] S. Chakraborty, S. Künzli, and L. Thiele, "A general framework for analysing system properties in platform-based embedded system designs," *Design, Automation and Test in Europe Conference and Exhibition, 2003*, pp. 190–195, 2003.

[4] T. Henzinger and S. Matic, "An Interface Algebra for Real-Time Components," in *Proc. of the IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, April 2006.

[5] J. Rox and R. Ernst, "Formal timing analysis of full duplex switched based ethernet network architectures," in *SAE World Congress*, vol. System Level Architecture Design Tools and Methods (AE318). Detroit, MI, USA: SAE International, Apr 2010.

[6] R. Davis, A. Burns, R. Bril, and J. Lukkien, "Controller area network (can) schedulability analysis: Refuted, revisited and revised," *Real-Time Systems*, vol. 35, pp. 239–272, 2007, 10.1007/s11241-007-9012-7. [Online]. Available: http://dx.doi.org/10.1007/s11241-007-9012-7

[7] http://www.mirabilisdesign.com/, "Mirabilis design," Internet.

[8] R. E. Kai Richter, "Early architecture exploration with symta/s." in *Automation, Assistence and Embedded Real Time Platforms for Transportation (AAET)*, Braunschweig, Germany, 2005.

[9] A. Hamann, M. Jersak, K. Richter, and R. Ernst, "Design space exploration and system optimization with symta/s - symbolic timing analysis for systems," in *Proc. 25th International Real-Time Systems Symposium (RTSS'04)*, Lisbon, Portugal, dec 2004.

[10] J. Rox and R. Ernst, "Exploiting inter-event stream correlations between output event streams of non-preemptively scheduled tasks," in *Proceedings Design, Automation and Test in Europe (DATE 10)*, Dresden, Germany, 2010, p. 226.

[11] K. Tindell, "Adding time-offsets to schedulability analysis," University of York, Tech. Rep., 1994.

[12] J. Palencia and M. Harbour, "Exploiting precedence relations in the schedulablilty analysis of distributed real-time systems," in *Proc. 20th IEEE Real-Time Systems Symposium (RTSS99)*, 1999.