A Zone-based Optical Architecture for Intra-Vehicle Backbone Networks

Onur ALPARSLAN[†], Shin'ichi ARAKAWA[†], and Masayuki MURATA[†]

[†] Graduate School of Information Science and Technology, Osaka University
1–5 Yamadaoka, Suita, Osaka 565–0871, Japan

Abstract Most of the traditional intra-vehicle data transmission architectures have a low transmission capacity, so they are not fast enough for carrying large amount of data like high resolution video. Ethernet is becoming popular in intra-vehicle networks as it can achieve high data transfer rates. However, ultra high speed Ethernet may have a high cost and high complexity. Recently, we presented an alternative cut-through switching optical backbone network architecture that can achieve high data rates with low end-to-end delays at a possibly cheaper cost. It was based on a zonal architecture for centralized the processing and decreasing the costs. In this paper, we present an improved scheduling algorithm for our intra-vehicle architecture. We show the new scheduling algorithm can further decrease the end-to-end delays.

Key words Vehicle backbone, In-vehicle networks, Quality of service, Ethernet, Optical networks

1. Introduction

Due to the competition among the manufacturers and the higher expectations of consumers, the vehicles are getting more and more advanced features. New features are usually added to the vehicles by mounting new embedded systems called electronic control unit (ECU). The number of ECUs in today's vehicles have already passed 150 and it keeps increasing [1]. However, adding new features and increasing the number of ECUs increase the number of data ports and the amount of data to be carried in the backbone of the vehicle. For example, high resolution entertainment systems are becoming popular. Each seat may have a separate entertainment system and a media server in the car may stream high resolution videos to the seats. Moreover, new wireless high-speed connection technologies like 5G allow watching high resolution video from online streaming services like Youtube or playing online games with high resolution video [2, 3]. Increasing the resolution and number of entertainment systems greatly increase the data transmitted in the backbone. Furthermore, new advanced driver-assistance systems (ADAS), which increase car and road safety, and high level autonomous systems like self-driving cars need lots of data from the sensors [4]. These features require carrying large amount of data with low latency and strict Quality of service (QoS) in the backbone.

Most vehicles use traditional CAN, LIN, MOST and Flex Ray intra-vehicle bus architectures, which have a low data rate that cannot satisfy the high bandwidth requirements of high resolution video sensors and infotainment systems. Ethernet is commonly proposed as a candidate for the next generation intra-vehicle backbone networks, because it can achieve high data transfer rates. As the traditional Ethernet does not satisfy QoS, resilience, cost requirements of future intra-vehicle networks, Time-Sensitive Networking (TSN) and Audio Video Bridging (AVB) [5] Ethernet standards are being proposed and adapted to automotive networks for satisfying these requirements. However, the new standards have high complexity that can greatly increase the cost at high link transmission rates.

Network topology is another important factor when satisfying the latency requirements. If the topology is not adequate, it may not be possible to achieve low latency even with a very high speed link layer technology. Previously, a gateway architecture, where few number of gateways are used for only data switching between ECUs, was commonly used. However, the industry started to shift to a domain based architecture, where related ECUs are connected to the same domain controller. The domain controller controls the operation of ECUs to increase the efficiency and decrease the costs. However, the related ECUs may be far away from each other, which increases the wiring length. Using longer wires increase the cost and the weight of the car. Now, there are plans to use a zonal architecture, where ECUs are connected to nearby zonal gateways based on the spatial distance, which can greatly decrease the wiring. The ECUs will be controlled by a limited number centralized processors, so the zonal gateways do not need a high processing capacity, which can further decrease the cost. However, the backbone of a zonal architecture should have a high transmission capacity and a low latency for receiving and processing large amount of data from ECUs with strict deadlines.

Previously, we presented a new zone-based optical ring-based intra-vehicle architecture [6]. We showed that it can achieve low end-to-end delays. In this paper, we present an improved scheduling algorithm that can further decrease the end-to-end delays in our architecture.

The remainder of this chapter is organized as follows. Section 2. introduces the intra-vehicle network topologies, the proposed network architecture and the slot scheduling algorithm, Section 3. describes the simulation scenario and presents the simulation results, Section 4. describes the conclusions and future works.

2. Intra-vehicle Networking

2.1 Network Topologies

There are multiple bus architectures like CAN, LIN, MOST and Flex Ray currently used in intra-vehicle networks. As these architectures cannot be connected to each other directly, the initial intra-vehicle networks with multiple bus architectures were based on a gateway architecture where ECUs using the same bus technology were grouped and connected to the same gateway. The gateways converted the data to other bus architectures, so ECUs with different bus technologies can communicate with each other. When there are few number of gateways, the wiring becomes costly because the ECUs using the same bus architecture may be far away from each other [7]. Moreover, many ECUs have overlapping functionalities and sensors, which increase the cost, but the gateways usually do only signal conversion without controlling and optimizing the ECUs.

As a solution, a domain-based architecture, where related ECUs are connected to the same domain controller, is becoming popular [8]. The domain controller controls the ECUs and does the processing of the data from the connected ECUs so that overlapping functionalities and sensors in ECUs can be avoided. This can decrease the cost and complexity of vehicles compared to the gateway architecture.

However, the domain based architecture may not be decrease the wiring cost, because the related ECUs may be far away from each other. As a solution, a zone-based architecture is being proposed by the industry [8]. In the zone-based architecture, zone gateways are placed at the different areas of the vehicle, away from each other. The ECUs are connected to the closest zone gateway independent of the bus architecture or the functionality of the ECU. Unlike the domain-based architecture, where the data of a ECU is processed by the domain controller that the ECUs is connected to, in the zone-based architecture the data from ECUs are processed by few number of centralized processors. Therefore, the number of processors and related costs in the vehicle are further decreased compared to a domain-based architecture. On the other hand, the zone-based architecture requires a high capacity and low latency backbone network to carry all the data from the ECUs and process them in centralized processors.

2.2 Optical backbone network

Previously, we proposed an optical cut-through intra-vehicle backbone architecture called Si-based In-vehicle Photonic Network (SIPhoN). The architecture is composed of a single master node and multiple zonal gateway nodes. The nodes in SIPhoN are connected by an unidirectional optical ring topology. The fibers in the optical architecture carry multiple channels. A single control channel is dedicated to the control packets and one or more channels are dedicated to the data packets. It is a slot-based transmission architecture where each slot in a data channel carries a packet to or from a gateway node in the backbone ring topology as in Fig. 1. The assignment of slots to the gateway nodes is done by the master node according to a slot scheduling algorithm. The master node informs the assigned slots to the gateway nodes by sending a control packet on the control channel before sending the associated data slot. The control and data channels apply cut-through switching, so the arrival time difference between a control packet and the associated data slot is fixed throughout the ring topology and equals to the duration of a guard band between the data slots, which is necessary for the reconfiguration of the optical switching fabric.

The types of actions defined in control packets are as follows:

- Listen: The zonal gateway node selected by the master node may receive data from the master node in the associated data slot.
- Talk: The zonal gateway node selected by the master node may send data to the master node in the associated data slot.
- Idle: The associated data slot is not used (empty).

As the laser diodes in an optical network have limited lifetime, the cost of replacing the broken laser diodes (LD) is an important cost factor. In SIPhoN architecture, only the master node carries the laser diodes for generating the optical and data slots, which decreases the long-term cost of the backbone architecture. The gateway nodes read and update the information in the data channel by using modulation and detection optical circuits (MD), so they do not need laser diodes. The gateway nodes receive the control information on the control channel by getting a copy of the optical control packet via an optical coupler. In case of a Listen or Talk action defined for this gateway node in the control packet, the gateway node updates its optical switch configuration during the guard band time for forwarding the upcoming data slot to its modulation and detection optical circuit.

An example of processing the incoming and outgoing control packets and the data slots in gateway 1 is shown in Fig. 1. The incoming control and data packets arrive to the gateway node from the left side of the figure and depart the gateway as shown in the right side of the figure. The slots carrying the data packets are shown in green, while the empty slots are shown in white. In Fig. 1, the first control packet entering gateway 1 is a Listen packet for gateway 1. It means that the associated data slot contains a frame containing a packet for this gateway sent from the master node. The gateway node updates its switching fabric, forwards the data slot to its MD circuit and extracts the frame in the slot by the MD circuit. Then, it decapsulates and forwards the data packet to the destination ECU. The second control packet is a Listen packet for gateway 4. As the destination of the data slot is another gateway node, the gateway 1 does not need to process this data slot, so it forwards the incoming data slot directly to its output backbone link. The third control packet entering the gateway 1 is a Talk packet for gateway 1, so gateway 1 can use the upcoming data slot for sending data to the master node. Gateway 1 reconfigures its switching fabric and in case there is a packet in its queue it encapsulates the packet in a frame and injects to the upcoming data slot by using the MD cir-

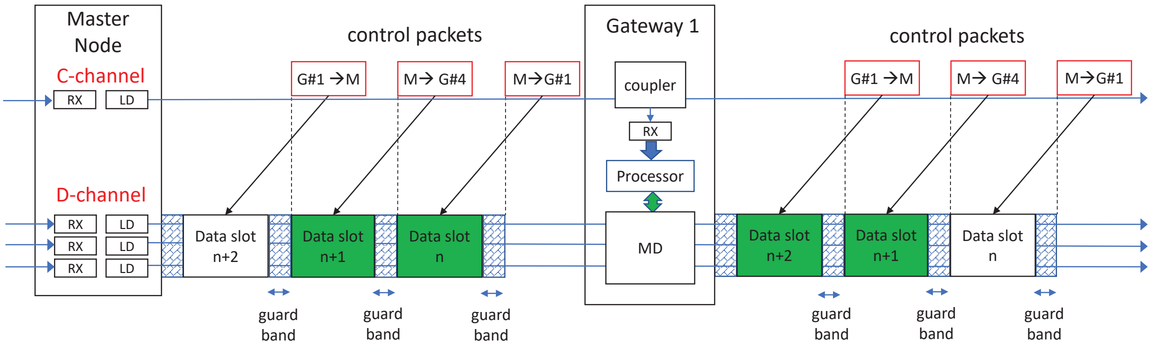

Fig. 1　An example of incoming and outgoing control packets and data slots in gateway 1.

cuit. Master node does the scheduling of slots and forwarding the data packets, so in case a gateway node wants to send a packet to another gateway node, it first sends the packet to the master node. Later, the master node forwards the packet to the destination gateway node.

As the optical fibers have the advantage of being resilient against environmental stress factors over copper wires, Multi-Gigabit Automotive Optical PHYs (OMEGA) Study Group has been established to standardize Ethernet networks with fiber links in intra-vehicle networks [9]. One important difference with SIPhoN is that OMEGA would require laser diodes in the gateways for sending data from the gateways. Laser diodes have a failure time of around 10^6 hours under typical conditions [10]. Using many laser diodes in the backbone network can considerably increase the failure rate and the repair costs.

2. 3　Slot Scheduling

In [6], for slot scheduling we used a fixed pattern of slot assignment like {Listen, Talk GW1, Talk GW2, Talk GW3, Talk GW4, Talk GW5, Talk GW3} that repeats in periodic cycles. The pattern was chosen based on the average amount of traffic sent to the backbone by the gateways. However, using a fixed pattern can increase the latency of high priority packets. Moreover, many flows like video flows, audio flows, control flows generate packets with a deterministic pattern. The latency of these flows can be decreased by using a clever scheduling algorithm making use of the pattern information.

In this paper, we propose a new slot scheduling algorithm for SIPhoN. The algorithm tries to the minimize the packet delays of flows that have a traffic generation pattern. The packet departures do not have to be exactly periodic. It is enough to know the expected departure time of the next packet of the flow at the departure time of a packet.

The new slot scheduling algorithm works as follows. The ECUs add a SIPhoN packet header to the packets of their deterministic flows for sending time information to the master node. When a packet with this information in the SIPhoN packet header arrives to the master node, the master node estimates the arrival time of the next packet of this flow to the gateway that the ECU is connected to. Then the master node can schedule a slot that will arrive to that gateway at the same time or a short time after the packet from the ECU arrives to the gateway. As a result, the waiting time of the packets in the gateway buffers for a data slot can be decreased.

When an ECU connected to a gateway sends a packet of a deterministic flow, the slot scheduling for the next packet is done as follows:

（1） The ECU writes the time difference between the departure time of the current packet and the expected departure time of the next packet to the SIPhoN packet header. Let's denote this time difference as D_{next}. In case there are multiple deterministic flows from the same ECU, the ECU also takes the queuing delays due to the contentions on its output link into account when calculating the departure time of the next packet.

（2） The current packet arrives to the gateway from the ECU. The gateway stores the arrival time of the packet to the gateway until the packet leaves. When the packet gets a data slot in the backbone and leaves the gateway, the gateway writes the time difference between the arrival and departure time of the packet to the SIPhoN packet header. This time difference shows how long the packet waited in the gateway buffer. Let's denote this time difference as G_{wait}. Moreover, the gateway writes the current buffer occupancy information to the SIPhoN packet header of the packet regardless of the packet is from a deterministic flow or not. Then the gateway encapsulates the packet in a frame and sends to the master node over the backbone by injecting it into a data slot.

（3） The frame arrives to the master node in a data slot. The master node decapsulates the packet from the frame and reads the SIPhoN packet headers and extracts the time difference information D_{next} and G_{wait}. The master node already knows the propagation delay in the backbone, which can be calculated by the time difference between the departure time of the data slots from the master node and their return time to the master node after a doing loop in the ring backbone.

（4） The master node calculates when the control slot associated with the arrived data slot departed from the master node by subtracting the sum of propagation delay in the backbone ring, interval between sending control packets and the duration of a control packet from the current time. Let's denote this variable as T_{prev}.

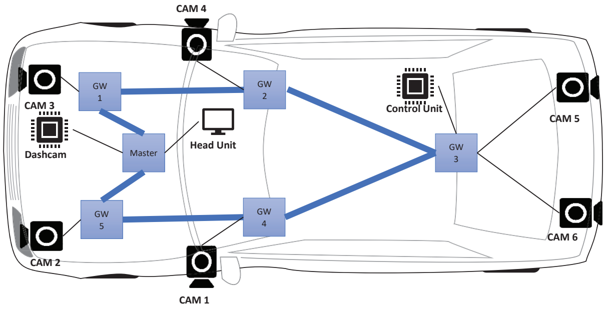

Fig. 2　Simulated intra-car network topology.

Then, the master node calculates when it should send the control packet for the next packet, which is denoted by T_{next}, by the formula $T_{next} = T_{prev} + D_{next} - G_{wait}$. As the architecture is slot-based, it may not be possible to schedule data slot exactly at T_{next}. Moreover, the optimum slot may have been reserved for another packet. In that case, the master node reserves the first empty slot after the time T_{next}.

In case an ECU is connected to the master node and the ECU wants to send a packet of a deterministic flow to an ECU connected to a gateway node, the slot reservation for the next packet is done as follows:

（1） The ECU writes the time difference between the departure time of the current packet and the expected departure time of the next packet to the SIPhoN packet header. Let's denote this time difference as D_{next}.

（2） When the current packet arrives to the master node, the master node reads the SIPhoN packet header and calculates when the next packet of this flow will arrive to the master node by simply adding D_{next} to the current time. Then it reserves the first empty slot that will minimize the waiting time of the next packet in the buffer.

At the beginning of each slot, the master node assigns the slot to a gateway as follows:

（1） The master node keeps a list of future slot reservations for the upcoming packets of deterministic flows. The reservations in this slot scheduling list have the highest priority. Each time before sending a slot, the master node first checks if there is a reservation for this slot. If there is a reservation, the master node assigns the slot to reserved gateway.

（2） If there is no reservation for the slot, the master node checks the buffer occupancy information of the gateways that it received in the SIPhoN headers of packets. It assigns the slot to the gateway with the biggest buffer occupancy. This is for carrying the data packets of non-deterministic flows like Internet traffic.

（3） If all the gateway buffers look empty, the master node checks the last time it assigned a slot to each gateway. It assigns the new slot to the gateway that the longest time has elapsed since the last slot assignment to this gateway.

3.　Simulation

We simulated the intra-vehicle network shown in Fig. 2 with a traffic matrix inspired from [11]. By using a simulator that we implemented on OMNeT++ framework, we simulated a traffic matrix with 11 flows (7 video flows and 4 control flows). The traffic sources are as follows:

（1） Six video cameras (CAM) send uncompressed 4K30p smooth video traffic (6 Gbps) to Dashcam.

（2） The Dashcam sends uncompressed 4K30p smooth video traffic (6 Gbps) to Head Unit.

（3） The Control unit sends warnings to the Dashcam. This control flow sends 46 Bytes data packets every 0.5ms.

（4） The Dashcam unit sends warnings to the Control Unit. This control flow sends 46 Bytes data packets every 0.5ms.

（5） The Head Unit sends real-time control messages to the Control Unit. This control flow sends 46 Bytes data packets every 0.5ms.

（6） The Control Unit sends real-time control messages to the monitor of Head Unit. This control flow sends 46 Bytes data packets every 0.5ms.

The simulation parameters are as follows. The optical backbone links carry a 100 Gbps data channel and a 1.25 Gbps control channel. The propagation delay in all links is 20ns. The links in the backbone are unidirectional and the transfer is in the clockwise direction. There is a 100ns guard band between the data slots. To support jumbo frames without fragmentation, we used both a slot capacity of and MTU of 9000 Bytes. As the maximum slot size is 9000 Bytes, we also set the video packet size to 9000 Bytes to fully use the slot capacity. The packets of video flows are paced at the source to prevent burstiness.

We simulated two slot scheduling algorithms and compared their results. In the first simulation, the slot scheduling algorithm applies a fixed and periodic set of 7 slots as {Listen, Talk GW1, Talk GW2, Talk GW3, Talk GW4, Talk GW5, Talk GW3} that repeats in cycles. The first Listen slot allows the master node to send data to one of the gateways nodes in case there is a packet destined to one of the ECUs connected to the gateways. The other Talk slots are used for transferring data from the indicated the gateways to the master node. As there are two video cameras connected to the gateway 3, two slots are assigned to transfer data from this gateway, which effectively makes the assigned capacity to this gateway twice the other gateways. In the second simulation, the slot scheduling is based on the dynamic algorithm introduced in this paper. It assigns the slots to the future packets of deterministic flows based on the time information provided by the packets.

We evaluated and compared the end-to-end packet delay of deterministic control flows and video flows when the fixed and dynamic scheduling algorithms are used.

Fig. 3 shows the end-to-end delay of control packets from Control Unit to Head Unit when fixed slot scheduling was used. The x-axis is the arrival time of the packet to the destination in terms of seconds. The y-axis is the end-to-end delay of the packets in terms of μs. Fig. 3 shows that most of the packet delays were between 0.8μs and 3.9μs. The base delay due to propagation and transmission of control packets was 0.8μs, so the delay caused by the

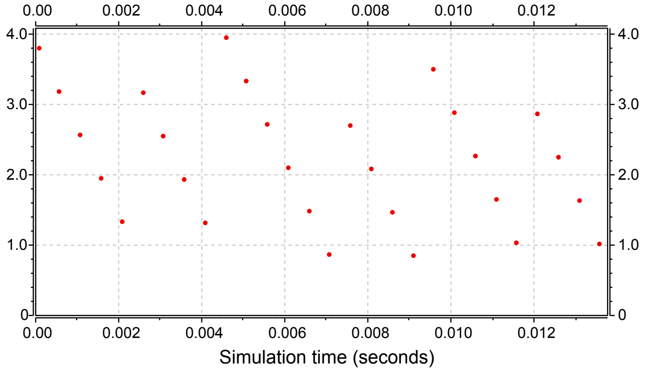

Fig. 3 Delay of control packets from Control Unit to Head Unit with fixed scheduling.

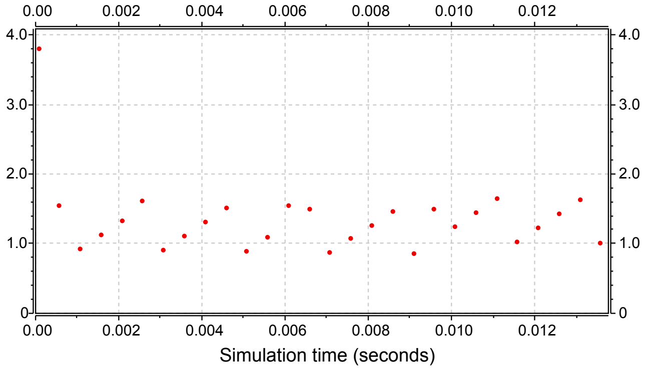

Fig. 4 Delay of control packets from Control Unit to Head Unit with dynamic scheduling.

slotted architecture, contentions and the fixed scheduling algorithm was between 0μs and 3.1μs. As a comparison, Fig. 4 shows the end-to-end delay of control packets from Control Unit to Head Unit when fixed scheduling was used. In the simulation results, most of the packet delays were between 0.8μs and 1.6μs. After subtracting 0.8μs, which is sum of propagation and transmission delays of control packets, the delay caused by the slotted architecture, contentions and the dynamic scheduling algorithm was between 0μs and 0.8μs. As a result, by using the dynamic scheduling, the maximum total end-to-end delay decreased to around half and the maximum delay due to the scheduling algorithm and the architecture decreased by around three times.

Fig. 5 shows the end-to-end delay of control packets from CAM 5 to Dashcam when fixed slot scheduling was used. Most of the packet delays were between 2.2μs and 5.4μs. The base delay due to propagation and transmission of video packets was 2.2μs, so the delay caused by the slotted architecture, contentions and the fixed scheduling algorithm was between 0μs and 3.2μs. As a comparison, Fig. 6 shows the end-to-end delay of control packets from CAM 5 to Head Unit when fixed scheduling was used. In the simulation results, most of the packet delays were between 2.2μs and 3.0μs. After subtracting 2.2μs, which is sum of propagation and transmission delays of control packets, the delay caused by the slotted architecture, contentions and the dynamic scheduling algorithm was between 0μs and 0.8μs. As a result, by using the dynamic scheduling, the maximum total end-to-end delay decreased to around the half and the maximum delay due to the scheduling algorithm and the architecture decreased by around four times. An in-depth examination also showed that when dynamic scheduling is used, unless there is a slot contention, the packets of deterministic flows from gateway 3 wait at most one slot time in the gateway buffers, while they may need wait around 3 or 4 slots when fixed scheduling is used.

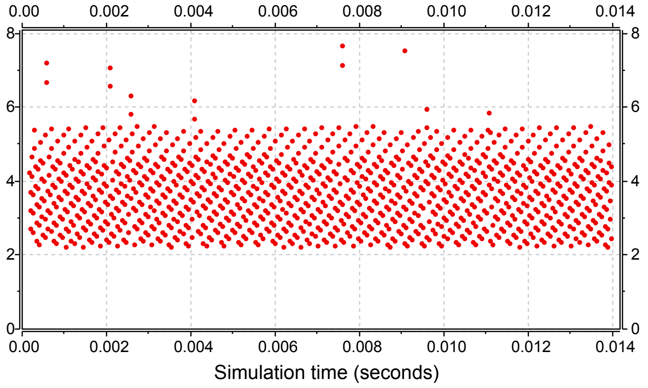

Fig. 5 Delay of control packets from CAM 5 to Dashcam with fixed scheduling.

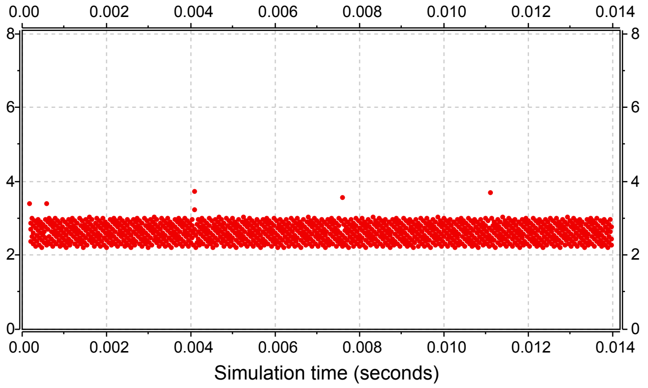

Fig. 6 Delay of control packets from CAM 5 to Dashcam with dynamic scheduling.

4. Conclusion

Recently, we presented an alternative cut-through switching optical backbone network architecture for intra-vehicle networks. In this paper, we presented an improved slot scheduling algorithm for this architecture. By a simulation study, we showed that the proposed algorithm decreases the end-to-end packet delays of flows that have a deterministic traffic generation pattern, compared to a fixed slot scheduling pattern. As a future work, we will compare the delay of the proposed architecture with TSN+AVB Ethernet. Moreover, we will extend the architecture by carrying multiple packets to different destinations in parallel by using subframes in the same slot by wavelength switching.

Acknowledgment

This work is partly supported by the National Institute of Information and Communications Technology (NICT).

References

[1] M. Rumez, D. Grimm, R. Kriesten, and E. Sax, “An overview of automotive service-oriented architectures and implications for security countermeasures,” *IEEE Access*, vol. 8, pp. 221852–221870, 2020.

[2] C. R. Storck and F. Duarte-Figueiredo, “A survey of 5G technology evolution, standards, and infrastructure associated with vehicle-to-everything communications by Internet of vehicles,” *IEEE Access*, vol. 8, pp. 117593–117614, 2020.

[3] D. Kombate and Wanglina, “The Internet of vehicles based on 5G communications,” in *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 445–448, 2016.

[4] M. N. Ahangar, Q. Z. Ahmed, F. A. Khan, and M. Hafeez, “A survey of autonomous vehicles: Enabling communication technologies and challenges,” *Sensors*, vol. 21, no. 3, 2021.

[5] “IEEE standard for local and metropolitan area networks–audio video bridging (AVB) systems,” *IEEE Std 802.1BA-2011*, pp. 1–45, 2011.

[6] O. Alparslan, S. Arakawa, and M. Murata, “Next generation intra-vehicle backbone network architectures,” in *Proceedings of the IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*, pp. 1–7, IEEE, 2021.

[7] J. Huang, M. Zhao, Y. Zhou, and C. Xing, “In-vehicle networking: Protocols, challenges, and solutions,” *IEEE Network*, vol. 33, no. 1, pp. 92–98, 2019.

[8] D. Pannell, L. Chen, J. Dorr, W. Lo, D. Pannell, M. Potts, H. Zinner, and A. Zu, “Use cases,” *IEEE P802.1DG V0.4*, pp. 1–47, 2019.

[9] “IEEE 802.3 multi gigabit automotive optical PHY study group,” 2020.

[10] Y. Deshayes, F. Verdier, L. Bechou, B. Tregon, Y. Danto, D. Laffitte, and J. Goudard, “Estimation of lifetime distributions on 1550-nm DFB laser diodes using Monte-Carlo statistic computations,” *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 5465, 09 2004.

[11] G. Patti and L. L. Bello, “Performance assessment of the IEEE 802.1Q in automotive applications,” in *2019 AEIT International Conference of Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*, pp. 1–6, 2019.