Asynchronous Time-Sensitive Networking (TSN) Implementation in Automotive Zone-Based Architecture

Abderrahmane Boulahdour
Université du Québec à Trois-Rivières (UQTR)
Québec, Canada
abderrahmane.boulahdour@uqtr.ca

Miloud Bagaa
Université du Québec à Trois-Rivières (UQTR)
Québec, Canada
miloud.bagaa@uqtr.ca

Ahmed Ouameur Messaoud
Université du Québec à Trois-Rivières (UQTR)
Québec, Canada
messaoud.ahmed.ouameur@uqtr.ca

Oussama Bekkouche
Aalto University
Espoo, FINLAND
oussama.bekkouche@aalto.fi

Adlen Ksentini
Campus SophiaTech, EURECOM
Biot, France
adlen.ksentini@eurecom.fr

***Abstract*—In the realm of modern automotive technology, vehicles now incorporate a range of advanced features, such as Advanced Driver Assistance Systems (ADAS), infotainment systems, and autonomous driving technologies. In response to the complexities of wiring and the imperative for improved network performance, a shift is occurring from conventional domain-based In-Vehicle Network (IVN) architectures to zone-based configurations. Similarly, there's a transition from traditional IVN protocols to Ethernet-based alternatives. Additionally, the adoption of Time-Sensitive Networking (TSN) is increasing, allowing for deterministic data transmission via Ethernet networks. The combination of TSN with zone-based architectures holds the potential for optimizing data transmission and reducing End-to-End (E2E) delay its variation (Jitter), hence ensuring the desired service level agreement (SLA) and enhancing the quality of experience (QoE). Our contribution involves simulating an IVN architecture where diverse Electronic Control Units (ECUs) are segregated into zones and interconnected via Ethernet. Our work includes implementing Asynchronous Traffic Shaping (ATS) standard in TSN for improved experiments with different ATS parameters and analyze their impact on E2E delay and jitter. The results demonstrate the efficiency of TSN, specifically ATS, in the IVN environment.**

I. INTRODUCTION

The autonomous vehicles have triggered a transformative wave in the automotive industry, with a significant enhancement of IVN protocols like Controller Area Network (CAN), Local Interconnect Network (LIN), FlexRay, and Media Oriented Systems Transport (MOST) [1]. These protocols serve as the backbone for facilitating communication between the multitude of ECUs dispersed throughout autonomous vehicles. As the complexity of autonomous systems continues to evolve [2], traditional IVN protocols face challenges in meeting the diverse data transmission requirements and stringent latency constraints demanded by ADAS and infotainment systems.

To address these challenges, there has been a notable shift towards Ethernet-based IVN protocols. Ethernet can support a wide bandwidth of 100 Mbit/s to 50 Gbit/s [3] and low data transmission delay especially when integrated with TSN [4], [5]. TSN is a collection of features in IEEE 802.1 standards that enable deterministic and time-sensitive communication over standard Ethernet networks. One protocol among IEEE 802.1 standards is Audio Video Bridging (AVB) that has gained prominence for its ability to provide high bandwidth, low latency, and compatibility with Internet protocols (IP). This integration with TSN not only ensures the efficient transmission of time-sensitive data but also enables seamless communication between ECUs that are critical for the reliable operation of autonomous vehicles.

TSN supports both synchronous and asynchronous communication based on application needs. In synchronous communication, parties operate in a coordinated manner with time synchronization, achieved through standards like IEEE 802.1AS [6] that utilizes Precision Time Protocol (PTP) for $\leq 1us$ jitter. IEEE 802.1Qbv [7] introduces time-aware shaping for scheduled transmission at specific intervals, while IEEE 802.1Qch [8] enables deterministic forwarding in Ethernet through cycle-based scheduling, minimizing delay and ensuring low latency and jitter for real-time traffic.

In addition, IEEE 802.1Qcr introduces ATS, a solution for managing diverse traffic in IVNs. ATS categorizes traffic into urgent and non-urgent types. Urgent traffic receives priority treatment using queuing and stream-reshaping techniques, ensuring swift handling and optimized bandwidth usage. ATS integrates seamlessly with existing network hardware, enhancing IVN efficiency by addressing latency concerns and facilitating timely information delivery.

In this paper, our primary contribution lies in establishing a zone-based IVN architecture with ATS and assessing its E2E delay. We have also evaluated the impact of different parameters, such as the rate, the payload size and transmission rate on E2E delay and jitter in TSN-enabled network.

II. Related Work

In recent years, the automotive industry has known significant research focused on advancing simulation modeling and performance evaluation of TSN and its applications in in-vehicle networks. These efforts have contributed valuable insights into this field. Chulsun Park and Sungkwon Park [9] delve into the Performance Evaluation of Zone-Based In-Vehicle Network Architecture for Autonomous Vehicles. Their research provides a comprehensive comparison with Domain-Based architectures, offering a thorough analysis of network performance within autonomous vehicular environments. Luo's research outlines a Design Methodology of Automotive Time-Sensitive Network Systems using OMNeT++ [10], Offering an in-depth exploration of the methodologies employed to develop effective automotive TSN systems [11].

Fang and Li offer Simulative Assessments of Credit-Based Shaping (CBS) defined in [12] and Asynchronous Traffic Shaping [13] within TSN. Their study enhances comprehension of traffic shaping techniques, comparing CBS, ATS, and Strict Priority Queuing (SPQ). Results indicate that under heavy network load, ATS performs better than CBS, particularly in ensuring real-time performance for aperiodic traffic [14]. Additionally, Nasrallah's study [15] compares IEEE 802.1 Time-Aware Shaper (TAS) and ATS performance in TSN environments, revealing TAS's consistent latency adherence and ATS's effectiveness with sporadic traffic but limitations under heavier periodic loads.

The synchronization aspect of TSN is not overlooked, as Lee et al. investigate the integration of IEEE 802.1AS synchronization mechanism with FlexRay synchronization to enhance synchronization performance in automotive heterogeneous networks [16]. Furthermore, Leonardi et al. have proposed a simulation of CQF (Credit-based Queueing Function) and infotainment systems to contribute to the broader understanding of TSN's performance implications within in-vehicle contexts [17].

III. Asynchronous Traffic Shaping (ATS)

A. ATS Per-stream classification and metering

In TSN networks, enhancing reliability involves utilizing flow classification and metering on frames received at a bridge port, which might have various transmission ports. Both bridge ports and end stations can utilize mechanisms like Per-Stream Filtering and Policing (PSFP), ATS filtering, and eligibility time assignment in accordance with [13]. These mechanisms are deployed as the final stage of the filtering process between the reception port and frame queuing, thereby bolstering network reliability.

When utilizing ATS, each frame is associated with a stream filter. If a matching stream filter is identified, it is used to process the frame; otherwise, it is queued for transmission. When the stream filter includes maximum Service Data Unit (SDU) size filtering, frames are processed and forwarded accordingly if they do not exceed the specified maximum SDU size. Stream filters specify a stream gate that is used

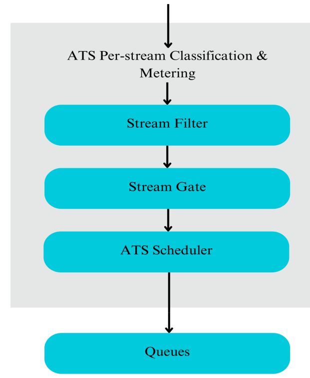

Fig. 1: Per-stream classification and metering at ATS switch [13]

in the frame processing. Frames might be discarded if they are received outside permitted intervals or if they surpass a data limit within an interval. Additionally, the frame's priority can also affect subsequent queuing decisions. ATS scheduler identified by the stream filter operates under the assumption that frames adhere to the associated $CommittedBurstSize$ parameter. It processes the frame by computing an eligibility time for subsequent use by the ATS transmission selection algorithm. However, if the computed eligibility time exceeds the maximum eligibility time, the frame may be discarded.

B. ATS transmission selection algorithm

In ATS transmission selection within queues, a frame becomes eligible for transmission if the queue holds one or more frames meeting the criteria for transmission. This determination is made by examining the frame's eligibility time and comparing it with the current time. The current time is determined by the specialized transmission selection clock within the system. Notably, this clock serves a dual function: Aside from indicating the current time, it also sets the selectability time for individual frames. This selectability time denotes when a frame is queued and prepared for transmission selection.

As frames reach their selectability time, they are selected for transmission in ascending order based on their assigned eligibility times. Frames with identical eligibility times follow the specified queue order during transmission selection. For frames with different eligibility times, the ATS scheduler guarantees compliance with the queue's required ordering by assigning eligibility times in a non-decreasing order.

The ATS scheduler state machine, guided by both ATS scheduler clocks used for determining frame arrival times and transmission selection clocks, operates under the $ProcessFrame$ (Algorithm 1) procedure and associated state variables. It updates crucial variables, such as bucket empty time and group eligibility time. These updates are influenced by parameters including $CommittedInformationRate$, $CommittedBurstSize$, $MaxResidenceTime$, frame $arrivalTime$, and total frame $length$ as defined in [13].

- **arrivalTime(frame):** This is the time when the ATS scheduler clock instance recognizes the arrival of the entire frame, in seconds.

Algorithm 1: Frame Processing Procedure defined in IEEE 802.1Qcr-2020 ATS [13]

procedure PROCESSFRAME(frame)
  $lengthRecoveryDuration \leftarrow length(frame)/CommittedInformationRate$
  $emptyToFullDuration \leftarrow CommittedBurstSize/CommittedInformationRate$
  $schedulerEligibilityTime \leftarrow BucketEmptyTime + lengthRecoveryDuration$
  $bucketFullTime \leftarrow BucketEmptyTime + emptyToFullDuration$
  $eligibilityTime \leftarrow \max(arrivalTime(frame), GroupEligibilityTime, schedulerEligibilityTime)$
  if $eligibilityTime \leq (arrivalTime(frame) + MaxResidenceTime/1.0e9)$:
    $GroupEligibilityTime \leftarrow eligibilityTime$
    $BucketEmptyTime \leftarrow (eligibilityTime < bucketFullTime)\ ?\ schedulerEligibilityTime : schedulerEligibilityTime + eligibilityTime - bucketFullTime$
    $AssignAndProceed(frame, eligibilityTime)$
  else:
    $Discard(frame)$
  end if
end procedure

- **AssignAndProceed(frame, eligibilityTime):** This procedure assigns a frame an eligibility time for processing based on the most recent time the ATS scheduler instance's token bucket was empty.
- **BucketEmptyTime:** This state variable records the most recent time at which the ATS scheduler's token bucket was empty, in seconds.
- **BucketFullTime:** This state variable records the time when the number of tokens in the bucket is equivalent to the $CommittedBurstSize$ parameter, in seconds.
- **CommittedBurstSize (CBS):** This is the maximum token capacity of the bucket, in bits.
- **CommittedInformationRate (CIR):** This parameter defines the rate at which the bucket is refilled until reaching its maximum capacity, in bits per second.
- **eligibilityTime:** The eligibility time of a frame without considering internal forwarding processing delays.
- **Discard:** This procedure discards the frame and increases the $DiscardedFramesCount$ counter of the associated reception port.
- **emptyToFullDuration:** The duration required to accumulate a number of tokens equivalent to the CBS parameter, in seconds.
- **GroupEligibilityTime:** This variable contains the recent eligibilityTime from the previous frame processed by any ATS scheduler instance in the same group, in seconds.
- **length(frame):** The length of the frame, in bits.
- **lengthRecoveryDuration:** The duration required to accumulate a number of tokens equivalent to length(frame), in seconds.
- **MaxResidenceTime** is a parameter limiting the duration for which frames can reside in a bridge.
- **schedulerEligibilityTime:** Time of bucket's token count equals the frame's arrival time, in seconds.

IV. AUTOMOTIVE ZONE-BASED ARCHITECTURE

Automotive zone-based architectures represent a paradigm shift in-vehicle electronic systems [18], moving away from the traditional centralized ECU model towards a decentralized approach that revolutionizes how functionalities are organized and managed. Departing from the monolithic structure of centralized ECUs, these architectures meticulously segment the vehicle's electronic functions into distinct zones, each empowered with its own dedicated set of ECUs and specialized tasks.

These zones encompass critical aspects, such as powertrain, chassis, body, infotainment, and connectivity, providing a comprehensive framework for the distribution of functions across the vehicle's ecosystem. At the core of this transformative architecture lies the zonal gateway, serving as a vital nexus for communication. Facilitating seamless data exchange and coordination among the various zones and ECUs, the zonal gateway ensures efficient operation and integration of the vehicle's electronic systems.

This architectural framework depends on resilient communication infrastructures, such as Ethernet or FlexRay to guarantee swift and dependable data transfer between ECUs and zones, consequently enhancing performance and responsiveness, thereby elevating the overall efficiency and effectiveness of vehicle electronics.

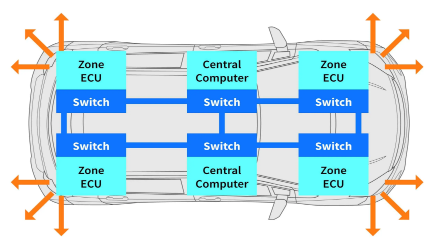

Fig. 2: Automotive Zone-Based Architecture - Toshiba [19]

Figure 2 illustrates a zone-based architecture where ECUs are grouped by physical proximity, each zone is equipped with its controller, while high-performance computing unit is centralized to facilitate computational tasks.

V. PROPOSED FRAMEWORK FOR A SIMULATION MODEL

A. ATS Simulation

For the ATS simulation, we have opted for OMNET++ [10] with INET [20] framework to leverage pre-existing modules available within this environment, providing us with a solid foundation to expedite our exploration of ATS implementation for experimental purposes. The TSN switch within INET framework encompasses a diverse range of modules, each contributing significantly to the intricacies of the simulated network. ATS functionality primarily resides in the bridging and outgoing interfaces layers, where it has been predominantly implemented.

Within the bridge layer, there is a stream decoder which responsible for decoding streams based on their PCP numbers. This decoding process occurs after the streams have been identified and encoded within TSN devices. Additionally, the encoder encodes the streams before they are transmitted to the subsequent TSN switch. In the `ingress` module of `streamFilter`, there are for each `i` stream two principal submodules: $meter[i]$ and $filter[i]$ tasked with specific functions related to metering and classification for ATS.

In the outgoing network interface of the TSN switch, traffic shaping occurs as streams pass through. This process involves queuing frames and sorting them based on their eligibility time upon arrival. Eligible frames that adhere to timing constraints are then transmitted.

B. Architecture

The proposed zone-based architecture comprises seven zones, each distinguished by TSN switches that serve as zone controllers. Within these zones, TSN devices emulate various ECUs, all interconnected via 100Mbit/s Ethernet channels (refer to Figure 3).

The ECUs generate frames with varying characteristics, such as priority (PCP), Ethernet payload size, data rate, and transmission interval, each corresponding to different traffic types (see Table III). These traffic types are grouped into three categories: Critical Data Traffic (CDT), Class A, and Class B, based on their priority levels for data transmission. CDT encompasses crucial data necessary for autonomous driving functionality, including control signals, LiDAR, chassis information, navigation data, V2X communication, radar generated millimeter-wave data, and wheel sensor data. In contrast, Class A and Class B traffic types consist of less critical data, such as data generated from ADAS to HUD, GPS images/data, and AVB data. The communication between the ECUs occurs via zonal TSN switches that represent the controller, facilitating efficient data exchange and management within the system.

To implement ATS per-stream filtering and metering, the process begins with identifying and encoding each stream, followed by applying per-stream traffic filtering and metering. Each traffic type corresponds to a stream and is associated with a specific class and PCP number based on its priority. Additionally, two parameters, CIR and CBS, are utilized in the ATS schedulers (meters) to calculate the eligibility time of frames. CIR denotes the data rate of the stream and determines the rate at which the token bucket is refilled with tokens to its maximum capacity CBS. Assuming that the bucket is filled four times per second, the values of the CBS parameter can be determined based on this assumption, as outlined in Table I.

TABLE I: Streams CIR and CBS Values

Class	**Stream**	**CIR** $[Mbps]$	**CBS** $[KBytes]$
CDT	control	0.064	3
	lidar	11	352
	navigation	0.2	6.4
	chassis	3.8	121.6
	v2x	0.26	8.32
	wheel	2.8	89.6
	millimeter-wave	0.52	16.64
Class A	audio	0.7	22.4
	hud	2.43	78
Class B	gps	33.3	1065.6
	video	40	1280
	fuel	0.3	9.6

All streams are configured with the same $MaxResidenceTime$ parameter, set at 5ms. Frames are dropped if the simulation time equals the eligibility time plus the maximum residence time. At the outgoing network interface of the TSN switches, traffic shaping occurs. Three queues are utilized per class to queue frames by their eligibility time, ensuring that they are not sent before their designated eligibility time.

VI. Simulation Results

This section outlines the results obtained from the simulation model developed in the previous section. Two simulations were conducted, each lasting for a duration of 5 seconds, strictly following the communication specifications outlined in Table III. In the first simulation, the same CIR and CBS values as defined in Table I were employed for all streams across all classes. In the second simulation, an attempt was made to increase the CIR values for the CDT class by multiplying the original values by 7, and for class A by 5 random constants, while retaining the default values for class B.

The principal aim was to evaluate the E2E delay and Jitter of frames transmitted by the ECUs, originating from various streams in both simulations. Table II provides a summary of the measured E2E delay^1, Jitter^1 and E2E delay^2, Jitter^2 for simulation 1 and 2, respectively.

TABLE II: E2E Delay $[ms]$ and Jitter $[us]$ Average - Simulation 1 and 2

Stream	**PCP**	**Delay**^{1}	**Delay**^{2}	**Jitter**^{1}	**Jitter**^{2}
control	7	4.59	4.24	1.51	0.71
lidar	7	0.17	0.16	0.02	0.02
navigation	7	5.42	0.11	2.03	0.03
chassis	7	4.25	0.14	1.19	0.02
v2x	7	5.93	0.07	2.06	0.02
wheel	7	3.21	0.15	0.54	0.01
millimeter-wave	7	4.22	0.14	1.02	0.01
audio	6	4.69	0.08	0.45	0.00
hud	6	4.54	0.13	0.21	0.00
gps	5	83.11	82.58	3.48	3.24
video	5	80.65	78.61	5.58	6.12
fuel	5	29.59	31.04	1.05	1.05

The values in the Table II and Figures 4, 5 underscore the differing operational methodologies between ATS and other TSN methods. In the two simulations, multiple streams with identical priorities belong to the same traffic class. This means, these streams were enqueued into the same queues during the

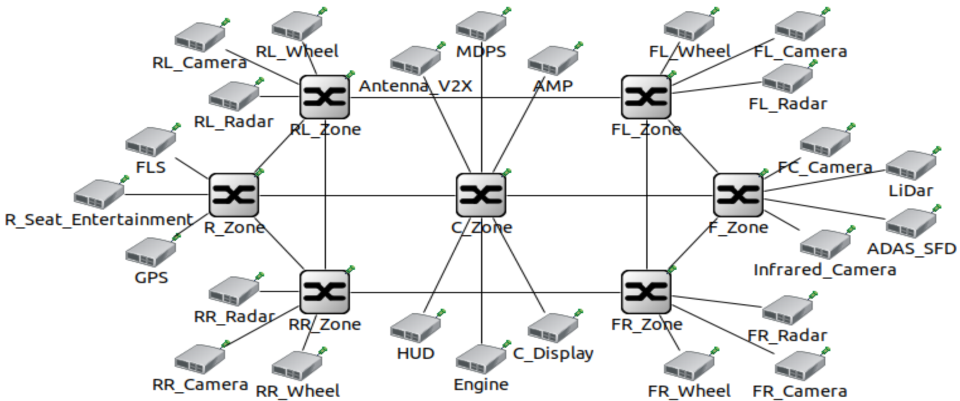

Fig. 3: Automotive Zone-Based Architecture - OMNET++

simulations. In case they were operated by TAS, CBS as an example, their processing remains the same. On the contrary, under ATS operation, despite sharing the same queue, the configuration parameters like CBS and CIR assigned to these streams can differ significantly, and the processing of these streams can be different. This variation in configuration leads to the values for both E2E delay and jitter of the streams that belong to the same traffic class.

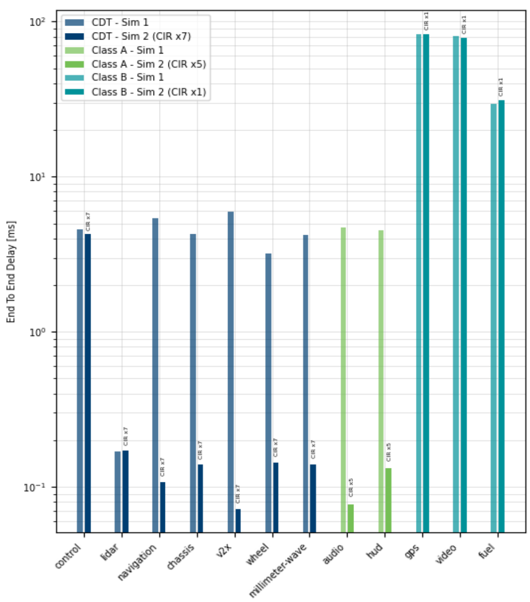

Fig. 4: Average of End-to-End Delay in simulation 1 and 2

The simulations show that CDT and class A streams have similar low delays and jitter, around $\sim 4.11ms$ and $\sim 1.17us$ in simulation 1, and $\sim 0.58ms$ and $\sim 0.09us$ in simulation 2. This holds a significant promise for critical traffic like control, wheel signals, lidar, navigation, and audio transmissions. In contrast, class B streams consistently experience notable delays and jitter, approximately $\sim 64.45ms$ and $\sim 3.37us$ in simulation 1, and $\sim 64.07ms$ and $\sim 3.47us$ in simulation 2, affecting GPS data and video transmission.

In simulation 2, significant improvements in delay and jitter were observed for both CDT and class A streams, attributed

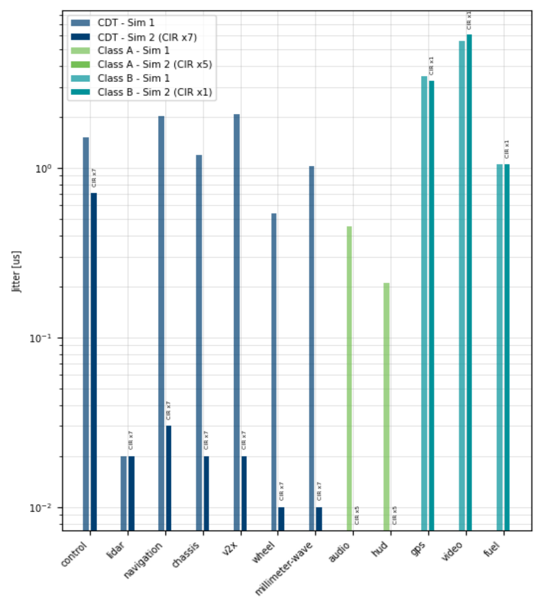

Fig. 5: Average of Jitter (Delay Variation) in simulation 1 and 2

to adjustments in CIR values. However, there was minimal change for class B streams since identical CIR values were maintained in both simulations. This improvement can be attributed to the implementation of ATS shaping with different values of CIR. If CIR values were not big enough that means the token bucket cannot be filled up in time, causing delays in frame transmission by ATS, thus increasing E2E delay and jitter during transmission.

VII. Conclusion

In conclusion, the proposed automotive zone-based architecture, incorporating ATS per-stream filtering, metering, and transmission selection demonstrates efficient data exchange and management within the system. CDT and Class A streams exhibit notably low average E2E delays and jitter, crucial for critical functions like LiDAR, wheel and control signals, and navigation. However, Class B streams consistently face considerable delays and jitter, affecting the transmission of data with high payloads, such as images and videos. Simulation results indicate that adjusting the CIR values in ATS can

TABLE III: Traffic Types Parameters

Traffic Type	**Class**	**PCP**	**Rate** $[Mbps]$	**Payload Size** $[Bytes]$	**Transmission Interval** $[\mu s]$	**Source**	**Destination**
audio	Class A	6	0.704	11	125	R_Seat_Entertainment	AMP
control	CDT	7	0.064	4	500	ADAS_SFD	Engine
navigation	CDT	7	0.2	12	500	GPS	ADAS_SFD
lidar	CDT	7	11	670	500	LiDar	ADAS_SFD
chassis	CDT	7	3.8	230	500	MDPS	ADAS_SFD
v2x	CDT	7	0.26	16	500	Antenna_V2X	ADAS_SFD
gps	Class B	5	33.3	1250	300	ADAS_SFD	C_Display
hud	Class A	6	2.43	30	125	ADAS_SFD	HUD
wheel	CDT	7	2.8	175	500	FL_Wheel	Engine
wheel	CDT	7	2.8	175	500	FR_Wheel	Engine
wheel	CDT	7	2.8	175	500	RL_Wheel	Engine
wheel	CDT	7	2.8	175	500	RR_Wheel	Engine
video	Class B	5	40	1250	250	FL_Camera	C_Display
video	Class B	5	40	1250	250	FR_Camera	C_Display
video	Class B	5	40	1250	250	RL_Camera	C_Display
video	Class B	5	40	1250	250	RR_Camera	C_Display
video	Class B	5	40	1250	250	F_Camera	C_Display
video	Class B	5	40	1250	250	Infrared_Camera	C_Display
millimeter-wave	CDT	7	0.52	42	650	FL_Radar	HUD
millimeter-wave	CDT	7	0.52	42	650	FR_Radar	HUD
millimeter-wave	CDT	7	0.52	42	650	RL_Radar	HUD
millimeter-wave	CDT	7	0.52	42	650	RR_Radar	HUD
fuel	Class B	5	0.3	7	200	FLS	Engine

improve delay performance as well as the jitter for critical streams, underscoring the importance of proper parameter configuration. Nonetheless, maintaining identical CIR values leads to minimal variation across simulations, highlighting the need for tailored configurations to address varying traffic priorities effectively. ATS shaping fulfilled a important role in mitigating delays and jitter, especially with higher CIR, to ensure reliable and timely data transmission within automotive communication systems. Further research and optimization in parameter tuning, aligned with network resources and traffic management, are essential to enhance overall system efficiency and responsiveness.

References

[1] T. Systems, "Comparison of fieldbus systems, can, ttcan, flexray and lin in automotive networks," Online, 2022, accessed: 2024-04-18. [Online]. Available: https://www.talbotsystems.com

[2] S. A. Bagloee, M. Tavana, M. Asadi, and T. Oliver, "Challenges and opportunities of in-vehicle networks in the era of automated driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 82–99, 2019.

[3] "Amendments to IEEE std 802.3™-2022 raise the speed bar for automotive ethernet," *IEEE Standards Association*, 2022.

[4] S. Samii and H. Zinner, "Level 5 by layer 2: Time-sensitive networking for autonomous vehicles," *IEEE Communications Standards Magazine*, vol. 2, no. 2, pp. 62–68, 2018.

[5] Y. Peng, B. Shi, T. Jiang, X. Tu, D. Xu, and K. Hua, "A survey on in-vehicle time-sensitive networking," *IEEE Internet of Things Journal*, vol. 10, no. 16, pp. 14 375–14 396, 2023.

[6] "IEEE standard for local and metropolitan area networks–timing and synchronization for time-sensitive applications," *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)*, pp. 1–421, 2020.

[7] "IEEE standard for local and metropolitan area networks – bridges and bridged networks - amendment 25: Enhancements for scheduled traffic," *IEEE Std 802.1Qbv-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, and IEEE Std 802.1Q-2014/Cor 1-2015)*, pp. 1–57, 2016.

[8] "IEEE standard for local and metropolitan area networks–bridges and bridged networks–amendment 29: Cyclic queuing and forwarding," *IEEE 802.1Qch-2017 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd(TM)-2015, IEEE Std 802.1Q-2014/Cor 1-2015, IEEE Std 802.1Qbv-2015, IEEE Std 802.1Qbu-2016, IEEE Std 802.1Qbz-2016, and IEEE Std 802.1Qci-2017)*, pp. 1–30, 2017.

[9] P. Chulsun and P. Sungkwon, "Performance evaluation of zone-based in-vehicle network architecture for autonomous vehicles," *Sensors*, vol. 23, no. 3, p. 669, 2023.

[10] OMNeT++ Discrete Event Simulator. Available online. https://omnetpp.org/ (Accessed: 2024-02-05).

[11] F. Luo, B. Wang, Z. Yang, P. Zhang, Y. Ma, Z. Fang, M. Wu, and Z. Sun, "Design methodology of automotive time-sensitive network system based on omnet++ simulation system," *Sensors*, vol. 22, no. 12, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/12/4580

[12] "IEEE standard for local and metropolitan area networks–frame replication and elimination for reliability," *IEEE Std 802.1CB-2017*, pp. 1–102, 2017.

[13] "IEEE standard for local and metropolitan area networks–bridges and bridged networks - amendment 34: Asynchronous traffic shaping," *IEEE Std 802.1Qcr-2020 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018, IEEE Std 802.1Qcc-2018, IEEE Std 802.1Qcy-2019, and IEEE Std 802.1Qcx-2020)*, pp. 1–151, 2020.

[14] B. Fang, Q. Li, Z. Gong, and H. Xiong, "Simulative assessments of credit-based shaping and asynchronous traffic shaping in time-sensitive networking," in *2020 12th International Conference on Advanced Infocomm Technology (ICAIT)*. IEEE, 2020, pp. 111–118.

[15] A. Nasrallah, A. S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. Elbakoury, "Performance comparison of IEEE 802.1 tsn time aware shaper (tas) and asynchronous traffic shaper (ats)," *IEEE Access*, vol. 7, pp. 44 165–44 181, 2019.

[16] Y. Lee, J. Kim, and J. Jeon, "Flexray and ethernet avb synchronization for high qos automotive gateway," *IEEE Transactions on Vehicular Technology*, vol. PP, pp. 1–1, 12 2016.

[17] L. Leonardi, L. Lo Bello, and G. Patti, "Performance assessment of the ieee 802.1qch in an automotive scenario," 11 2020, pp. 1–6.

[18] H. Askaripoor, M. Hashemi Farzaneh, and A. Knoll, "E/e architecture synthesis: Challenges and technologies," *Electronics*, vol. 11, no. 4, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/4/518

[19] Toshiba Electronic Devices and Storage Corporation. Available online. https://toshiba.semicon-storage.com/eu/semiconductor/design-development/innovationcentre /articles tcm0645˙AutomotiveTC9563XBG.html (Accessed: 2024-04-01).

[20] INET Framework. Available online. https://inet.omnetpp.org/ (Accessed: 2024-02-15).