Investigating and Analyzing CAN-to-TSN Gateway Forwarding Techniques

Aldin Berisa, Mohammad Ashjaei, Masoud Daneshtalab, Mikael Sjödin, Saad Mubeen
School of Innovation, Design and Engineering
Mälardalen University, Västerås, Sweden
{firstname.lastname}@mdu.se

***Abstract*—Controller Area Network (CAN) and Ethernet network are expected to co-exist in automotive industry as Ethernet provides a high-bandwidth communication, while CAN is a legacy cost-effective solution. Due to the shortcomings of conventional switched Etherent, such as determinism, IEEE Time Sensitive Networking (TSN) task group developed a set of standards to enhance the switched Ethernet technology providing low-jitter and deterministic communication. Considering these two network domains, we investigate various design approaches for a gateway that connects a CAN domain to a TSN domain. We present three gateway forwarding techniques and we develop end-to-end delay analysis methods for them. Via the analysis methods and applying them to synthetic use cases we show that the intuitive existing approach of encapsulating multiple CAN frames into a single Ethernet frame is not necessarily an efficient solution. In fact, we demonstrate several cases where it is preferable to encapsulate only one CAN frame into a TSN frame, in particular when we use a high speed TSN network. The results have a significant impact on developing such gateways as the implementation of the one-to-one frame encapsulation is considerably simpler than other complex gateway-forwarding techniques.**

I. Introduction

Modern cars contain several tens of Electronic Control Units (ECUs) that are connected by various in-vehicle networks [11]. Controller Area Network (CAN) [18] is one of the most widely used in-vehicle real-time network protocols. The classical CAN protocol is limited by its speed (maximum network speed is 1 Mbit/s) and the size of data that can be transmitted in a single CAN frame (maximum data size is 8 bytes). Consequently, CAN is unable to meet the high-bandwidth communication requirements in the contemporary and the next generation of self-driving vehicles.

Switched Ethernet is being adopted by the automotive industry because it provides the required high-bandwidth support of up to 10 Gbit/s and beyond. Switched Ethernet does not support low-jitter and timing predictable communication on its own, which is required by many vehicular applications [10]. To address these shortcomings, the IEEE Time Sensitive Networking (TSN) task group [1] developed a set of TSN standards that are used on top of switched Ethernet. The TSN standards offer a promising solution to support high-bandwidth, low-latency, low-jitter and timing predictable communication within vehicles [2]. Although a complete transition to real-time Ethernet, such as TSN, is anticipated in the future, it will be a gradual process because the legacy in-vehicle networks like CAN are still widely used in the automotive domain due to their low cost [22]. Hence, the networks like CAN and TSN are expected to coexist within vehicles, where multiple CAN domains would communicate via a TSN backbone domain [6].

There are various gateway forwarding techniques that enable communication between CAN and Ethernet. These techniques encapsulate multiple CAN frames inside an Ethernet frame to efficiently utilize the Ethernet bandwidth [17]. The encapsulation process can incur large delays for CAN frames while they wait to be encapsulated. To limit this delay, the gateway techniques often include the concepts of timers and "urgent" frames that immediately trigger the transmission of the Ethernet frame without waiting for encapsulation of all the CAN frames [9], [19]. However, the gateway forwarding techniques in the existing works have focused only on CAN and Ethernet. There is only one work that presents a gateway for CAN and the precursor of TSN, called the Ethernet Audio-Video Bridging (AVB) [7]. To the best of our knowledge, none of the existing works addresses gateway techniques for CAN and TSN.

An intuitive approach to develop a CAN-TSN gateway mechanism is to build upon the existing CAN-Ethernet gateway mechanisms. These mechanisms advocate to encapsulate multiple CAN frames into a single Ethernet frame. In this paper, we investigate this encapsulation mechanism on CAN-TSN gateway and argue that it may not be an efficient solution. In fact, we demonstrate several cases where it is preferable to encapsulate only one CAN frame into a TSN frame, such as when using a 1Gbit/s speed of TSN network. The evaluation results indicate that the one-to-one frame encapsulation mechanism in CAN-TSN gateway can improve end-to-end delays and bandwidth utilization in the TSN domain. Furthermore, this mechanism significantly reduces the complexity of CAN-TSN gateways.

A. Paper contribution

In this paper, we investigate different gateway forwarding techniques to connect a CAN domain to another CAN domain via a TSN network. The main contributions in this paper are as follows.

- We present three gateway forwarding techniques to connect CAN and TSN networks. These techniques are based on CAN and legacy Ethernet gateways.

- We propose end-to-end delay analysis methods for the presented CAN-TSN gateway forwarding techniques.
- We evaluate the CAN-TSN gateway forwarding techniques using the proposed analysis methods based on synthetic use cases. In particular, we conduct the performance evaluation of these techniques with respect to end-to-end delays and TSN bandwidth utilization.

B. Paper Layout

The rest of the paper is organized as follows. We provide an overview of the background and the existing related work in CAN-Ethernet gateways in Section II. We present our gateway model and forwarding techniques in Section III. We develop a compositional end-to-end delay analysis for each gateway forwarding technique in Section IV. Section V presents the evaluation based on synthetic use cases, and finally Section VI presents the conclusion.

II. Background and Related work

A. Controller Area Network (CAN)

CAN is a communication protocol that is widely used in embedded systems, particularly when dealing with real-time applications. It was designed in the 1980s by Robert Bosch GmbH and is still the most popular in-vehicle network today. CAN supports the network speeds of up to 1 Mbit/s, but most real-world implementations operate at 500/250 Kbit/s. Each CAN frame has a defined payload size (up to 8 bytes). CAN broadcasts frames using the fixed-priority non-preemptive scheduling, which means that once a frame starts its transmission, it cannot be aborted, and the highest priority frame is transmitted first. Because CAN uses bus topology, it enables ECU communication without the need for a host ECU. The main advantage of CAN is that it allows for predictable real-time data transmission with high reliability and low latencies. CAN has evolved over time, with the most recent generation, CAN Extra Long (XL) [8], supporting much higher speeds and larger payloads, among other improvements. The standardization of CAN-XL is ongoing.

B. Time sensitive networking

Time-sensitive networking (TSN) standards include a set of mechanisms built on top of standard switched Ethernet by the TSN task group [1] to support high-bandwidth, low-jitter, timing predictable and real-time communication in addition to non-real-time traffic. The standards provide time-critical services such as time synchronization with IEEE 802.1AS, time-aware shaper IEEE 802.1Qbv, frame preemption IEEE 802.1Qbu, and much more.

One of the most important mechanisms of TSN is that all network devices share a common sense of time. This is accomplished through the use of IEEE 802.1AS time synchronization, which provides a precision clock synchronization mechanism that synchronizes the clocks of the devices with sub-microsecond accuracy. Another important feature of TSN is traffic scheduling, which allows bandwidth to be allocated only to specific traffic classes using IEEE 802.1Qbv and 802.1Qbu, which provide time-aware scheduling and frame preemption. This allows offline scheduled traffic (ST) to be transmitted without latency using these mechanisms. TSN was built on top of IEEE 802.1AVB and thus supports real-time rate-constrained traffic managed by a credit based shaper and is scheduled online.

C. Related work

Many gateway solutions exist that allow interfacing between CAN and Ethernet. The majority of the research suggests various techniques for mapping CAN frames into Ethernet frames, with Ethernet serving as a backbone for connecting multiple CAN domains.

Scharbarg et al. [17] were among the first to propose a gateway technique between multiple CAN domains and Legacy Ethernet by waiting for a queue to fill up with a specific number of CAN frames, encapsulating them inside one Ethernet frame and then sending it across the Ethernet network to the gateway of the receiving CAN domain. The CAN frames experience a significant delay while waiting for the queue to fill up in this approach, hence they propose a timer that, when it expires, encapsulates the current CAN frames already in the queue and sends out the Ethernet frame. The main shortcoming of this work is that the solution only supports legacy switched Ethernet. Kern et al. [9] also introduce the concept of "urgent" frames on a CAN to Switched Ethernet gateway, which cause an immediate transmission. They also calculate the average end-to-end latency of CAN frames traversing between CAN domains and measure the used Ethernet bandwidth in their work.

Nacer et al. [16] propose a "traffic shaping" mechanism for a CAN to Switched Ethernet gateway that reduces the additional load on the CAN bus caused by the decapsulated CAN frames released by the gateway. It works by delaying the frame until exactly one cycle has passed since the gateway released the last instance of the same frame. The disadvantage of this approach is that the CAN frames released by the gateway have an additional delay while waiting for the cycle to pass. Moreover, Thiele et al. [19] present existing gateway techniques as event models that can be combined to calculate the worst-case timing of the Ethernet network and gateways.

The above-mentioned works considered the standard switched Ethernet with no timing guarantee. Therefore, Herber et al. [7] present a CAN-AVB gateway approach in which different queuing techniques for CAN frames are evaluated to forward the CAN messages to the AVB network. The AVB frames are sent out cyclically with a period determined by the number of CAN frames encapsulated in the AVB frame. Although they use AVB as a predictable switched Ethernet, the delays are calculated only for the gateway waiting time, i.e., the end-to-end delay was not considered. To the best of our knowledge, none of the existing works investigated gateway solutions for CAN-TSN networks. In this paper, thus, we show that using the already proposed techniques, e.g., CAN-AVB are not necessarily an efficient way of forwarding techniques for CAN-TSN gateways.

Considering the timing analysis methods, many works addressed worst-case response time (WCRT) analysis of CAN and TSN. For instance, Tindell et al. [20] presented one of the early works for calculating the worst-case response time of CAN frames which was later revised by Davis et al. [5]. Following that, there were numerous extensions to support the analysis of CAN frames in various scenarios. Mubeen et al. [14] added support for mixed messages, while support for mixed messages with offsets was added in [12]. Further, support for FIFO and priority-based queues in CAN controllers were added in [13].

In the context of TSN, several works developed an analysis technique for computing the worst-case response time of TSN messages. Bordoloi et al. [4] presented an analysis for messages in TSN when there is no time-aware shaper enabled. Later, Lo Bello et al. [3] proposed a timing analysis technique to compute the worst-case response time of messages in TSN when preemption is enabled. Similarly, a network calculus-based techniques was proposed by Zhao et al. [21] to calculate the worst-case delay of messages in TSN networks.

While there are analysis techniques for CAN and TSN seperately, there is no work addressing integration of these two network domains, i.e., calculation of worst-case delay for a CAN messages traverses to a TSN network via a gateway. We develop such an analysis in this paper.

TABLE I: Summary of notation.

Symbol	Meaning
$fwd(q)$	Set of CAN frames forwarded to queue q
$T_{TSN}(q)$	Period of the TSN frame created from CAN frames forwarded to queue q
Fi	CAN frame with ID i
T_i	Period of the inter-domain CAN frame i
β	The number of CAN frames encapsulated inside a TSN frame
D_i	Total delay experienced by frame i
$R_i^{CAN_s}$	Response time of frame i while being transmitted in the sending CAN domain
D_i^{fwd}	Forwarding delay experienced by frame i when it waits in the queue to be encapsulated
ε^E	Encapsulation delay
R_i^{TSN}	Response time of the TSN frame that encapsulates the CAN frame i
ε^D	Decapsulation delay
$R_i^{CAN_r}$	Response time of frame i while being transmitted in the receiving CAN domain
$\mu_{FIFO}(q)$	Number of TSN frame instances frame i in queue q has to wait until encapsulated with FIFO queuing
$D_{FIFO}^{fwd}(q)$	Worst case FIFO delay for a frame in queue q in terms of time unit
$\mu_{i,FP}(q)$	Number of TSN frame instances frame i in queue q has to wait until encapsulated with FP queuing
$D_{i,FP}^{fwd}(q)$	Worst case FP delay for frame i in queue q in terms of time unit

III. CAN-TSN GATEWAY FORWARDING TECHNIQUES

This section presents the proposed gateway model including the gateway structure, forwarding techniques in the gateway, and the traffic model.

A. Gateway Model

A CAN-TSN gateway is a node that transfers CAN frames from the CAN network domain to the TSN network domain using an encapsulation and forwarding technique. The techniques will be explained in detail in the next subsection. The CAN frames that traverse between different domains are regarded as the inter-domain frames. The maximum number of inter-domain CAN frames that can be encapsulated in a single TSN frame is limited by the maximum size of payload in a TSN frame. A TSN frame can carry a maximum payload of 1500 bytes, while the maximum size of any CAN frame including its overhead is approximately 17 bytes (135 bits to be precise). Therefore, in the worst-case scenario, where each CAN frame has the maximum payload of 8 bytes, up to 88 CAN frames can be encapsulated in one TSN frame.

The gateway can only generate periodic TSN frames with classes A and B, while the scheduled traffic (ST) class is not supported. The reason is that CAN is an event-driven network that uses online scheduling, whereas the ST traffic in TSN is scheduled offline. Furthermore, sporadic frames are also not supported in the gateway as they would require a more sophisticated technique for encapsulation in a single TSN frame, which would also have to be transmitted sporadically. The support for ST frames and sporadic TSN frames in the gateway is left for the future work. Table I summarizes all notations that are used in this work.

Various components of the gateway and how the received frames are forwarded in the gateway are depicted in Figure 1. After a CAN frame is sent to the gateway via the CAN network, it is stored in the receive buffer and an interrupt is generated. The frame is then read by the gateway dispatcher and stored in the appropriate memory queue based on the destination in the TSN network. The order in which the frames are stored in the queues depends on the queuing technique set at each queue, which can be First-in-First-out (FIFO), Fixed-priority (FP) or a one-to-one technique.

Each memory queue generates a TSN frame by forwarding the specified number of CAN frames from the queue to Ethernet MAC to be encapsulated inside the TSN frame. After expereincing the encapsulation delay, the generated TSN frame resides in the Ethernet MAC buffer until the specified period when the TSN frame is transmitted onward. The cyclic transmission of TSN frames allows for their predictable transmission while also limiting the delays experienced by CAN frames. This cyclic transmission is similar to the timer strategy used in [9], [17] and is preferred compared to waiting for the buffer to fill up with the specified number of CAN frames, which would result in an unpredictable transmission of the TSN frame and increase CAN frame delays. After the TSN frame is generated, the frame is sent to the Ethernet PHY for transmission across the TSN network. The other way around is very simple, as follows. Once the TSN frame reaches the destination gateway, the CAN frames are decapsulated by the

Ethernet MAC and then sent to the dispatcher to be transmitted across the CAN network. The CAN frames have the same period as the TSN frame, which allows them to be transmitted across the receiving CAN network as soon as they are released from the dispatcher.

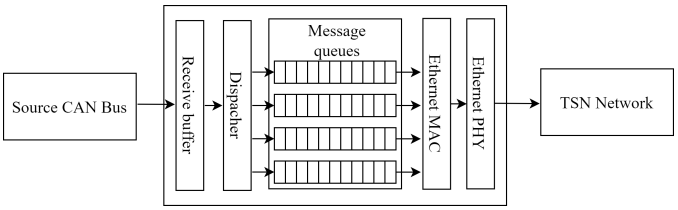


Fig. 1: High level diagram of a CAN-TSN gateway.

B. Gateway forwarding techniques

1) First-In-First-Out (FIFO) technique: In the First-In-First-Out (FIFO) forwarding technique, the CAN frames are added to one of the queues each time and the order in which they arrived is the order in which they will be dequeued. The main advantage of using FIFO is that it ensures fairness of frame forwarding and is simple to implement. The disadvantage is that the higher-priority frames can suffer from longer delays if lower-priority frames are queued ahead of them.

2) Fixed-priority (FP) technique: Fixed-priority (FP) queuing is a forwarding technique in which each CAN frame is assigned a priority, which in our case are the IDs of the CAN frames. The IDs of the CAN frames are assigned using rate monotonic ordering, so frames with lower periods will have higher priority. The ordering in the queue is determined by the priority of the frames, with the high-priority frames being dequeued first. The main advantage of this forwarding technique is that it reduces the delay experienced by high-priority frames, however at the cost of increasing the delay experienced by low-priority frames. The priority ordering in this work will be based on the CAN frame IDs, and the frame IDs are assigned with a rate-monotonic ordering, hence the lower the period, the higher the priority of the CAN frame.

3) One-to-one technique: Instead of encapsulating multiple CAN frames in a single TSN frame, a simpler approach is to encapsulate one CAN frame per TSN frame. The main advantage of this strategy is that the CAN frame does not experience the queuing delay that is required to wait for other CAN frames before it can be encapsulated in a TSN frame as in the case of other techniques. Furthermore, this technique is easy to implement and simple to use. In this technique, a CAN Frame is sent to the Ethernet MAC for encapsulation as soon as it arrives at the gateway. The encapsulating TSN frame created has the same period as that of the encapsulated CAN frame. The drawback of this technique is that it can incur a large overhead as the maximum size of a standard CAN frame (encapsulated frame) that uses the maximum data size of 8 bytes is approximately 17 bytes (135 bits), whereas the minimum size of a TSN frame (encapsulating frame) is 64 bytes including the 42-byte header. As a result, padding will be required to meet the minimum length of the TSN frame. Furthermore, this technique cannot utilize the large payloads supported by TSN frames (up to 1500 bytes of payload in a single frame). Another disadvantage is that this technique creates one TSN frame for each CAN frame, which consumes more TSN network bandwidth as compared to other techniques that encapsulate multiple CAN frames in a single TSN frame.

C. Architecture of a System Utilizing the CAN-TSN Gateways

A CAN-TSN gateway can be used in a variety of different architectures. A suitable architecture would have one or more CAN domains that use TSN as a backbone. The architecture in Figure 2 is inspired by a use case from the automotive industry that is provided by one of our industrial partners. The TSN network has endpoints that communicate with one another, and the CAN endpoints may also be able to communicate with them. In this work, we are only focusing on gateway-to-gateway communication but the approach can be applied for communication with any TSN endpoint.

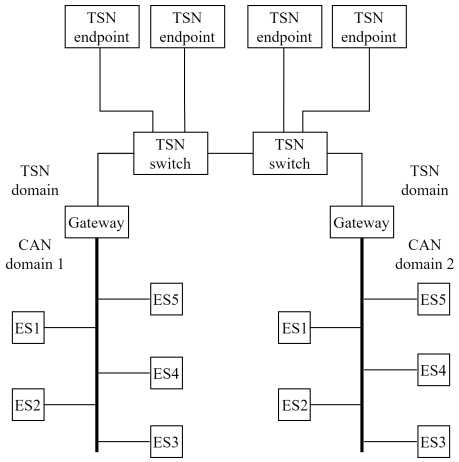


Fig. 2: A system architecture with two CAN domains and TSN as the backbone network.

IV. Proposed Timing Analysis

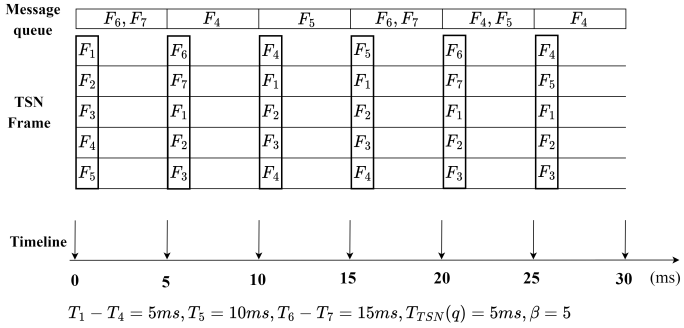
In this section, we present the timing analysis to analyze the delays experienced by the inter-domain CAN frames until they reach the destination node in the other CAN domain via the presented gateways. Note that the proposed analysis is compositional in the sense that the total delay can be composed by adding the individual delays incurred by various components in the gateway.

All of the TSN frames generated by the gateway are sent in a cyclic manner. The transmission period of the TSN frame can be calculated according to Equation (1), proposed by Herber at el. [7]. It works by calculating the number of frame instances per time unit for each CAN frame that is forwarded to the same queue q in the gateway. The summation in Equation (1) provides total frame rate in frames per time unit. Dividing β, which is the number of CAN frames encapsulated in one TSN frame, with the summation gives us the average rate for the

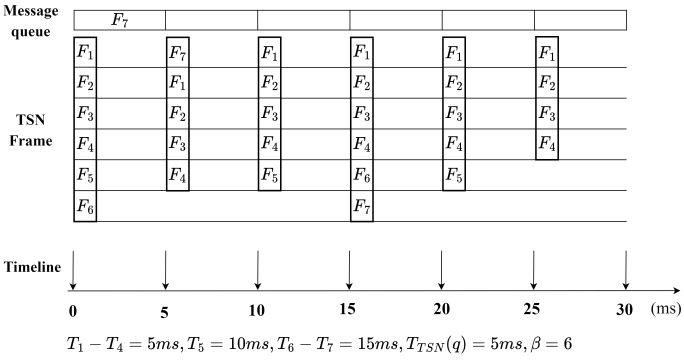
TSN frame. We additionally add the floor function over the whole value since the TSN frame period derived would not be a whole number which would create an enormous hyperperiod on the TSN network.

$$T_{TSN}(q) = \left\lfloor \beta / \sum_{\forall i \in fwd(q)} \frac{1}{T_i} \right\rfloor \quad (1)$$

We note that if the encapsulation size β is not carefully selected then not all CAN frame instances can be forwarded to the TSN network domain. For example, Figure 3 shows a timeline that consists of 7 inter-domain CAN frames until their hyperperiod of 30 ms. The periods of the CAN frames F_1-F_4 are 5 ms each, F_5 has a period of 10 ms, while F_6 and F_7 have 15 ms periods. The top part of this figure shows pending CAN frame instances that wait in the gateway queue to be encapsulated in the TSN frame. The middle part of the figure shows the CAN frame instances that are encapsulated in each instance of the encapsulating TSN frame. Whereas, the bottom part of the figure depicts the timeline where each downward arrow shows the time when each instance of the TSN frame is transmitted.



(a) Frame set with an infeasible β.



(b) Frame set with a feasible β.

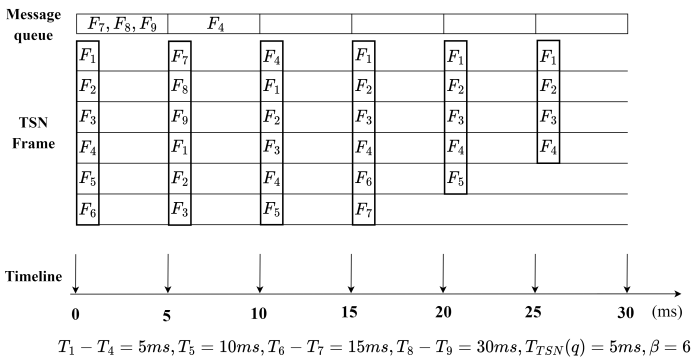
Fig. 3: Example of how the encapsulation size β can impact feasibility.

In Figure 3a the period of the TSN frame generated from the queue $T_{TSN}(q)$ is 5 ms, the encapsulation size β is 5, and the queuing technique used is FIFO. It can be observed that not all queued instances of the CAN frames will get encapsulated within the hyperperiod. The reason is that there are not enough frame slots available in each TSN frame instance to encapsulate all the queued instances of the CAN frames. If we increase β to 6 as shown in Figure 3b, all the instances of CAN frames will be encapsulated within the hyperperiod. We developed a utilization bound function as shown in Equation (2), that checks if a β is feasible or not. The equation must be true for β to be feasible. It calculates the total rate of inter-domain CAN frames arriving at the queue q, based on the CAN frames' periods T_i, which is then compared to the rate of the TSN frame being sent out of queue q with period $T_{TSN}(q)$ and multiplied with β. Where, β denotes the number of CAN frames that are encapsulated in the TSN frame. For the equation to be true, the rate at which CAN frames are encapsulated has to be equal to or higher than the rate at which the CAN messages are recieved in the queue.

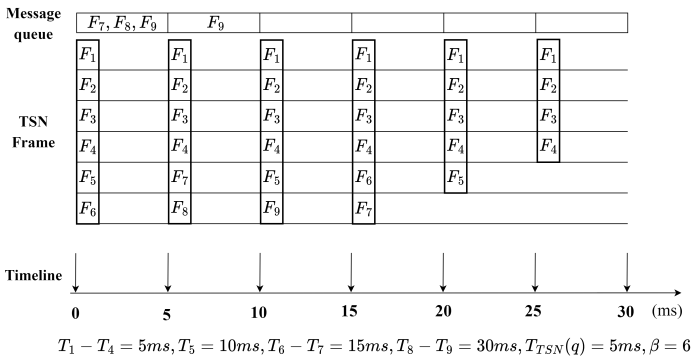
$$\sum_{\forall i \in fwd(q)} \frac{1}{T_i} \leq \frac{1}{T_{TSN}(q)} * \beta \quad (2)$$

An inter-domain CAN frame experiences different types of delays when transmitted throughout different domains. When an inter-domain CAN frame i initially gets released by the sending node, and until it reaches the gateway, the delay corresponds to the response time of the frame in the CAN domain. This delay is denoted by $R_i^{CAN_s}$. The response time of the frames in CAN domain are calculated by using the existing response-time analysis for CAN [5], [14], [15]. Next, the frame needs to wait in the appropriate queue until it can be forwarded for transmission. We regard this delay as the forwarding delay and denote it by D_i^{fwd}. The CAN frame may experience different forwarding delays depending on the queuing approach. After it has been forwarded, the CAN frames are moved to the Ethernet MAC buffer, encapsulated inside a TSN frame, and then wait to be sent out at the specified period. We define the delay while the CAN frames are encapsulated into a TSN frame as encapsulation delay, denoted by ε^E. The time required by the TSN frame to reach the destination gateway is regarded as the response time of the TSN frame, denoted by R_i^{TSN}. The response time of the TSN frame is calculated using the response-time analysis of TSN [3]. The gateway then needs to decapsulate the CAN frames from the TSN frame, which we define as the decapsulation delay, denoted by ε^D. Lastly, the CAN frame needs to be sent to the destination node in the receiving CAN network. The delay in this segment is the response-time of the CAN frame in the receiving CAN network, denoted by $R_i^{CAN_r}$. The total delay, D_i, experienced by the inter-domain CAN frame i is the sum of all the mentioned delays as shown in Equation (3).

$$D_i = R_i^{CAN_s} + D_i^{fwd} + \varepsilon^E + R_i^{TSN} + \varepsilon^D + R_i^{CAN_r} \quad (3)$$



(a) An example of FIFO queuing strategy.



(b) An example of FP queuing strategy.

Fig. 4: An example of a frame set using FIFO and FP gateway forwarding techniques.

A. First-In-First-Out (FIFO) forwarding delay

To derive the FIFO delay we need to take in consideration the worst-case scenario for a CAN frame, which is when all of the CAN frames have been queued ahead of the CAN frame under analysis. An example scenario is described in Figure 4a which contains 9 inter-domain CAN frames F_1-F_9. The periods of these frames are represented by T_1-T_9 respectively. The periods of the CAN frames F_1-F_4 are 5 ms each, F_5 has a period of 10 ms, the period of each of F_6 and F_7 is 15 ms, and the period of each of F_8 and F_9 is 30 ms. The period of the TSN frame generated from the queue $T_{TSN}(q)$ is 5 ms, the encapsulation size β is 6. The top part of Figure 4a shows pending CAN frame instances that wait in the gateway queue to be encapsulated in the TSN frame. The middle part of Figure 4a shows the CAN frame instances that are encapsulated in each instance of the encapsulating TSN frame. The bottom part of Figure 4a shows the timeline that consists of 9 inter-domain CAN frames until their hyperperiod of 30 ms, where each downward arrow shows the time when each instance of the TSN frame is transmitted.

If the frame under analysis is F_9, which is the last one to be queued, we can see in Figure 4a that this frame is forwarded for encapsulation in the second instance of the TSN frame. The number of TSN frame instances that must be transmitted before the last queued CAN frame can be encapsulated, denoted by $\mu_{FIFO}(q)$, is calculated using Equation (4). This is done by calculating the total number of all frame instances released during the TSN period and then dividing it by β. The number of TSN frame instances is incremented by 1 because we use the floor function in Equation (4).

$$\mu_{FIFO}(q) = \left\lfloor \frac{\sum_{\forall i \in fwd(q)} \lceil \frac{T_{TSN}(q)}{T_i} \rceil}{\beta} \right\rfloor \tag{4}$$

The worst-case delay in the queue for any frame under the FIFO gateway forwarding technique, denoted by $D^{fwd}_{FIFO}(q)$, is calculated by Equation (5). This delay is the product of the number of TSN frame instances incremented by 1 and the TSN frame period.

$$D^{fwd}_{FIFO}(q) = (\mu_{FIFO}(q) + 1) \times T_{TSN}(q) \tag{5}$$

B. Fixed-priority (FP) forwarding delay

In the case of the fixed-priority forwarding delay, each CAN frame experiences a different amount of delay depending upon its priority. It can be see in Figure 4b that the frame F_9 is encapsulated in the third instance of the TSN frame. This is due to the reason that F_9 is the lowest-priority CAN frame. Whereas, the CAN frame F_1 is encapsulated in every instance of the TSN frame due to it being the highest priority frame. To derive the forwarding delay for a CAN frame under analysis, we calculate the maximum number of TSN frame instances that are transmitted until the frame under analysis is encapsulated. This depends on the total number of instances of the high-priority frames that need to be encapsulated before the CAN frame under analysis.

The maximum number of TSN frame instances for which the CAN frame under analysis F_i has to wait in the queue, q, to get encapsulated in the TSN frame, denoted by $\mu_{i,FP}(q)$, can be calculated using Equation (6). This equation considers the total number of instances of the CAN frames that have higher or equal priority than the CAN frame under analysis F_i. The equation also takes into account the encapsulation delay ε^E, which specifies the time required for the CAN frames to be encapsulated into a TSN frame and then sent out at the specified period. Once the total number of instances of the higher or equal priority CAN frames is derived, we divide it by the encapsulation size β. This equation requires fixed-point iterations, with the initial number of instances set to zero.

$$\mu^{n+1}_{i,FP}(q) = \left\lfloor \frac{\sum_{\forall m \in hep(i)} \lceil \frac{\mu^n_{m,FP}(q) \times T_{TSN}(q) + \varepsilon^E}{T_m} \rceil}{\beta} \right\rfloor \tag{6}$$

Finally, in Equation (7), we multiply the calculated value of $\mu_{i,FP}(q)$ and incremented by 1 with the TSN frame period

to get the worst-case delay of the CAN frame F_i, denoted by $D^{fwd}_{i,FP}(q)$.

$$D^{fwd}_{i,FP}(q) = (\mu_{i,FP}(q) + 1) \times T_{TSN}(q) \quad (7)$$

V. Evaluation

In this section, we present a reproducible evaluation scenario where the delays experienced by inter-domain CAN frames are calculated using the proposed compositional delay analysis and the bandwidth utilization of the TSN backbone based on different encapsulation sizes β and TSN link speeds.

A. Evaluation scenario

The use-case scenario is designed to represent a realistic automotive application. The system architecture of the use case is depicted in Figure 2. The architecture consists of two CAN domains with six nodes, one of which is a gateway, and a TSN backbone with two switches and six nodes, two of which are gateways to the CAN domains.

In the scenario, the local frames for the CAN domains have periods of {10, 20, 50, 100} ms, which are assigned with probabilities of 4.8%, 14.3%, 33.3%, and 47.6 % respectively, as done in [7]. The payloads of the CAN frames are randomly chosen from the range of 0 to 8 bytes. Regarding local traffic of the TSN network, the TSN frames have periods of {1, 5, 10, 20, 50, 100} ms, with an equal probability distribution to be assigned to the TSN frames. The local TSN frames have an equal probability distribution as well for being assigned ST, A or B traffic classes. The payload of the TSN frames are set to 1500 bytes. The utilization of all domains is set to 50%. The encapsulation and decapsulation delays are both set to 1 ms; however, this value is technology and implementation dependent and may vary across hardware implementations.

We generated 100 synthetic test cases, each with 20 inter-domain CAN frames with the properties shown in Table II which are encapsulated into TSN frames of class A. Each inter-domain frame is transmitted from CAN domain 1 to CAN domain 2. All of the generated test cases were run with different sizes of β {1,5,10,15,20} to demonstrate their impact on the end-to-end delay of the CAN frames. We also calculated the TSN network bandwidth utilization based on β size and TSN link speed which was either 1000 Mbit/s or 100 Mbit/s. All β configurations were run with the previously defined FIFO and FP gateway forwarding techniques. These techniques are compared with the one-to-one gateway forwarding technique, where β is set to 1. The one-to-one technique is the most basic approach in which one CAN frame is encapsulated in one TSN frame as soon as the CAN frame arrives.

B. Evaluation results

Figures 5, 6, 7, and 8 show graphical representations of the end-to-end delays experienced by inter-domain CAN frames. Looking at the graphs, we can see that changing the encapsulation size β has a significant impact on the delays experienced by the CAN frames. The delays experienced by the CAN frames are also impacted by the gateway forwarding techniques. When we set β = 1, the frames have the least amount of delay because there is no queuing delay. We can see an increase in the delays of individual frames as we increase β from 1 to 5. The difference in the delays becomes even more noticeable when we increase β to 20. The relationship between the encapsulation size β and the period of the created TSN frame T_{TSN} causes the delays to increase significantly as we increase β. The T_{TSN} is calculated using the periods of the forwarded CAN frames and the encapsulation size β. If we increase β, the period of the TSN frame T_{TSN} becomes larger and thus the TSN frame is transmitted less frequently. This means that the CAN frames wait longer in the queue to be encapsulated.

Because CAN frames experience the least amount of delay, β = 1 is the most preferable encapsulation size for all CAN frames since there is no forwarding delay. CAN frames are sent to the Ethernet MAC for encapsulation as soon as they arrive at the gateway. The disadvantage of this technique is that padding will be required to meet the minimum length for the TSN frame. Furthermore, amount of payload that the TSN frame can carry (maximum of 1500 bytes) is underutilized. Another disadvantage is that 20 different TSN frames are created to transmit 20 CAN frames, which does not efficiently utilize the TSN network bandwidth compared to the other gateway forwarding techniques.

The gateway forwarding techniques also have an impact on the inter-domain frame delays. We can see that FP may be preferable for high-priority frames, shown with frames in Figure 5, because they experience less delay, whereas low-priority frames experience significantly large delays which can be noticed in Figure 8. In some cases, FIFO may be preferred over FP, the most notable being when β is equal to 5 or 10. In those cases, we can see that the low-priority frames have significantly less delay than FP. Whereas, the high-priority frames experience more delay because they are no longer queued ahead of the other frames. When we increase β to 15 and 20, FP appears to be more preferred because the low-priority frames experience the same delay as if FIFO was used, while the high-priority frames experience less delay. The main reason that low-priority frames in FP with lower β have higher delay compared to FIFO is that the low-priority frames may get stuck for a long time in the queue due to high frequency of high-priority frames arriving to the queue. This is the case specially when the priorities are selected based on Rate Monotonic (RM) algorithm, i.e., shorter periods, higher priority.

In the graph shown in Figure 9, the bandwidth utilization of the TSN network is displayed on the y-axis. The graph depicts two TSN network speeds {100, 1000} Mbit/s, and the varying encapsulation sizes {1, 5, 10, 15, 20} used to encapsulate CAN frames into an Ethernet frame. We can immediately see that if we use a TSN network with 1000 Mbit/s speed, there is not much of a difference between encapsulating multiple CAN frames that use around 1% of the bandwidth and encapsulating one CAN frame in one TSN frame which in total uses 2.5% of the bandwidth. When we reduce the speed of the TSN links to 100 Mbit/s, the difference becomes more noticeable. The

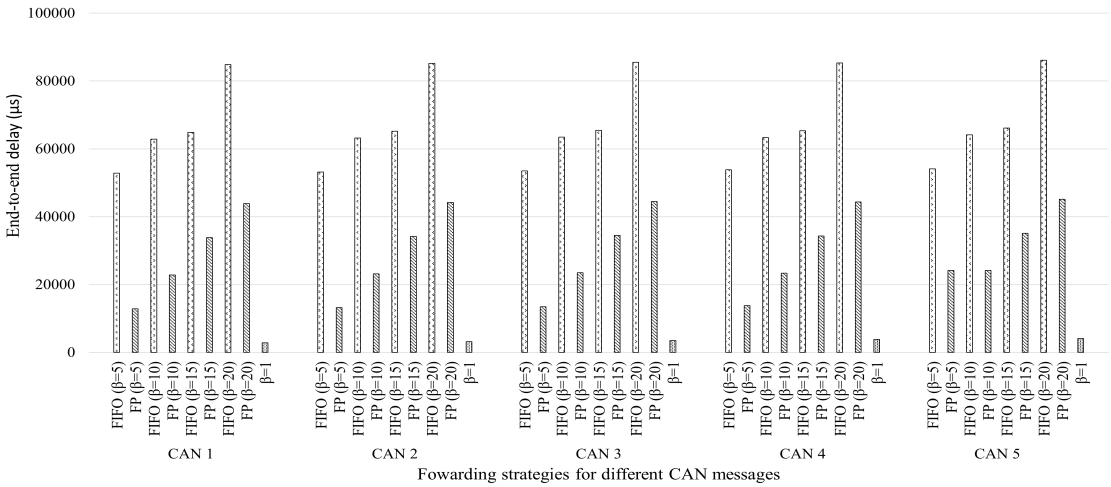


Fig. 5: Evaluation results for inter-domain CAN frames 1-5.

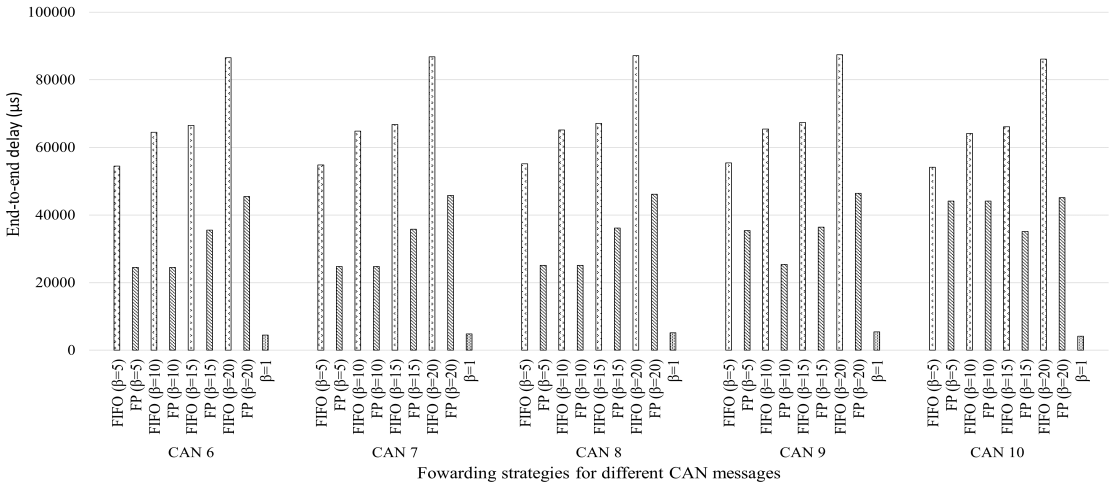


Fig. 6: Evaluation results for inter-domain CAN frames 6-10.

ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
T (ms)	10	20	20	50	50	50	50	50	50	50	100	100	100	100	100	100	100	100	100	100
DLC	1	3	2	3	2	4	2	3	1	5	3	6	2	2	1	2	8	2	7	1
TX	ES4	ES3	ES4	ES3	ES4	ES5	ES5	ES3	ES4	ES1	ES4	ES3	ES2	ES5	ES3	ES1	ES3	ES4	ES3	ES1
RX	ES4	ES5	ES4	ES5	ES4	ES4	ES4	ES1	ES4	ES5	ES4	ES4	ES4	ES5	ES3	ES2	ES4	ES5	ES3	ES1

TABLE II: Properties of the inter-domain CAN frames used in the evaluation: ID of the frame, period (T), data length code (DLC), sending node (TX), and receiving node in the other CAN domain (RX).

bandwidth utilization in this case is significantly higher if we use β = 1, which is 25%, and it can be reduced to 7.1% if we use higher values of β, in this case β = 20. It is worth mentioning that we have tested the network speed of 10 Mbit/s for this experiment and the network was over-utilized in case of β = 1. However, we believe that using speed below 100 Mbit/s is not common in industry as the solution will lose its interest.

Even though using β = 1 introduces padding overhead and consumes more bandwidth in the TSN network than using higher values of β, the lower frame delays experienced with this approach suggest that it may be preferable compared to encapsulating multiple CAN frames into a single TSN frame. This is especially the case if we use 1000 Mbit/s speed on the links in the TSN network. There may be times when encapsulating multiple CAN frames makes sense, such as when using TSN links with speeds of 100 Mbit/s, 10 Mbit/s, or when it makes sense to pack some CAN frames together and send them

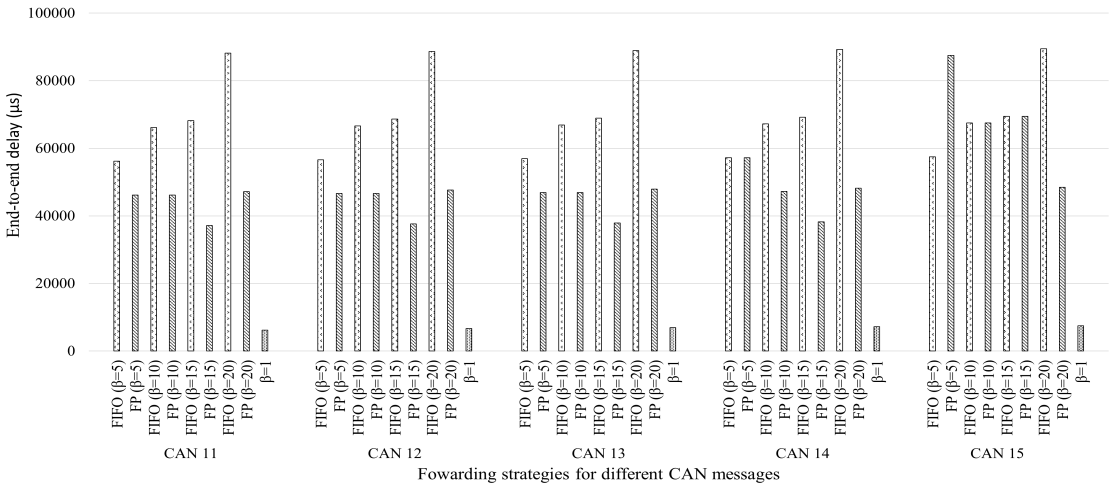


Fig. 7: Evaluation results for inter-domain CAN frames 11-15.

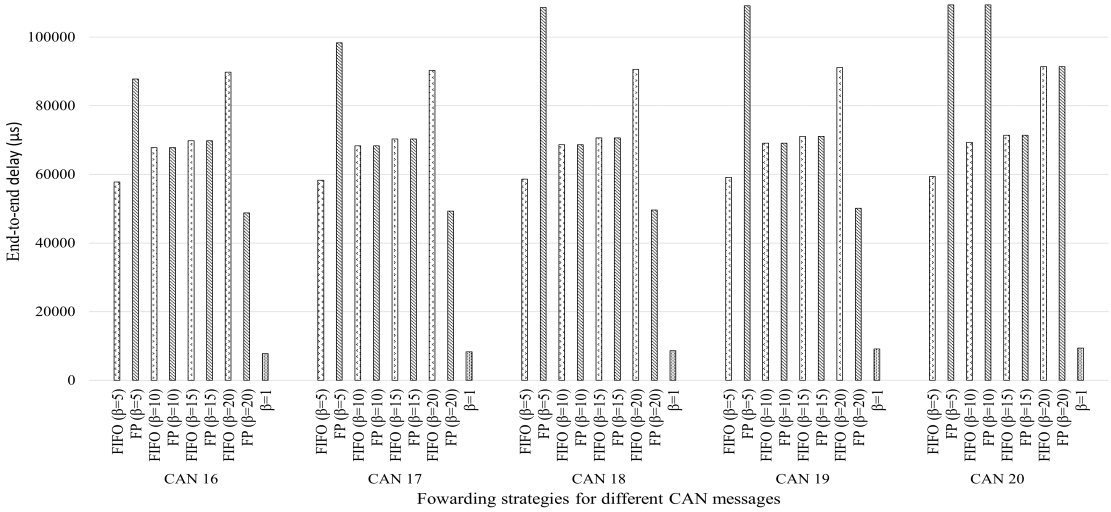


Fig. 8: Evaluation results for inter-domain CAN frames 16-20.

in the same TSN frame. In these cases, we recommend using lower values of β because CAN frames will experience less delay. As for the gateway forwarding technique, it depends on whether the system requires high-priority CAN frames to be transmitted as soon as possible while low-priority frames can tolerate high delay. If this is the case, then the FP gateway forwarding technique is recommended. Otherwise, the FIFO gateway forwarding technique is recommended because of its simplicity as long as the system can tolerate delays for high-priority CAN frames.

VI. Conclusion

In this paper, we proposed a CAN-TSN gateway, on which we evaluated the existing gateway forwarding techniques. The techniques being evaluated include First-in-first-out (FIFO), Fixed priority (FP), and one-to-one technique. We have developed end-to-end delay analysis methods for the forwarding techniques. We performed a set of experiments on a common multi-domain network architecture by generating synthetic traffic. Through these experiments and applying the delay analysis methods we showed that the common intuitive approach of encapsulating multiple CAN frames into one Ethernet frame is not necessarily an efficient solution when the gateway is connecting a CAN domain to a TSN domain. We showed that in high speed TSN networks it is better to use a one-to-one forwarding technique in the gateway rather than complex queuing techniques, both from the end-to-end delay and TSN bandwidth utilization perspective. This finding will have a significant impact in developing CAN-TSN gateways as it will reduce the complexity of design significantly.

VII. Acknowledgment

This work is supported by the Swedish Governmental Agency for Innovation Systems (VINNOVA) via the INTER-

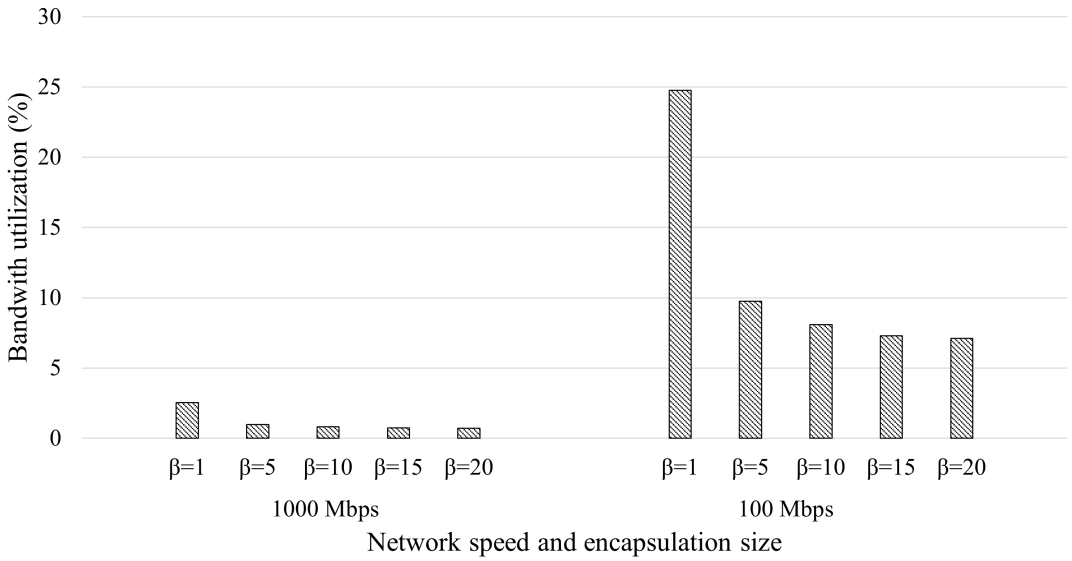


Fig. 9: Evaluation of TSN network bandwidth utilization for the inter-domain CAN frames.

CONNECT, PROVIDENT and DESTINE projects and by the Swedish Knowledge Foundation via the projects DPAC & HERO. We would like to thank all industrial partners, in particular HIAB, Arcticus Systems and Volvo Construction Equipment.

REFERENCES

[1] IEEE Time-Sensitive Networking (TSN) Task Group, 2016, http://www.ieee802.org/1/pages/tsn.html.

[2] Mohammad Ashjaei, Lucia Lo Bello, Masoud Daneshtalab, Gaetano Patti, Sergio Saponara, and Saad Mubeen. Time-sensitive networking in automotive embedded systems: State of the art and research opportunities. *J. Syst. Archit.*, 117(C), aug 2021.

[3] Lucia Lo Bello, Mohammad Ashjaei, Gaetano Patti, and Moris Behnam. Schedulability analysis of time-sensitive networks with scheduled traffic and preemption support. *Journal of Parallel and Distributed Computing*, 144:153–171, 2020.

[4] Unmesh D Bordoloi, Amir Aminifar, Petru Eles, and Zebo Peng. Schedulability analysis of ethernet avb switches. In *2014 IEEE 20th International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 1–10. IEEE, 2014.

[5] Robert I Davis, Alan Burns, Reinder J Bril, and Johan J Lukkien. Controller area network (can) schedulability analysis: Refuted, revisited and revised. *Real-Time Systems*, 35:239–272, 2007.

[6] Peter Hank, Steffen Müller, Ovidiu Vermesan, and Jeroen Van Den Keybus. Automotive ethernet: in-vehicle networking and smart mobility. In *2013 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1735–1739. IEEE, 2013.

[7] Christian Herber, Andre Richter, Thomas Wild, and Andreas Herkersdorf. Real-time capable can to avb ethernet gateway using frame aggregation and scheduling. In *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 61–66. IEEE, 2015.

[8] CAN in Automation (CiA) e.V. Cia draft specification 610-1 can xl specifications and test plans — part 1: Data link layer and physical coding sub-layer requirements — version 1.0.0, March 2022.

[9] Andreas Kern, Dominik Reinhard, Thilo Streichert, and Jürgen Teich. Gateway strategies for embedding of automotive can-frames into ethernet-packets and vice versa. In *International Conference on Architecture of Computing Systems*, pages 259–270. Springer, 2011.

[10] Hyung-Taek Lim, Lars Völker, and Daniel Herrscher. Challenges in a future ip/ethernet-based in-car network for real-time applications. In *Proceedings of the 48th Design Automation Conference*, pages 7–12, 2011.

[11] L. Lo Bello, R. Mariani, S. Mubeen, and S. Saponara. Recent advances and trends in on-board embedded and networked automotive systems. *IEEE Transactions on Industrial Informatics*, 15(2), 2019.

[12] Saad Mubeen, Jukka Mäki-Turja, and Mikael Sjödin. Worst-case response-time analysis for mixed messages with offsets in controller area network. In *Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation (ETFA)*, pages 1–10. IEEE, 2012.

[13] Saad Mubeen, Jukka Mäki-Turja, and Mikael Sjödin. Extending worst case response-time analysis for mixed messages in controller area network with priority and fifo queues. *IEEE ACCESS*, 2:365–380, 2014.

[14] Saad Mubeen, Jukka Mäm-Turja, and Mikael Sjödin. Extending schedulability analysis of controller area network (can) for mixed (periodic/sporadic) messages. In *ETFA*, pages 1–10. IEEE, 2011.

[15] Saad Mubeen, Jukka Mäki-Turja, and Mikael Sjödin. Integrating mixed transmission and practical limitations with the worst-case response-time analysis for controller area network. *Journal of Systems and Software*, 99:66–84, 2015.

[16] Abdelaziz Ahmed Nacer, Katia Jaffres-Runser, Jean-Luc Scharbarg, and Christian Fraboul. Strategies for the interconnection of can buses through an ethernet switch. In *2013 8th IEEE International Symposium on Industrial Embedded Systems (SIES)*, pages 77–80. IEEE, 2013.

[17] J-L Scharbarg, Marc Boyer, and Christian Fraboul. Can-ethernet architectures for real-time applications. In *2005 IEEE Conference on Emerging Technologies and Factory Automation*, volume 2, pages 8–pp. IEEE, 2005.

[18] ISO Standard. 11898: Road vehicles—interchange of digital information—controller area network (can) for high-speed communication. *International Standards Organization, Switzerland*, 1993.

[19] Daniel Thiele, Johannes Schlatow, Philip Axer, and Rolf Ernst. Formal timing analysis of can-to-ethernet gateway strategies in automotive networks. *Real-time systems*, 52(1):88–112, 2016.

[20] Ken Tindell, Alan Burns, and Andy J Wellings. Calculating controller area network (can) message response times. *Control engineering practice*, 3(8):1163–1169, 1995.

[21] Luxi Zhao, Paul Pop, and Silviu S Craciunas. Worst-case latency analysis for ieee 802.1 qbv time sensitive networks using network calculus. *IEEE ACCESS*, 6:41803–41815, 2018.

[22] Helge Zinner, Josef Noebauer, Thomas Gallner, Jochen Seitz, and Thomas Waas. Application and realization of gateways between conventional automotive and ip/ethernet-based networks. In *Proceedings of the 48th Design Automation Conference*, pages 1–6, 2011.