POLITECNICO DI TORINO

Master's Degree in Mechatronic Engineering

Master's Degree Thesis

Automotive Ethernet: Construction and Performance Analysis of a 10BASE-T1S Network

Supervisors
Prof. Claudio Ettore Casetti
Prof. Ivan Cibrario Bertolotti

Candidate
Martina Ciraolo

Internhip Tutors
Teoresi S.p.A.
Ing. Bernardo Sessa
Ing. Giuseppe Giuliano

Academic Year 2023-2024

Abstract

During the latter half of the 20th century, advancements in electronics had a profound impact on the automotive industry. Specifically, they paved the way for numerous opportunities to enhance performance, reliability, passenger safety, and overall comfort. To meet these evolving demands, in-vehicle networks were introduced. While the controller area network (CAN) emerged as the most prevalent technology, other alternatives such as LIN, FlexRay, MOST, and Ethernet also gained prominence. Notably, Ethernet networks have garnered increased attention for future development due to their ability to fulfill the requirements for high data rates. To cope with the growing complexity of modern systems like ADAS (Advanced Driver Assistance Systems), automobile manufacturers are adopting innovative approaches. They're streamlining wiring harnesses, reducing vehicle weight, and simultaneously enhancing software complexity. The integration of Ethernet Networks foresees a shift from domain-based to zonal architecture. This entails clustering domain functions based on their physical location within the vehicle. This transition empowers sensors and actuators to operate autonomously, independent of the central vehicle compute node. The thesis will illustrate the structure of the Ethernet frame and its interactions with the MAC sublayer. Subsequently, it will provide a comprehensive overview of the upper OSI (Open System Interconnection) layers, offering valuable insights into their composition and functions. It will provide a comprehensive overview of the IEEE 802.3 standard and trace the development of Ethernet technologies from 10BASE5 to 10BASE-T1S, with a specific emphasis on the latter in the experimental section. In 10BASE-T1S communication among nodes can be managed through two access control methods: CSMA/CD (Carrier Sense Multiple Access with Collision Detection) and PLCA (PHY-Level Collision Avoidance); their operational principles will be elaborated upon to provide a deeper understanding of their functioning. The primary objective of this study was to thoroughly examine and assess the capabilities of an automotive Ethernet network utilizing the 10BASE-T1S protocol. The environment within which the experimental tests are conducted consists of four integral elements, including a computer equipped with Vehiclespy software, a RAD-Comet, and two RAD-Meteors, all developed by Intrepid Control Systems and provided by Teoresi S.p.A., the company where the thesis was conducted. Operating from the central computer, each simulation was meticulously orchestrated to facilitate data transmission among the RAD-Meteor devices and the PC itself, offering a unique opportunity to observe the interplay of their activities within the shared network. Data traffic flowing through the network was captured by Vehiclespy and recorded in .pcap files for further analysis. Moreover, a custom Matlab script

was devised to extract and analyze message data from each file, shedding light on how message traffic influences network performance metrics such as latency and jitter. Additionally, comparative evaluations were conducted to discern the impact of different network access control methods, namely PLCA and CSMA/CD, on overall network performance.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This chapter will present an overview of how the in-vehicle network evolved in the past years, it will discuss the process that led electronics to progressively reach a key role in the automotive industry. Furthermore, there will be a focus on the opportunities and advantages offered by using Ethernet networks for in-vehicle applications.

1.1 Thesis Outline

This thesis may be subdivided into two primary segments: the initial section will provide an insightful examination of electronic communication systems within the automotive industry, with a dedicated focus on theoretical aspects and state of the art. Subsequently, the latter portion will expound upon the development of an Ethernet 10BASE-T1S network.
Chapter 1 gives an insight into the history of in-vehicle networks and the role of Ethernet
Chapter 2 Illustrates the embedded networks utilized by the automotive industry
Chapter 3 describes the Ethernet networks, with a focus on technical aspects.
Chapter 4 introduces the most well-known Ethernet Technologies while paying particular attention to the 10BASE-T1S, which will have a key role in the experimental part of this thesis.
Chapter 5 gives an insight into the CSMA/CD and PLCA protocols that are deployed within Ethernet networks in order to correctly transmit data whilst avoiding collisions.
Chapter 6 Gives a description of the testing environment and setup.
Chapter 7 Encompasses the results of the simulations.
Chapter 8 Contains the conclusions and considerations for future works.

1.2 Historical Overview

The development of electronics that occurred in the second part of the XX century had an astonishing impact on the automotive industry. The 1970s remark the years in which there was an exponential increase in the number of electronic systems that progressively substituted purely mechanical and hydraulic ones [1]. This growing trend opened many more opportunities in terms of better performance, reliability, safety for the passengers and comfort; for example, without electronics, we wouldn't have essential devices such as ABS (antilock braking systems), ESP (electronic stability program), active suspensions, etc... The first ECUs were introduced in order to implement each new single function and all the subsystems were connected by a dedicated cable, hence, as time passed, this configuration turned out insufficient for dealing with the huge amount of data to be exchanged since the connections were point-to-point. As a result, a solution was constituted by the development of in-vehicle networks and the definition of their protocols; furthermore, to minimize the number of cables, fieldbuses have been implemented to allow communication between nodes connected with them [2].

The first appearance of such networks can be dated to the mid-1980s, in those years Robert Bosch proposed the Controller Area Network (CAN), which was integrated into the vehicles produced by Mercedes in the early 1990s; to this very day CAN is still one of the most widespread networks in the automotive industry. Afterwards, each car manufacturing company proposed its own network technology, and as a consequence, it impelled the need for standardization and CAN was designated as the most suitable solution. Traditional CAN, however, is characterized by low-latency communication but cannot provide for high bandwidth, this is a remarkable drawback since modern cars integrate many sensors that generate a massive quantity of megabytes to be transmitted. As time passed by, new improvements for these networks occurred: higher-level CAN protocols were proposed such as CAN Kingdom, CANopen, and DeviceNet, or, in terms of higher speed of data transmission, CAN-FD and CAN XL. Furthermore, other fieldbus technologies were developed as a response to the several requirements of the car components, in terms of bandwidth, latency, and flexibility: LIN, Ethernet, FlexRay, and MOST are the most well-known networks deployed in today's car manufacturing alongside with CAN. All these technologies will later be discussed in detail.
Throughout the last decades, on-board electronics have had a significant impact on both the complexity of car design and costs [3], but nonetheless, the benefits are outstanding: just consider that BMW succeeded in lightening by 15kg the weight of one of their cars just by replacing the wiring harness with a LAN network in correspondence with the four doors.[1]

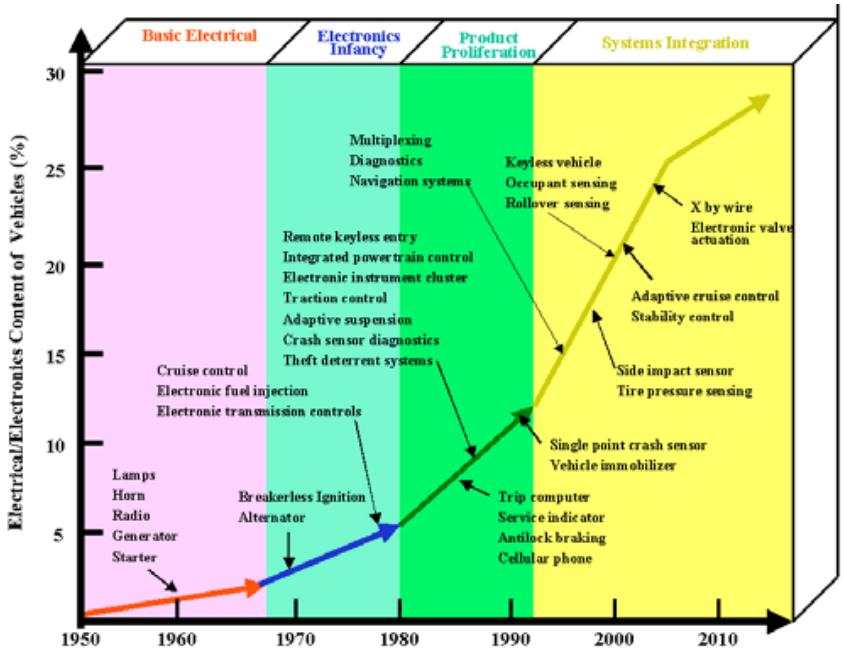

Figure 1.1: History of Automotive Electronics, source:[4]

The picture above shows the evolution of automotive electronics throughout the years. It is possible to notice how as the requirements on safety, reliability, and passenger comfort were more stringent the number of ECUs inside the vehicle increased exponentially. It is noticeable that the last decades have been characterized by this tendency to gradually replace as much as possible mechanical harnesses (such as the ones controlling steering and braking) with electronic systems, in order to ensure more reliability and overcome their technological limitations. This recent trend moved the focus to the x-by-wire systems.
Embedded software has emerged as a pivotal factor for moving towards advanced capabilities and functionalities within automotive systems, from contemporary cars on the road to specialized off-road vehicles like construction machinery. A remarkable outgrowth of the request for innovative software-driven features is the stunning surge in complexity of the automotive software. This, in turn, constitutes an interesting challenge in the process of development; just consider that in modern cars it is possible to encounter almost 100 million lines of source code. [5]

The new technologies that have been introduced lately are the following: [6]

- **"Connected cars"**: internet access through Wi-Fi, remote diagnostics, real-time information, firmware updates.

- **Augmented Reality**: to obtain information about the hazards along the road.
- **V2V**: vehicle-to-vehicle communication, this technology has been estimated to reduce crashes by 79%.
- **Self-Driving Vehicles** that make use of many sensors, lasers, and cameras. Many experiments have been led by Google even on public roads.

In recent years the automotive industry has witnessed many innovations, might mention, above all, the ADAS (Advanced Driving Assistance Systems), as well as infotainment and even the aforementioned Automated Driving. All these technologies require large bandwidths, scalability, and low latency, that in some cases cannot be provided by classical networks. Hence, a possible solution is constituted by Ethernet networks, which are the main focus of this thesis.

1.3 Ethernet in the Automotive industry

Although Ethernet constitutes a significant improvement in the automotive industry, it hasn't previously been integrated within the car body since it lacked some important requirements:

- Difficulty in controlling bandwidth allocation
- Distress in synchronization between devices
- Did not meet the OEM EMI/RFI requirements
- Sensitivity to noise (Especially for 100Mbps Ethernet).
- Absence of an economical wiring technology

The development of Automotive Ethernet can be subdivided into three generations as follows: [7]

1. **Generation 1: Diagnostics over IP** through Ethernet 100BASE-TX and CAT5.
2. **Generation 2: ADAS and Infotainment** camera systems, sensors etc...
3. **Generation 3: Ethernet as network backbone**, characterized by a hierarchical structure with the ECUs connected through the backbone. [7]

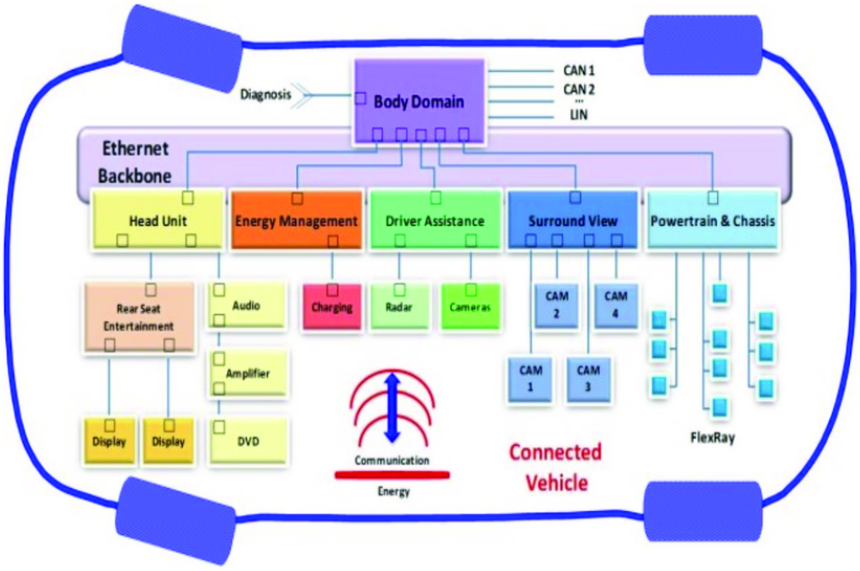

Figure 1.2: Ethernet Backbone in Domain Architecture, source:[7]

1.4 Why Integrating Ethernet Networks?

Ethernet networks have emerged as a versatile platform for the automotive sector, achieving increasing attention in recent years. Recognizing its considerable potential, automotive manufacturers have actively supported novel research attempts aimed at refining and establishing standardized Ethernet solutions.
Ethernet's success can be attributed to many factors:

1. **Higher bandwidth**
2. **Higher range of data-rates** compared with current in-car networks.
3. **Lower weight**, the cabling harness is the 3rd highest-cost and heaviest component in the car, automotive Ethernet reduces the wiring weight by 30%. This would result in a discernible abatement in fuel expenditure.
4. **Lower impact on the supply chain** labor costs are reduced by 80%.
5. **Cost reduction** for the electronic and wiring harness.
6. **Scalability** large numbers of sensors and actuators can be added to the vehicle without being directly connected to the zonal controllers.
7. **Flexibility**

8. **Reliability**
9. **Support** to the Internet Protocol (IP).
10. **IEEE Standardization**

Figure 1.3: Wiring Harness, source:[6]

Ethernet provides a large spectrum of bit rates from 10 Mbps to 10 Gbps and its standardized physical layers are accessible and reliable in demanding environmental conditions. Additionally, the family of IEEE Time-Sensitive Networking (TSN) standards encompasses a range of services specifically for Automotive communication such as fault tolerance, precise time synchronization, predictable latency, facilitation of scheduled traffic, and high reliability. It is estimated that in the upcoming generation of hybrid and electric cars, Ethernet will play an essential role in vehicle communication, allowing a massive amount of data to be exchanged and introducing advanced features.
The foremost objective of Automotive Ethernet was to meet the exigencies for elevated data rates. Although, even now, developers find themselves confronted with fresh challenges as they are required to fulfill the prospective demands of the market regarding new technologies: [8]

- Mobile services and data
- E-Mobility
- Autonomous Driving

In particular Autonomous driving requires higher resolution for sensor data and displays, and fast communication systems, hence why Ethernet happens to be the most suitable solution with respect to other networks. The table below aims to compare the performance and characteristics of Ethernet with respect to CAN-FD and FlexRay.

	Automotive Ethernet	CAN, FlexRay
Number of ECUs in network	256-> unlimited, bandwidth can be added	CAN e.g. 16 (SAE), limited by bandwidth
Bandwidth efficiency	50-90%	20-55%
Topologies	Extremely Flexible	Limited
Priority and timing schemes	AVB-TSN (various)	Message or ID/ timeslot based (one)
Addressing	MAC or IP	None
Security	State of the art	Limited
Domain separation	Physical or virtual	Physical
Extendibility	Switch port, no changes to existing nodes	CAN: no changes to existing nodes, FR: may need active star port + timing
Service Orientation	Possible	Not really possible

Table 1.1: Comparison between Ethernet and CAN/FlexRay source:[8]

Chapter 2

Anatomy of Automotive Electronics

In this chapter, an insight will be provided regarding the primary domains delineating the service areas within a car, along with their respective requisites. Subsequently, an exposition will arise on the present and prospective connectivity methodologies, placing emphasis on the most widespread protocols such as CAN, LIN, FlexRay, and MOST, elucidating their functionalities and objectives. Finally, an exploration of the concept of Time-Sensitive Networks will be introduced.

2.1 Vehicle Domains

It is quite common practice to consider the car as subdivided into different domains. Each one of them is characterized by a specific and independent electronic harness, since, based on their purpose, every single domain has specific and unique requirements in terms of bandwidth and latency. Nowadays, even though all those areas are more interconnected than before, they keep maintaining standalone computing entities [6]. The aforementioned domains are specifically the following:

- **Powertrain**
  Comprises all those constitutive elements that contribute to the generation of power such as engines, rods, transmission organs, etc... Nevertheless encompasses a plethora of sensors that gather information about flow, pressure, shaft torque, fuel temperature, and engine speed to name a few.
  Powertrain domain requires a rather powerful computer unit in order to ensure efficiency and reliance, this hallmark, combined with the multitude of sensors, establishes stringent requirements in terms of high bandwidth and very low latencies.

- **Chassis**
  Holds for the structural support of the vehicle's body and components. This framework comprises for example: suspensions, steering, brakes... Along with the powertrain, even the chassis domain entails many sensors and accounts for essential services; hence, as a consequence, it demands precise timing communication with regulated upper limits on latency.

- **Body and Comfort**
  Involving facilities to assist the passengers and optimize well-being throughout the journey, A/C, seating and windows adjustments and lights.
  Since those are not essential services, communication systems with low bandwidth and high latency are sufficient.

- **Infotainment and Driver Assistance**
  These are a set of facilities that have been introduced in recent years, and their main purpose is to assist the pilot while driving/parking and enhance their safety; namely GPS, cruise control, collision avoidance systems, lane change assist, blind spot recognition. This category of services is among the most rapidly evolving sectors reaching the attention spot in the contemporary automotive system development.
  Infotainment accounts for entertainment benefits such as music and video coupled with information about the weather, traffic conditions, and access to the internet; it might require high bandwidth. [9]

Along with this one might mention Air-bag which is a passive safety system, X-by-wire subsystems, Wireless and Telematics.
The picture below (Fig.2.1) summarizes what was said before, providing a visual representation of the various domains inside a vehicle.

2.2 Requirements on Communication Networks

Every component within the automotive domains is characterized by distinct prerequisites, determined by their operational purpose and their contribution on the safety of drivers and passengers. Notably, the development of fieldbus technologies is meticulously tailored to precisely accommodate the following criteria [2]:

- Bandwidth: it is important to take into account the quantity of data to transmit and the cost of the devices since, occasionally, even lower-bandwidth buses are sufficient.

- Latency/Determinism: accounting for correctness in transmission timing and message reception.

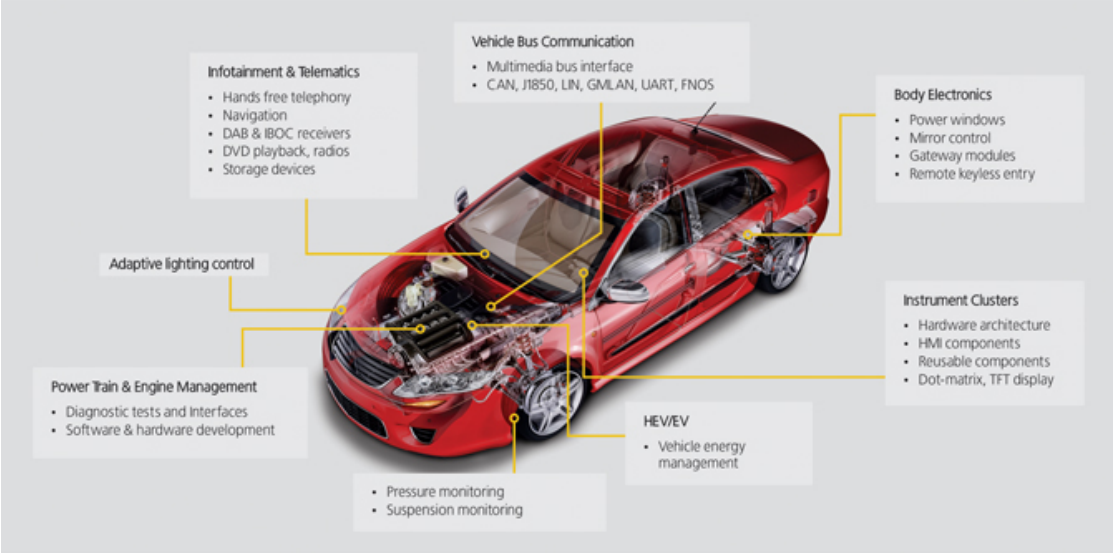

Figure 2.1: Automotive domains, source: [10]

- Flexibility: dealing with overloads of dataflow, time-triggered messages, etc...
- Fault Tolerance: refers to the system's ability to manage and continue operating without interruption or degradation in performance despite the occurrence of malfunctions. For example, instances of circuit defects.
- Security: protecting the data transmitted by possible attacks.

2.3 Domain vs Zonal Architecture

The integration of electronic systems and components within vehicles has intensely influenced their design and manufacturing processes. The progression towards interconnected automobiles and autonomous vehicles has led to a divergence in the methodologies adopted by automakers in configuring the communication architecture governing a vehicle's electronics. Simultaneously, the incorporation of sensors into the vehicle's architectural framework, collecting a massive quantity of data to be analyzed, has exponentially intensified the request for augmented computing power. This condition paved the way for an evolution from the domain to the zonal architecture. [11]

2.3.1 Domain Architecture

The Domain Architecture is based on the aggregation of functionally related Electronic Control Units (ECUs) under a shared domain controller or gateway.

This approach delineated distinct subsets of ECUs within powertrain, chassis, infotainment, and passenger comfort domains. Nevertheless, this particular domain architecture does not prioritize the optimization of wiring, as the ECUs associated with each functional subsystem were dispersed throughout the vehicle rather than being strategically centralized.[11]

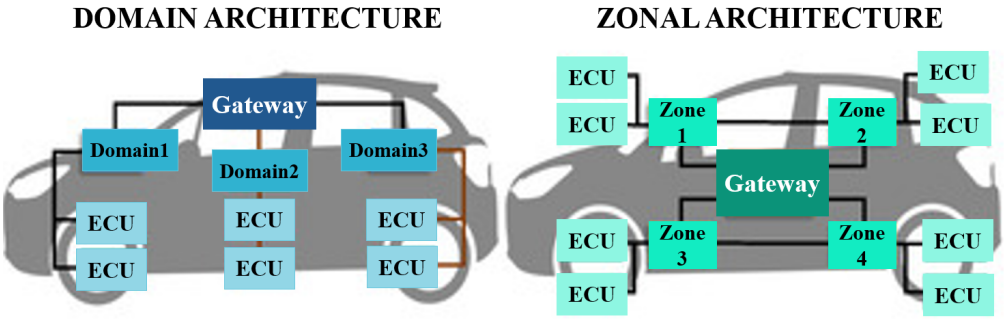

Figure 2.2: Comparison between domain and zonal architectures, source:[12]

2.3.2 Zonal Architecture

In order to hold for the increasing complexity of new systems such as the ADAS, car manufacturers are moving towards new solutions, reducing the number of wiring harnesses and lightening the weight of the car, while at the same time increasing the complexity of the software. One of the most important features of zonal architecture is the opportunity to accommodate a greater or lesser number of sensors and electronic components across the entirety of the vehicle, providing enhanced flexibility for different variations of a vehicle model. Essentially, zonal architecture consists of grouping numerous, if not all, domain functions according to their geographical location or zone within the vehicle. [11]
The transition from a domain to a zone architecture will enable the autonomy of sensors and actuators from the central vehicle compute node. In practical terms, this means that hardware and software update cycles can operate independently, and the designs of sensors and actuators can endure across multiple vehicle design cycles. The zone architecture empowers Original Equipment Manufacturers (OEMs) with heightened control, encompassing high-level software maintenance through over-the-air updates, firmware-over-the-air (FOTA) updates, and an always-on-cloud connection. This comprehensive control facilitates implementing new functions and enhancing features, particularly in realms like autonomous driving. Zone modules facilitate more optimized power distribution topologies, including the ability to

power down unused modules. This feature proves especially advantageous in battery electric vehicles and hybrid electric vehicles. The shift in power distribution involves transitioning from a centralized to a decentralized implementation, employing smart fuses within the zonal modules. As a result, sensors and actuators will exhibit a higher level of intelligence. Certain functionalities, such as control loops, will migrate to zonal modules, allowing for an increased emphasis on service-based communication rather than signal-based communication.

The zone Electrical/Electronic (E/E) architecture has a substantial impact on sensing and actuation functions at the periphery of a vehicle, commonly referred to as the edge. In domain architectures, specialized Electronic Control Units (ECUs) situated in close proximity to sensors or actuators handle these functions. The incorporation of new features and functions typically results in the introduction of new ECUs, each requiring dedicated battery power and networking wires. This contributes to a further escalation of harness complexity. The integration of zonal modules presents an effective solution to mitigate this complexity. It achieves this by consolidating the logical input/output (I/O) functions of multiple ECUs into the zonal module and preserving the locations of sensors and actuators. This, in turn, significantly reduces harness complexity. [13] The shift to a zone topol-

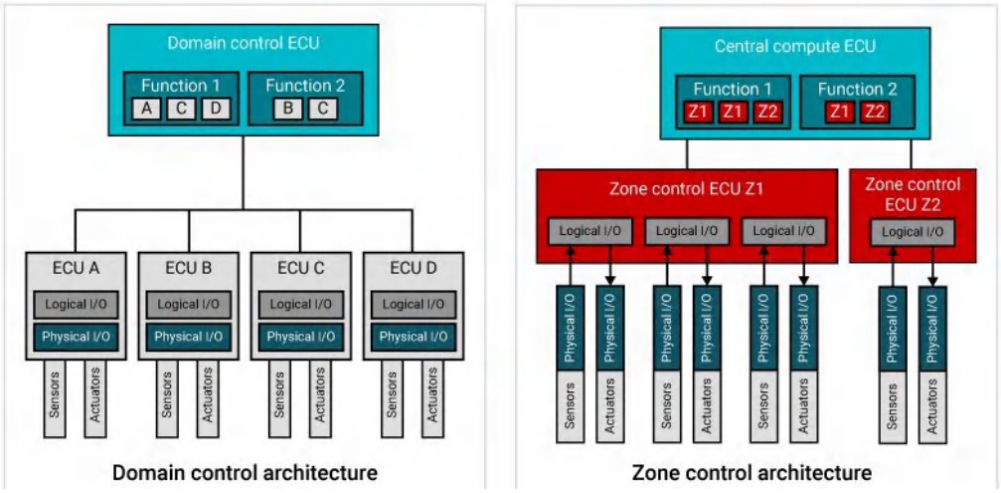

Figure 2.3: Logical and physical I/O functionalities from in domain and zone architecture, source [13]

ogy underscores the need for an increased focus on networking, with a growing preference for high-bandwidth interfaces like Peripheral Component Interconnect Express (PCIe) and Gigabit Ethernet. Choosing the right physical layer (PHY) is pivotal to meeting bandwidth requirements. A typical zonal module block diagram incorporates high-speed communication links to cater to diverse throughput

needs and facilitate traffic between the zonal module and central computing. The significance of Gigabit Ethernet and, potentially, PCIe cannot be overstated in addressing these demands. In some scenarios, leveraging PCIe retimer or redriver devices for lengthy cable connections proves advantageous. Connections to and from sensors and actuators may require more economical, lower-bandwidth bus systems like LIN. Despite varying bandwidth requirements, a standardized bus with standardized links is indispensable for all connections. To ensure future-proofing, the network topology allocates spare bandwidth, particularly between the zonal module and central computing, facilitating software upgrades while preserving existing validated hardware. [13]

In the near future zonal architecture will be characterized by relevant features; for example sensors and actuators will be linked to zonal gateway ECUs, enhancing the efficiency and coordination of data flow within the vehicle's architecture and the communication infrastructure will incorporate Automotive Ethernet TSN backbone, offering high bandwidth and deterministic real-time communication capabilities. The Zonal Gateway Electronic Control Unit (ECU) provides and distributes both data and power within a designated vehicle zone, supporting a wide range of interfaces for sensors, actuators, and displays. This encompasses various network differences or signal types, providing adaptability across different components. The introduction of 10BASE-T1S presents an opportunity to replace other interfaces such as CAN FD and FlexRay. This can streamline communication within the vehicle's systems and contribute to enhanced efficiency. Acting as a gateway, switch, and smart junction box, the Zonal Gateway ECU consolidates multiple functionalities into a single component. This integration aims to optimize the management of data, power, and communication within the designated vehicle zone. [14]

2.4 Networking technologies: overview on CAN, LIN, FlexRay and MOST

As mentioned before, in response to the different requirements demanded by the domains new wiring technologies arose, the ones mentioned in this section have been largely adopted and standardized.

2.4.1 CAN

The original version of Controller Area Network was developed around 1983, it's characterized by quite low bandwidth (1Mbps). CAN domains of application are mainly in powertrain, chassis, and body electronics. It is a multi-master serial

communication protocol that connects multiple ECUs. Characterized by a solid two-wire structure that has low weight. CAN maximum speed ranges from 125 Kbps (even less in some cases) up to 1Mbps, with greater bus lengths corresponding to lower speeds. Its maximum payload is 8 bytes. An improvement for this protocol is constituted by CAN-FD and CAN-XL [9].

- **CAN-FD** provides higher transmission speeds up to 5Mbps and, at the same time, longer payloads (up to 64 bytes).

- **CAN-XL** can provide transmission speeds up to 20Mbps and, at the same time, longer payloads (up to 2048 bytes). [3]

2.4.2 LIN

The ideation of the Local Interconnected Network can be traced back to 1998, owing its conceptualization to the collaborative efforts of Audi, Volvo, Volkswagen, BMW, Chrysler, and Motorola. Its first integration into automotive production can be dated back to 2001, following its initial standardization in the year 2000. LIN requires only one shared wire and is characterized by a master-slave architecture that can accommodate up to fifteen slave nodes, however, its maximum length is 40 m. In this configuration, the master node requests information from the single slave nodes through a header, and each slave responds by sending the required data (see Fig.2.4). Its primary strength lies in its affordability. The performance in terms of data rate, however, is quite modest (20Kbps), as a consequence it is implemented mostly for the components comprised in the "Body and Comfort" domain and connected to the other networks through a LIN/CAN gateway. [2]

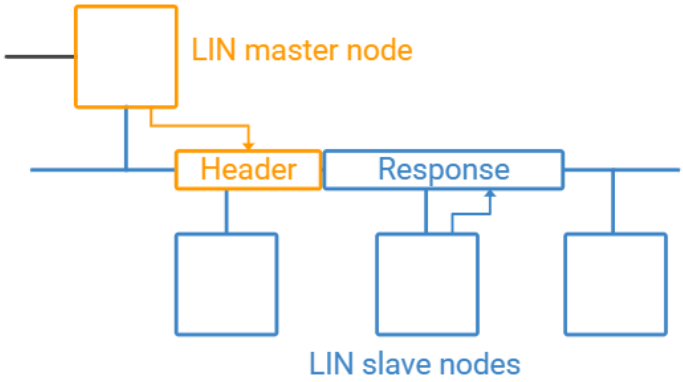

Figure 2.4: LIN bus data flow, source: [15]

2.4.3 FlexRay

It has been developed in order to face up with the new requirements in the optics of the evolution of car electronics towards x-by-wire, its availability can be dated back to 2005. FlexRay consists of two twisted unshielded pairs of cables configured in a structure that can be either be bus, star or hybrid. FlexRay provides data-rate speeds up to 10/20 Mbps and holds for 64-byte long payloads. FlexRay was developed to assure safety-critical applications, but on the other hand, it has a higher cost. Its protocol was introduced in the year 2004 [2]. It can be implemented in high-performance powertrain and safety e.g. adaptive cruise control.[6] It is believed that, due to its lower flexibility, in the future, it will be substituted by Automotive Ethernet.

2.4.4 MOST

Media Oriented Systems Transports. Its development can be dated to 1997, supported by Audi, BMW and Daimler-Chrysler. It is mostly intended for multimedia devices such as GPS, music, and speakers since it can interconnect up to 64 devices. It consists of a synchronous network driven by a single Timing Master that transmits synchronous frames. MOST is widely used by more than 60 companies [2] It can hold the transmission of 25 up to 150 Mbps through an optical fiber in a shared ring topology.

2.4.5 Other Technologies

ByteFlight. Developed by BMW in 1996, characterized by maximum data rates of 10 MBps. Its main field of application is in safety systems such as airbags or seat belt tensioners. Future development will request higher bandwidth hence it will not be used alongside CAN anymore, it is believed to be integrated within FlexRay to serve x-by-wire systems. [2]

LVDS. Low Voltage Differential Signaling (LVDS) operates as a point-to-point bus offering a less expensive alternative to MOST. However, one of its limitations is that it can only interface with a single camera or video input.

As already mentioned the future of car electronics is focused on replacing hydraulic and mechanic systems by means of the x-by-systems that require fault-tolerant communication and make use of deterministic messages, all these requirements can be handled through TDMA protocols, in particular, the most widely used are TTP and TT-CAN. [2]

- **TTP Time-Triggered Protocol** can be dated to 1994. It's available in two

versions: TTP/A and TTP/C. The first is a master/slave protocol that can reach up to 25 MBps of datastream speed, while the latter is fully distributed but is characterized by its high cost and limits on bandwidth. TTP is highly fault-tolerant even though, unlike FlexRay, it lacks support for event traffic.

- **TT-CAN Time Triggered CAN**, was first introduced in 1999, accounts for both time- and event-triggered messages. [2]

Functional Domain	Description	Networking Technologies
Powertrain	Control of engine and transmission	CAN, CAN/FD, FlexRay
Chassis	Control of the vehicle stability and dynamics according to steering/ braking solicitations and driving conditions (e.g., ground surface, wind, etc...)	CAN, CAN FD, FlexRay
Body and Comfort	Control of doors, windows, roof, and seats, climate control, etc.	LIN, CAN, CAN/FD
Multimedia/ Infotainment	Audio CD, DVD players, MP3 players, TV, Rear Seat Entertainment, navigation information services, etc.	MOST, CAN
Human Machine Interface	Advanced Display technologies	MOST, CAN
ADAS	Lane Departure Warning, Traffic Sign Recognition, Night vision, Pedestrian detection, Parking assistant, etc.	CAN, FlexRay

Table 2.1: Automotive Function Domains and their networking technologies source:[3]

Chapter 3

Ethernet

Automotive Ethernet represents a specialized evolution of Ethernet networks designed to tailor the physical layer to the unique requirements of automotive applications. Ethernet networks offer network designers a spectrum of configuration options. These encompass different protocols, methodologies, and components, enabling the prioritized delivery of essential data, signals, and services, all within the framework of a singular physical network that accommodates various types of data. Ethernet is the most widespread LAN (local area network), its first version was developed by Xerox PARC between 1973 and 1974 as a method for facilitating communication among Alto computers. It uses an unshielded twisted pair cabling instead of the coaxial cabling used in older technologies such as 10BASE2 and 10BASE5 networks, consequently, it proves to be a cheaper and lighter solution. [16]

3.1 Historical Notes

In the 70s a consortium of three companies (Digital Equipment Corp., Intel Corp. and Xerox Corp.) called DIX set the stage for the first version of Ethernet (1.0) operating at 10Mb/s. At the same time, the IEEE committee was working on the 802.3 standard based on Ethernet, which was approved as an ISO standard in 1989, and, across the years was updated and deepened, evolving from LAN to Switched LAN reaching data rates up to 400 Gb/s. In the mid-90s the 100BASE-TX standard was introduced, while the conclusion of the initial decade of the 2000s witnessed the establishment of Ethernet standardization at 10 Gb/s. [17]. The first implementations of ethernet networks were realized employing multimodal optical fibers 100/140, while in the last years, they have been substituted by 62.5/125 and 50/125 ones.

3.2 Physical Layer

The picture below shows a type of physical layer currently used is the BroadR-Reach technology promoted by the OPEN alliance for Ethernet 100BASE-TX technology. The purpose of the BroadR-Reach is to convert a stream of 100 Mb/s data into a ternary signal that can assume three possible values: +1V, 0, and -1V; furthermore, unlike traditional automotive networks, it supports fully duplex communication allowing the nodes connected to transmit and receive signal at the same time. This particular opportunity is made possible thanks to echo cancellation and it can be tested by means of the PHY chip from DSP (digital signal processing), channel quality and bit errors. [18]

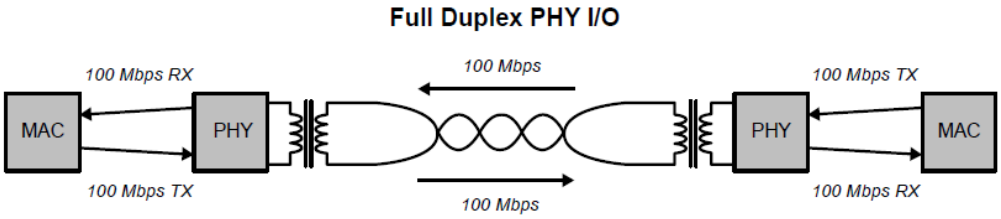

Figure 3.1: Full-Duplex physical layer, source: [18]

At the Physical Layer level, the interconnections are point-to-point since each UTP cable can connect up to a maximum of two nodes and the integration of a switch is necessary. Since, generally speaking, networks contain more than 2 devices, they will contain multiple port switches connected to the other nodes with a dedicated UTP cable; the result is what is called a network segment, and the technique is called microsegmentation.
Since Automotive Ethernet is characterized by these elements, one of the most common network configurations whenever there is only one switch is the star topology. The main aspect of this topology is the fact that its broadening is straightforward, as it involves simply adding more connections up to the number of ports on the switch. Each star can be further connected to other ones creating a tree topology. Thanks to this topology, in the presence of unicast traffic the data transmitted from one node to the switch is then redirected only to the port corresponding to a predefined receiver.

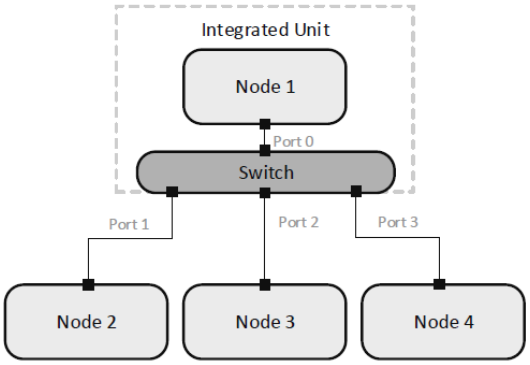

Figure 3.2: Ethernet Star Topology, source: [18]

3.2.1 Format of the Ethernet Frame

The Ethernet frame in its elementary structure is defined by the IEEE 802.3 standard. The frame begins with the preamble and the SFD (Start Frame Delimiter), the Ethernet Header is 14 bytes long and is composed of the Destination address, Source address, and Length, successively it terminates with the Data to be transmitted that can accommodate up to 1500 byte and ends with the frame check sequence whose objective is to find errors. [19]

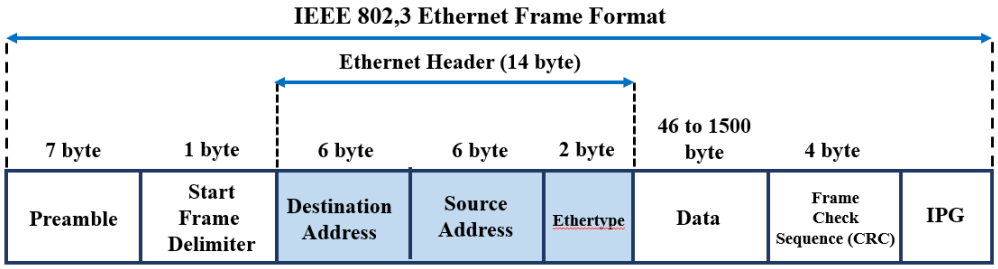

Figure 3.3: Ethernet Frame Format, source: [19]

- **Preamble**
  It's a sequence of 7 bytes that alternates 0s and 1s and allows synchronization from the sender through the receiver.

- **Start of frame Delimiter (SFD)**
  A 1-byte fragment that is always set to 10101011, informs the station or stations that synchronization is currently unattainable.

- **Destination Address**
  Is a 6-byte section that holds the MAC address of the device to which the data is directed.
- **Source Address**
  Is a 6-byte section that holds the MAC address of the device that is sending the frame.
- **Ethertype**
  Is a 2-byte element that indicates the total size of the Ethernet frame.
- **Data**
  Also known as payload contains the data to be transmitted.
- **Frame Check Sequence FCS**
  Cyclic Redundancy Check (CRC). A 4-byte long sequence contains a code that allows one to detect, whether the data has been damaged throughout the transmission.

For the sake of completeness, in contemporary networking, Ethernet frames may now encompass an additional 4-byte header known as the "IEEE 802.1Q header," positioned between the Source MAC address and the Length field. The first two bytes allow the header identification, while the last two bytes constitute the Tag Control Information. The TCI is subdivided as follows: the first three bits carry the priority information, one bit carries the Drop Eligible Identifier (DEI), and the last 12 bits are the Virtual LAN Identifier, specifying the VLAN to which the packet belongs. [20]

While describing the ethernet frame it is possible to notice the absence of a beacon that indicates the end of the packet, to overcome this problem, this role is assumed Inter-Packet Gap whose duration should not be less than the minimum value specified as 96-bit time. A network is said to be transmitting “wire speed” when the IPG between two minimum-length packets has also the minimum possible length.[19]

3.3 Data Link layer

The Data Link Layer as defined in IEEE 802.3 is composed of two sublayers: MAC (Media Access Control) and LLC (Logical Link Control).

3.3.1 MAC sublayer

MAC sublayer offers an abstraction layer for the Logical Link Control, Its duty involves encapsulating frames to render them apt for transmission through the physical medium and, in the event of collision executes resolution operations and retransmission (collisions may happen only with certain kinds of medium). The MAC is responsible for the arbitration within the network, and sorting packets to their destination. [21]

MAC Address

MAC addresses are essential elements for intra-network communication.
The MAC address is composed of 6 bytes (12 hexadecimal digits), grouped into two 24-bit long sections, the OUI and a serial number:

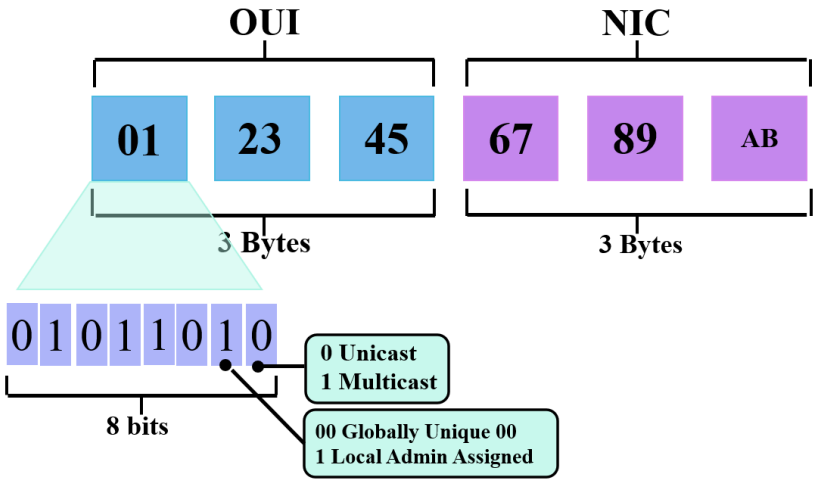

Figure 3.4: MAC address

- **OUI**: Organization Unique Identifier, occupies the 3 most significant bytes and carries the code, assigned by IEEE, that identifies the manufacturer.
- **NIC**: Network Interface Controller, occupies the 3 less significant bytes and contains a code assigned by the manufacturer and uniquely identifies the device.[22]

There are three types of MAC addresses:

- **Unicast**: this kind of address conveys a single LAN interface, the frame will be sent to a unique receiver.

- **Multicast**: this kind of address conveys multiple LAN interfaces, the frame will be sent to a group of devices. They are characterized by a 1 in correspondence to the least significant bit of the first octet of the address, hence an example for a generic structure of a multicast Ethernet address is 01:00:0C:CC:CC:CC

- **Broadcast**: this kind of address conveys all the interfaces within the LAN, the frame will be forwarded to all the devices. The broadcast address is FF:FF:FF:FF:FF:FF.[23]

The main parameters of the MAC sublayer are:

Slot time	512 bit time	Backoff time between retransmission
Inter Frame Spacing	96 bit time	Minimum distance between two packets
Attempt limit	16	Maximum number of retransmission attempts
Backoff limit	10	Number of attempts beyond which the randomness of the back-off no longer increases
Jam size	32 bit	jam sequence length
Max frame size	1518 bytes	Maximum package length
Min frame size	64 bytes	Minimum package length
Address size	48 bit	MAC address length

Table 3.1: MAC parameters source:[17]

3.3.2 LLC sublayer

The Logical Link Control sublayer acts as an interface between the MAC sublayer and the OSI network layer, allowing the coexistence of several multiple protocols such as IP, IPX, etc... Its main purpose is to transmit, via multiplexing, protocols sent through the MAC layer and demultiplexing them. [24] The IEEE 802.2 standard specifies the LLC sublayer for communication between network devices. Other features of the LLC layer are acknowledgment and error recovery via Automatic Repeat Request (ARQ). The receiving station acknowledges the reception of a frame, facilitating synchronization between the sending and receiving nodes leading to better flow control. Error-checking function ensures that no data was lost during the multiplexing or demultiplexing operations.[21] Ethernet lacks flow control or automatic repeat request (ARQ) mechanisms. This implies that erroneous packets are identified but not retransmitted, except in cases of collisions detected by the CSMA/CD MAC layer protocol. Retransmissions primarily depend on higher-layer protocols. [25]

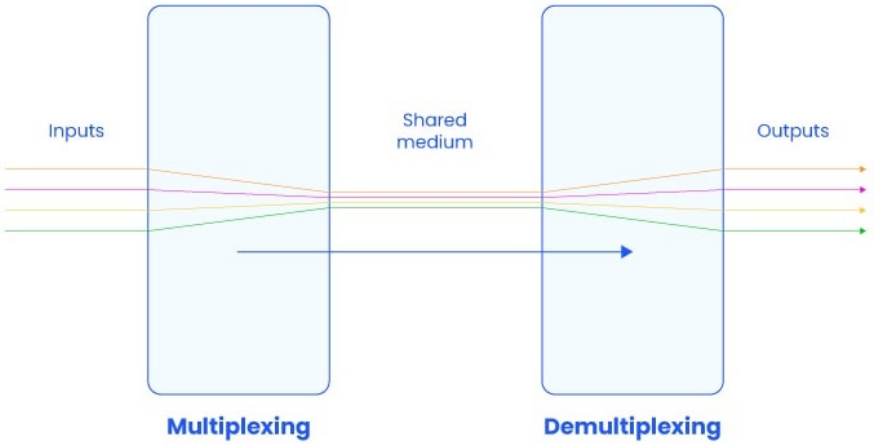

Figure 3.5: Multiplexing and Demultiplexing in LLC, source: [21]

3.4 Network Layer

The previous paragraphs described the first two OSI layers; as observed in the segmentation of the Data Link Layer into MAC and LLC sublayers. The Network Layer, finds a definition through the IP protocol described in this section. Meanwhile, the Transport layer could be characterized by either the TCP (Transmission Control Protocol) or UDP (User Datagram Protocol). The Application Layer is deployed by means of for instance HTTP (Hypertext Transfer Protocol), SNMP (Simple Network Management Protocol), SMTP (Simple Mail Transfer Protocol) and Profinet (developed by Siemens). [26]

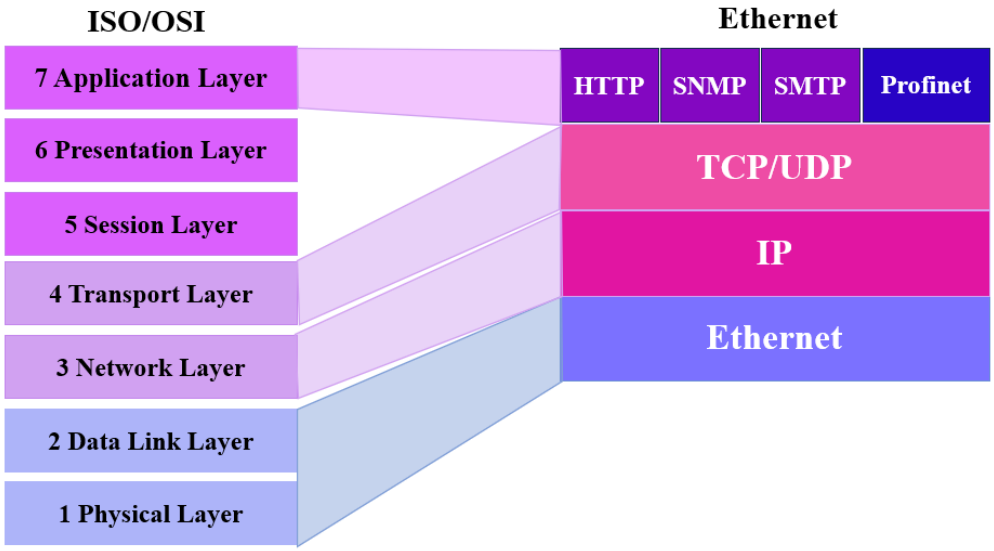

Figure 3.6: Overview of the OSI and Ethernet layers, source: [26]

3.4.1 IP: Internet Protocol

The Internet Protocol stands as an indispensable protocol in facilitating and harnessing the capabilities of the Internet. It constitutes an integral component within a suite of protocols, fostering communication among electronic devices free of vendor constraints and electronic system dependencies. Its first version can be dated to 1974, published in "Specification of Internet Transmission Control Protocol".
The main purpose of the IP is addressing which means forwarding packets from a source to its destination throughout different devices from networks to any place around the planet. Hence each IP address is univocally assigned and defined by Regional Internet Registers and made available thanks to the Internet Assigned

Numbers Authority. The most widespread versions of the Internet Protocol are the IPv4 and IPv6. [27]

IPv4

IPv4 was conceived in the first years of the 1970s by the Defence Advanced Research Projects Agency (DARPA) to improve the communication between diverse computer systems, furthermore, its standardization can be traced back to 1981. Nevertheless, IPv4 soon started showing its limitations such as scaling problems and a lack of a flexible transition mechanism for the contemporary internet; furthermore, the main problem was the shortage of address space that led, in the early 2000s, to a depletion of IPv4 addresses. The resolution of this problem has paved the way for IPv6 and IPng even though IPv4 (+NAT) is still widely used today. [28]
The main feature of IPv4 is identifying hosts from the logical address. In the year 1977, the specification for the IPv4 address was formalized, defining its length as 32 bits allowing 4.3 billion unique combinations. The address can be subdivided into two sections:

- **Network**: Defines a unique number associated with the specific network and outlines the assigned class.
- **Host**: Section that uniquely identifies a specific machine within the network.

An example of a configuration for an IPv4 address is shown in the picture below:

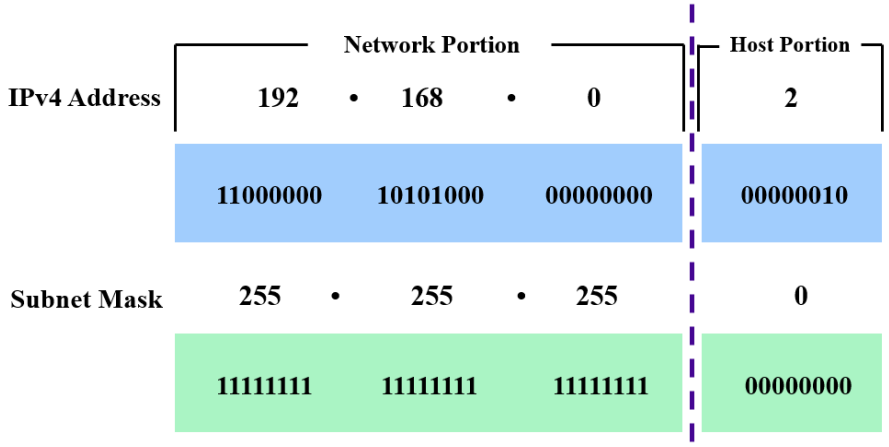

Figure 3.7: IPv4 address and subnet mask, source: [29]

The IPv4 addressing method allows a distinction between network hosts and addresses by means of a subnet mask(32-bit binary number); i.e. having a 255.255.255.0 subnet mask and a 192.168.0.2 IP address, the 2 in the last octet indicates the host while 192.198.0 specifies the class C to which it belongs. [30] IPv4 is subdivided into five classes: A, B, C, D, and E; A, B and C classes define a network host, class D encompasses multicasting while class E is allotted for upcoming applications. [30]

- Class A This class is meant to be used for large networks, the first bit encompasses the values 0.0.0.0 to 127.255.255.255. It allocates 8 bits for the network and 24 bits for the host.
- Class B This class is designed for medium/big networks, the first two bits encompass the values 128.0.0.0 and 191.255.255.255. It allocates 16 bits for the network and 16 bits for the host.
- Class C Intended for small LANs, uses three octets spanning from 192.0.0.0 to 223.255.255.255 and allocates 24 network bits, and 8 host bits. [31]
- Class D Used only for multicasting. Its range class spans from 224.0.0.0 to 239.255.255.255
- Class E Reserved only for experimental and study purposes. It ranges from 240.0.0.0 to 255.255.255 [27]

Network Class	Number of Networks	Number of Hosts
A	126	16.777.214
B	16.384	65.534
C	2.097.150	254
D	n/a	221.000.000
C	n/a	315.000.000

Table 3.2: Networks and Hosts for class A, B C, D, E source:[27]

IPv6

As already mentioned, the IPv4 is not sufficient to provide enough unique IP addresses. IPv6 was first introduced in 1997 by the Engineering Task Force (IETF) and it started being used in 2004; these types of addresses allow, in combination with IPv4, to identify sources and destinations of packets across the networks. An essential feature is the fact that it can co-exist with the IPv4 protocol. IPv6 provides a larger address space: 128 bits represented by 16-bit hexadecimal number sections.[30] An example for an IPv6 address is shown below:

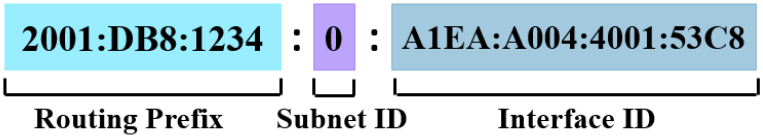

Figure 3.8: IPv6 address, source: [32]

- **Site Prefix** Comprises the first 48 bits, describes the public topology and is assigned by the Internet Registries and Internet Service Providers (ISPs)
- **Subnet ID** 16-bit long field, defines the private topology or site topology
- **Interface ID** Rightmost 64-bits, also known as token which can be manually configured

IPv4 and IPv6 can co-exist but cannot directly communicate, hence IETF defines three mechanisms: Dual Stack, Translation and Tunneling. The latter consists of the creation of a tunnel between IPv6 nodes that are linked through IPv4 nodes. The IPv6 sender encapsulates the IPv6 datagram within the data field of an IPv4 datagram; subsequently, this composite packet is forwarded through IPv4 routers to the receiving IPv6 whose task is to extract the IPv6 datagram from the received IPv4 datagram.[33]

For the automotive industry, the necessity of moving from IPv4 to IPv6 technology, might be impelling since vehicles connected to the external world have to be unequivocally identified in order to prevent any kind of issues in the future. However, if considering devices that only communicate within the vehicle, the usage of static IPv4 addressing might even be preferred since it has lower complexity with respect to IPv6. [27]

3.5 Transport Layer

The Transport Layer receives data from the lower Network Layer and sends data to the upper Application Layer. The transmission of information can happen by means of execution of either the TCP (Transmission Control Protocol) or the UDP (User Datagram Protocol).
A network socket is a software construct functioning as a termination point for the transmission and reception of data throughout the network.[34] It is the interface between the application and transport layer; characterized by a port number whose purpose is to allow TCP and UDP protocols to forward a specific application, or instance, within a terminal system. Port Numbers are composed of 16 bits and are defined according to the RFC1700 and RFC3232 standards. In particular, there are three different ranges, each one associated with a port typology:

1. Well-Known: port numbers in the 0-1023 range, reserved for specific applications.

Description	Application Protocol	Port Number
File transfer	FTP	20/21
Remote login	Telnet	23
E-mail	SMTP	25
Domain Name System	DNS	53
Host Name Server	Host NS	42
World wide web	HTTP	80
Remote e-mail access	POP-3	110
Simple Network Management Protocol	SNMP	161

Table 3.3: Well-known port numbers source:[35]

2. Registered: port numbers in the 1024-49.151 range, their usage is registered by IANA (Internet Assigned Numbers Authority).

3. Dynamic: port numbers in the 49.152-65.535 range. Their usage is totally free and dynamically assignable by applications. [35]

3.5.1 User Datagram Protocol

The User Datagram Protocol UDP is defined in the RFC768 protocol, it's a quite simple transport protocol since its main purpose is to multiplex many application functions into a single UDP flux. Furthermore, the UDP protocol is connectionless (no explicit connection setup is required before sending data), its main issue might be the impossibility of checking if some segments have been lost since they are not numbered.
The UDP segment is composed of four 16-bit long fields:

1. **Source Port**: associated with the application transmitting information.
2. **Destination Port**: contains the port number associated with the destination application.
3. **UDP length**: defines the total length of the UDP datagram.
4. **Checksum**: it's a code that allows checking whether the datagram contains any errors.

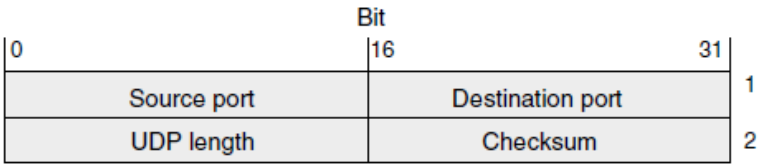

Figure 3.9: UDP Header, source: [35]

The multiplexing function is carried out thanks to the information contained in the source port and destination port fields. The picture 3.10 gives an example of interaction between two hosts H1 and H2 and a server S1 that provides three services DNS, RIP, and SNMP. H1 accesses to DNS and RIP services through port number 4400 and 8800, while H2 accesses to SNMP through port number 8800. The port numbers associated with the three services are 53 for DNS, 520 for RIP and 161 for SNMP. Hence each one of the UDP segments will contain these identifiers:

<H1,4400>, <S1,53>
<H1,8800>, <S1,520>
<H2,8800>, <S1,161>

In this scenario, there is no problem of ambiguity during the demultiplexing process, even if two different hosts select the same source port number. The destination port (PND) effectively identifies the receiving process, and the server uses the identifying pairs to accurately direct response segments to each request received from clients.[35]

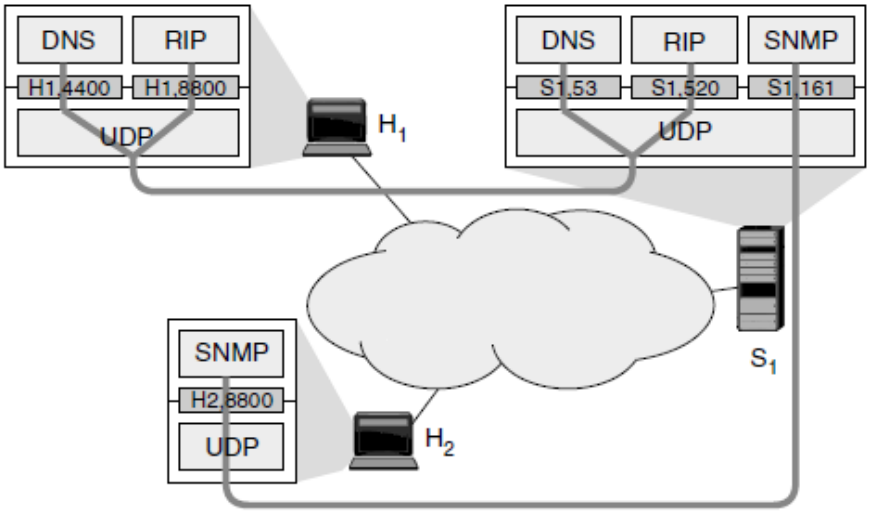

Figure 3.10: Multiplexing within UDP protocol, source: [35]

3.5.2 Transmission Control Protocol

The Transmission Control Protocol TCP provides a connection-oriented service, hence, unlike UDP, in the event of occurrences like loss, duplication, or the delivery of information units in a flow to the destination TCP system, the system carries out all essential functions to reconstruct the initially emitted flow. This reconstruction relies on a systematic numbering of the byte positions.

TCP owes its reliability to the byte-by-byte numbering of transferred data in both directions of the connection established between terminal systems. Two buffers are employed: a transmission one, Tx, and a receiving one, Rx. Tx stores the bytes received from the application ready to be transmitted over the TCP connection as segments; while the Rx stores the received segments along the connection, waiting for the delivery of informative bytes to the destination application process through the respective socket.

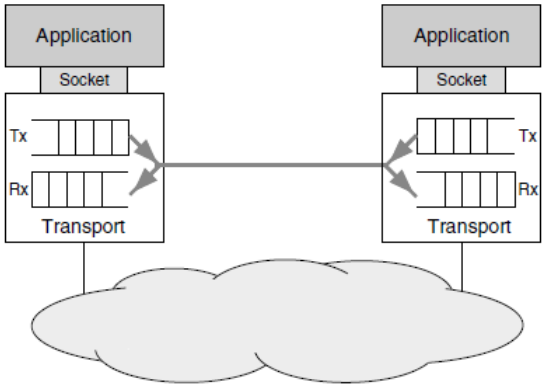

Figure 3.11: Tx and Rx buffers in TCP, source: [35]

The TCP header is composed of 11 fields:

1. **Source Port**: (16 bits), port number linked to the application generating the information unit.

2. **Destination Port**: (16-bits) port number linked to the application receiving the information unit.

3. **Sequence Number**: SN (32 bit) sequence number of the initial data byte within the payload of the information unit in the transmitted flow from the source.

4. **Acknowledgment Number**: AN (32 bits) sequence number of the upcoming byte anticipated by the terminal system in the next segment; for example, if AN = x, the terminal system has acknowledged all received bytes up to x – 1.

5. **HLEN**: (4 bits) header length; since the TCP segment header exhibits variable length, ranging from a minimum of 20 bytes to a maximum of 60 bytes; thus, the HLEN field can take values between 5 and 15;

6. **Reserved**

7. **Code bits**: (8 bits), each one of those bits has a specific purpose:
   CWR (congestion window reduced): if CWR=1 signals to the remote TCP that the local TCP has reduced the "congestion window, thus slowing down the flow of data sent over the TCP connection.
   ECE (ECN-Echo): if ECE = 1 tells the remote TCP that the local TCP has received an indication of congestion from the network (ECN, Explicit Congestion Notification), requesting a slowdown in the flow of data.
   URG (Urgent): if URG = 1 indicates the presence in the TCP segment of

data to be urgently delivered to the destination, with its position indicated by the urgent pointer field.
ACK (Acknowledgment): indicates whether the acknowledgment number is valid (ACK = 1) or if it should be ignored
PSH (Push): when PSH = 1 requests the receiving terminal to deliver the payload of the segment to the destination application, regardless of the content of the receiving buffer.
RST (Reset): RST = 1 signals the release of the connection, rejection of a connection request, or rejection of a segment.
SYN (Synchronize): SYN = 1 to establish a TCP connection; FIN (Finalize): FIN = 1 indicates the request for the release of the TCP connection.

8. **Window**: (16 bit) accounts for opening the receiving window, that is, the number of bytes that the source of the TCP segment is willing to receive from the remote TCP and implementing an end-to-end flow control mechanism.

9. **Checksum**: (16 bit) holds for the error detection on the TCP segment.

10. **Urgent Pointer**: UP (16 bits) indicates where urgent data is positioned within the segment; specifically, it denotes the offset in terms of byte count from the sequence number SN = z of the message where the urgent data concludes.

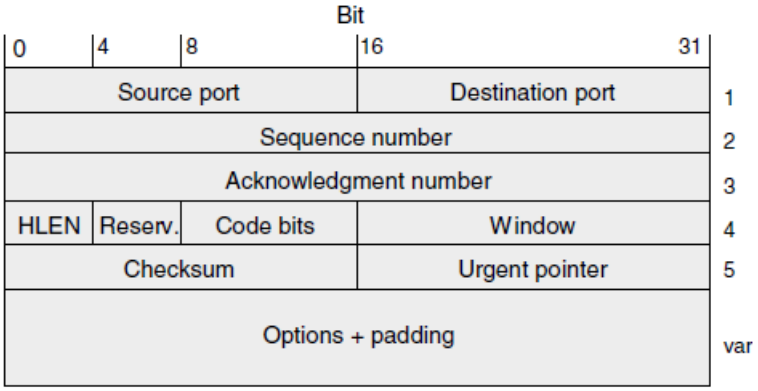

Figure 3.12: TCP header, source: [35]

11. **Options**: field of variable length, but always a multiple of 32 bits using the padding field, allowing the exchange of additional information between the ends of the connection to perform specific functions. The generic format of the options field, includes three components: kind (the type of option), length (total length in bytes of the option), and option data (content of the option).

As already mentioned, TCP operates as a connection-oriented protocol, indeed the exchange of information units between applications necessitates the establishment of a TCP connection in advance. Throughout this process, a socket pair is linked to the TCP connection, defining the users of the service. As a result, the connection is characterized by the quadruple <NA(s), PN(s), NA(d), PN(d)>. For example, having a server S1 (providing access to the browsing service through HTTP protocol and email through SMTP) and two hosts H1 (whose clients have access to HTTP via ports 2002 and 4004) and H2 (whose clients have access to HTTP via port 4004 and SMTP via port number 3003). The port number associated with the HTTP service is 80 and with SMTP is 25. The four connections are defined by:

<H1,2002,S1,80>
<H1,4004,S1,80>
<H2,4004,S1,80>
<H2,3003,S1,25>

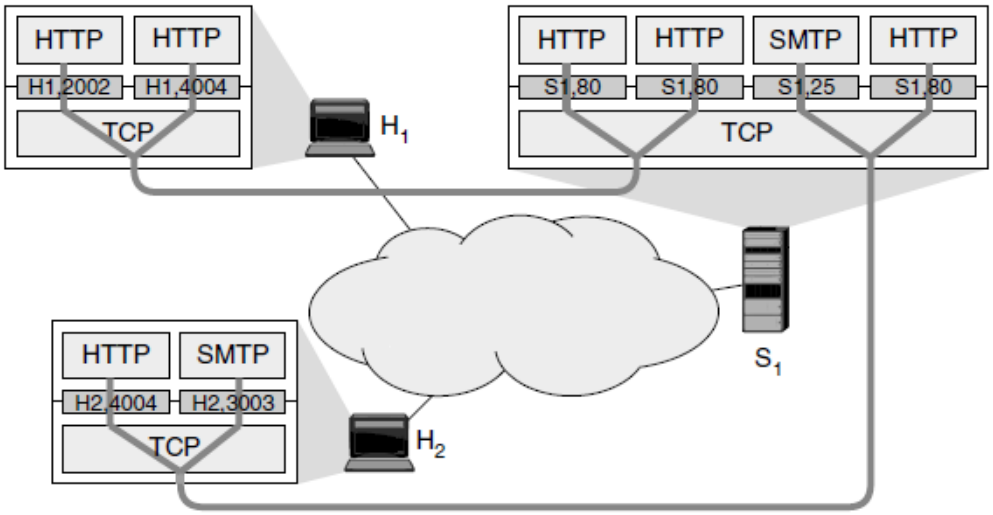

Figure 3.13: TCP multiplexing, source: [35]

3.6 Application Layer

The application layer is the highest layer in the architecture defined by OSI. The interactions within the various applications are defined either by the "client-server" or the "peer-to-peer" mechanism. The application protocols delineate the methods of communication by means of messages through which applications interact with one another.
Peer-to-peer paradigm (P2P) contemplates that all the processes, called peers, can assume at the occurrence both the role of server and client.
Typically, the interaction between applications follows the client-server (C-S) paradigm: one of the two processes is the server, which provides information and services; the other one is the client which requests these last through request messages. One of the main issues of the client-server paradigm is scalability: when the number of clients grows exponentially, the server must cope with the requests of all clients. The most common application protocols are HTTP, SMTP, SNMP, and DNS. [36]

3.6.1 HTTP

The HyperText Transfer Protocol is defined in the RFC2616 protocol. It defines the interaction between the web server and the browser (client) by means of request and response messages.
The first version of HTTP was proposed in 1991 by Barnes-Lee and was the HTTP 0.9, the main characteristics are:

- Client-Server protocol
- Requests and Responses were single ASCII streams
- Transferring hypertexts documents
- Server-Client connection is closed after every request

HTTP/1.0 was proposed in order to overcome the mere delivery of hypertext documents; this protocol offers enhanced metadata concerning both the request and the response, facilitating content negotiation, and encompassing additional functionalities, in 1996, RFC 1945 was published defining the common usage of HTTP/1.0 integrations.
HTTP/1.1 was introduced for the first time in RFC 2068 in 1997 and addresses numerous protocol issues present in its predecessors while introducing pivotal performance enhancements. These include the incorporation of additional caching mechanisms, the implementation of keepalive connections, support for byte-range requests, utilization of chunked encoding transfers, adoption of various transfer

encodings, and the introduction of request pipelining. [37] Regarding the HTTP/2 RFC7540], the main focus is on the mode of data transport between the client and server with the aim of improving latency performance in retrieving documents from web servers, also reducing the transport overhead due to HTTP message headers. The main features of HTTP/2 are:

- Multiplexing on TCP Connection: HTTP/2 allows the multiplexing of multiple TCP sessions on a single TCP connection, reducing the need to create multiple sockets.
- Resolution of HOL Blocking: Introduces a "framing" mechanism that breaks down each HTTP message into a sequence of frames, improving the delay associated with head-of-line (HOL) blocking.
- Request Prioritization: Implements a priority mechanism, allowing the client to assign priority levels to requests to influence the order of frame transmission by the server.
- Server Push: Introduces a mode where the server can proactively send objects to the client without an explicit request, reducing the overall transfer time of documents.
- Efficiency Through Encoding: Optimizes the use of transmission capacity through encoding techniques, such as Huffman encoding for transferred strings and efficient handling of headers.
- Continuity with HTTP/1.1: Emphasizes that despite the version increment, the high-level semantics of the protocol, including HTTP headers, values, and use cases, remain unchanged.[36]

3.6.2 SOME/IP

Scalable service-Oriented MiddlewarE over IP operates at the highest OSI layers and accounts for the transmission of complex data between software elements. The term “middleware” indicates all those functions that facilitate the exchange of information between software components that normally would be decoupled. The main requirements for SOME/IP are:

- Compatibility with AUTOSAR
- Small Footprint
- Scalabity
- Service-based communication

- Flexibility regarding most common operating systems for the automotive industry such as AUTOSAR, OSEK, QNX and Linux[27]

SOME/IP header

The SOME/IP header is composed as follows:

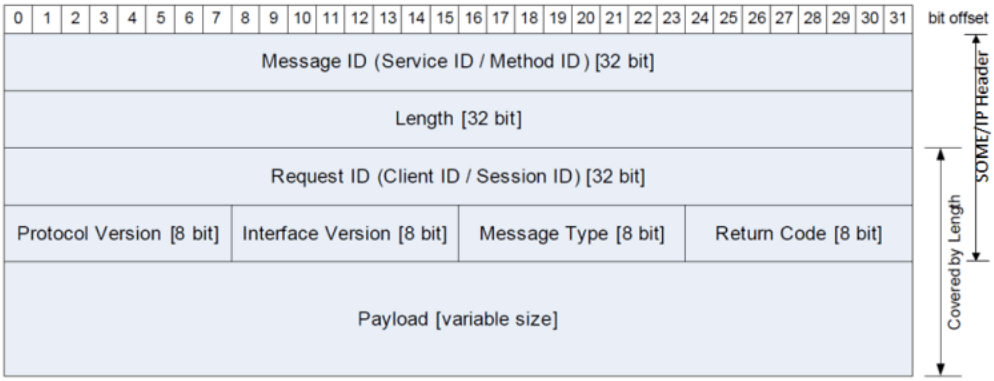

Figure 3.14: SOME/IP header, source: [38]

- **Message ID**:32-bit long field that identifies the RPC call to a method of an application. It is subdivided into Service ID (16 bit) and Method ID(16 bit).
  - **Service ID**: indicates the overall structure of the middleware communication.
  - **Method ID**: defines methods, events, and fields that compose a service
- **Length**: 32-bit long field that indicates the length in bytes contained from Request ID/Client ID until the end of the SOME/IP message
- **Request ID**: (32-bit long) enables a provider and subscriber to distinguish among various simultaneous applications of the same method, event, getter, or setter. Request IDs are unique for each provider-subscriber couple. The Client ID (first 16 bits) identifies the client, the other 16 bits are the Session ID which serves as a distinctive identifier, facilitating the differentiation of sequential messages or requests originating from the same sender.
- **Protocol Version**: 8-bit long field, identifies the used SOME/IP Header format (the one currently used is the version 1).
- **Interface Version**: (8 bits) indicates the Major Version of the Service Interface.

- **Message Type**: (8 bits) employed to distinguish various message types and is expected to include the following values:

Number	Value	Description
0x00	REQUEST	Request expecting a response
0x01	REQUEST_NO_RETURN	Fire and forget request
0x02	NOTIFICATION	A request of a notification/ event callback expecting no response
0x80	RESPONSE	The response message
0x81	ERROR	The response containing an error
0x20	TP_REQUEST	A TP (Transport Protocol) request expecting a response (even void)
0x21	TP_REQUEST_NO_RETURN	A TP fire&forget request
0x22	TP_NOTIFICATION	A TP request of a notification/ event callback expecting no response
0x23	TP_RESPONSE	The TP response message
0x24	TP_ERROR	The TP response containing an error

Table 3.4: Message Types, source:[38]

- **Return Code**: is employed to indicate whether a request has been successfully processed.
- **Payload**: This field has a variable size and contains SOME/IP messages parameters. In the case of UDP, the SOME/IP payload length is expected to range between 0 and 1400 Bytes. The limitation to 1400 Bytes is necessary

to accommodate potential future modifications to the protocol stack, such as transitioning to IPv6 or incorporating additional security measures. Given that TCP supports payload segmentation, larger sizes are inherently supported.[38]

Communication Principles

SOME/IP determines a service by means of Service Interfaces that can be subdivided in three categories:

- **Request/Response**: One of the most prevalent communication patterns. A communication participant (Client) dispatches a request message, and this is reciprocated by another communication participant (Server). To form the payload of a request message, serialize all input or in-out arguments of the method in accordance with the order specified in the method's signature while to create the payload of a response message, serialize all output or in-out arguments of the method based on the order of the arguments within the method's signature.

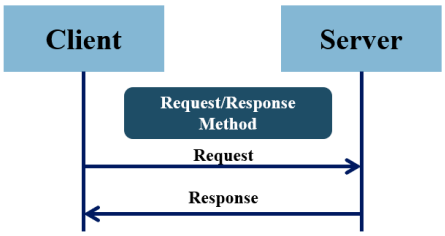

Figure 3.15: Request/Response method, source: [27]

- **Fire&Forget**: It's a method characterized by only Request messages, the client does not expect a response.

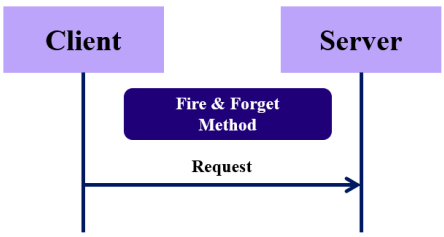

Figure 3.16: Fire & Forget method, source: [27]

- **Events**: Notifications essentially encapsulate a Publish/Subscribe concept. Typically, a server publishes a service, and a client subscribes to it. In specific

instances, the server dispatches an event to the client, such as an updated value or a triggered occurrence. It's important to note that SOME/IP is exclusively employed for transporting the updated value and is not involved in the actual mechanisms of publishing and subscription. These mechanisms are managed and executed by SOME/IP-SD. In situations where there are multiple subscribed clients on the same ECU, the system is required to manage the replication of notifications. This ensures efficiency in transmissions over the communication medium, particularly when notifications are conveyed through multicast messages. [38]

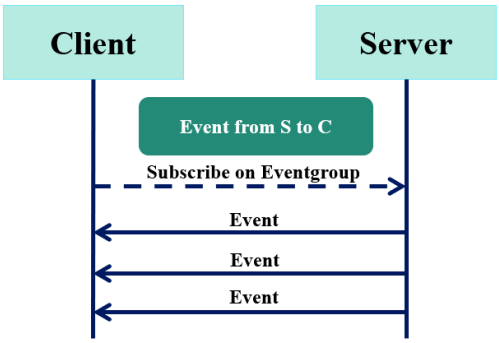

Figure 3.17: Events method, source: [27]

- **Fields**: A field denotes a status and holds a value. Subscribers who subscribe to the field promptly receive the field value as an initial event. Unlike events, which are only valid within the time range they're happening, fields can be accessed remotely and, furthermore, they can even be set by the client [27].

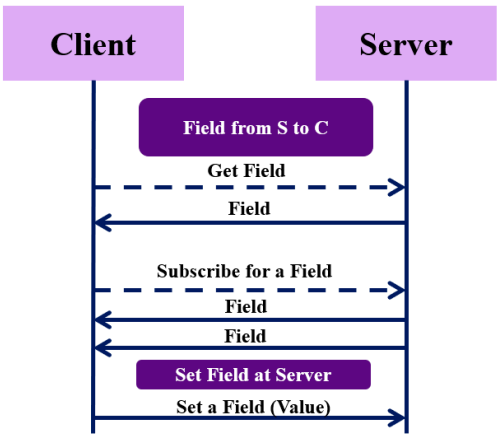

Figure 3.18: Fields method, source: [27]

3.7 A Further Insight on Protocols for Automotive Ethernet

Within the spectrum of services provided by Ethernet-based communication networks, a noteworthy feature is the potential for repurposing protocols that have previously undergone successful development and testing within other industries. The illustration below delineates a protocol stack, with a particular focus on AVB and ARP, which are two protocols intricately involving OSI layers 3 to 7, alongside TSN, representing the evolutionary progression from AVB. [8]

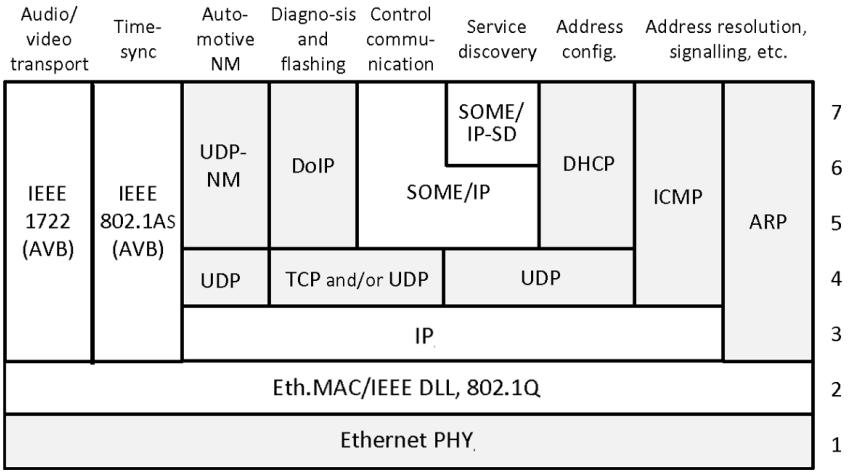

Figure 3.19: Example of a protocol stack for Automotive Ethernet, source: [39]

3.7.1 Audio Video Bridging

AVB is a set of technical standards designed to enhance synchronization, minimize latency, and bolster reliability within switched Ethernet networks. Regarding the application for automotive production, it may be employed to facilitate concurrent audio streams of varying priorities, synchronize the playback of audiovisual content, and manage camera data for driver assistance purposes. Its paramount focus lies in enhancing driving safety.
AVB comprises "Talkers", which transmit data streams, and "Listeners" which receive, both of them are connected by means of AVB Switches. A critical guideline in the configuration of an AVB network is ensuring that the talker and the listener are both linked to a switch compatible with AVB. Additionally, all AVB devices within the network must synchronize their operations through a common virtual clock, precisely dictating the timing for the playback of AVB packets. AVB

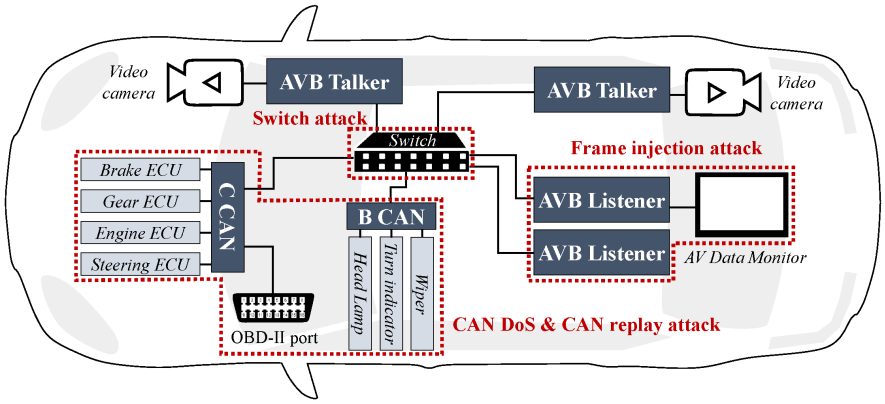

Figure 3.20: Example of Talker-Listener AVB system within a car, source: [40]

essentially operates by appropriating a dedicated segment of the accessible Ethernet bandwidth exclusively for its own data transmission. By dispatching AVB data packets at regular intervals within these allocated time slots within the reserved bandwidth, the process ensures an absence of interruptions or interference, thereby endowing AVB with an exceptionally high degree of reliability. real-time traffic is transmitted in synchronization with an 8 kHz pulse, at precise intervals of 125 µs, all real-time streams seamlessly dispatch their data ensuring that prioritized real-time traffic is accorded ample bandwidth through the application of the Stream Reservation Protocol.[41]
First generation Audio Video Bridging is associated with a set of standards:

- **IEEE 802.1Qav:** "Forwarding and Queuing Enhancements for Time-Sensitive Streams" whose objective was to enhance the quality of the transmission by equally distributing over time the packets of each individual stream.
- **IEEE 802.1Qat:** "Stream Reservation Protocol (SRP)", facilitates the precise allocation of bandwidth within the AVB cloud, enabling dedicated provisioning for individual applications and distinct traffic streams.
- **IEEE 802.1AS:** "Timing and Synchronization for Time-Sensitive Applications"; its primary objective is to synchronize all nodes within an AVB cloud to a unified reference time, in particular, direct neighbor nodes have to be synchronized with each other within nanoseconds.
- **IEEE 1733.2011:** "Protocols for Time-Sensitive Applications in Local Area Networks" enhances the adaptability of technologies leveraging AVB networks by delineating the methodologies for transporting AV streams over layer 3/IP-based networks.

- **IEEE 1722:** "Transport Protocol for Time-Sensitive Applications in a Bridged Local Area Network" facilitates interoperable streaming through the definition of several key elements, including:
  - Media formats and encapsulations for both raw and compressed audio/video formats.
  - Bridging IEEE 1394 LANs over AVB networks.
  - Mechanisms for media synchronization.
  - Reconstruction and synchronization of media clocks.
  - Latency normalization and optimization.
  - Assignment of multicast addresses.
  - Allocation of AVB Stream IDs.
  - Determination of the media clock master. [42]
- **IEEE 802.1BA -2021:** "IEEE Standard for Local and Metropolitan Area Networks–Audio Video Bridging (AVB) Systems" establishes profiles and default configurations to facilitate straightforward management, particularly in situations where in-depth network expertise cannot be assumed.
- **IEEE 1722.1-2021:** "Device Discovery, Connection Management, and Control Protocol for 1722 Based Devices", this standard outlines the protocol, device discovery, connection management, and device control procedures designed to enable interoperability among audio and video-based End Stations utilizing IEEE 1722-based Streams on IEEE 802-based networks. [27]

3.7.2 Time Sensitive Networks

The presence of several sensors within the car has highlighted the demand for accurate computation execution timing; hence, since 2012, IEEE has proposed some standards for time synchronization within time-sensitive applications from audio/video to various types of flows which may have more stringent requirements. Time Sensitive Networks is a set of standards, developed by IEEE, that introduces some protocols characterized by specific properties such as low latency, reliability, robustness, and clock synchronization. TSN ought to be seamlessly incorporated into contemporary model-based development processes for embedded systems in automotive applications. It aims at broadening the IEEE 802.1Q switches.[3]
The most relevant TSN standards are:

- 802.1Qca-2015: Path Control and Reservation
  Defines the configuration of multiple paths within bridged networks; reserving bandwidth, offering data protection and redundancy, and disseminating flow synchronization and flow control messages

- 802.1Qbv-2015: Enhancements for Scheduled Traffic (further incorporated into 802.1Q
  Provides bounded low latency and even deterministic behavior for frame transmission that follows an accurate time schedule. It introduces the transmission gate mechanism.

- 802.1Qbu-2016: Frame Preemption
  Enhances the capacity to minimize access delay time for critical traffic Implemented by means of dual-MAC stack within IEEE 802.1Q to assure frame preemption for distributed real-time frames.

- 802.1Qci-2017: Per-Stream Filtering and Policing
  Its objective is to improve reliability, ensuring error detection and confinement.

- 802.1Qch-2017: Cyclic Queueing and Forwarding
  For receiving and broadcasting frames alternatively for a fixed interval of time.

- 802.1QCB-2017: Frame Replication and Elimination for Reliability
  Proposes frame replication and elimination for reliability (FRER) transmitting along detached paths.

- 802.1Qcc-2018: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements
  This amendment introduces protocols, procedures, and managed objects designed for bridges and end stations that maintain compatibility with existing mechanisms while offering the following enhancements: support for a higher number of streams, improved description of stream characteristics, support for Layer streaming, and deterministic stream reservation.

- 802.1Qcr-2020: Asynchronous Traffic Shaping (ATS)
  Full duplex links with constant bit data rates, it can handle the traffic bursts generated by radars and LiDARs by applying token-bucket shaping on a per-flow basis.

- 802.1AS-2020: Timing and Synchronizations for Time-Sensitive Applications
  This standard delineates the protocols, procedures, and managed objects employed for conveying timing information across local area networks. It encompasses the transmission of synchronized time, the determination of the optimal timing source, and the indication of instances and extents of timing disturbances, specifically phase and frequency discontinuities.

- P802.1DG: Time-Sensitive Networking Profile For Automotive In-Vehicle Ethernet Communications

Outlines profiles tailored for the establishment of secure, reliable, and deterministically low-latency Ethernet networks within automotive in-vehicle environments. These profiles are specifically crafted to offer guidelines for designers and implementers. They address the spectrum of in-vehicle applications such as those necessitating security measures, high availability, reliability, maintainability, and precisely bounded latency.

- P802.1AEdk: MAC Privacy protection
  Assures confidentiality in communications and initiates measures in response to security breaches.

Within the subsets of Time Sensitive networks, we have AV Bridging which is characterized by soft real-time systems specifications providing bounded latency. AVB can hold for the broadcastings of audio/video streams in the multimedia domain but appears to be unfitting for safety- and time-critical applications.

TSN Automotive System Development via Model-Based Software

In the past years, multiple domain-specific modeling procedures and languages have been proposed to accommodate the development of CAN, LIN and FlexRay networks: AUTOSAR, AMALTHEA, Rubus Component Model, EAST-ADL, CORBA, COMDES. However, among these, AUTOSAR combined with SymTA/S is the only one suitable for modeling high-bandwidth networks, but there is still a lack of support for modeling TSN. To overcome this problem a comprehensive approach based on the Rubus Component Model (RCM) has been proposed and, in alternative to this, some developers are working on a Unified Modeling Language (UML). [5]

3.7.3 Address Resolution Protocol

The TCP/IP protocol comprises three addressing modalities: Port Number, Internet Protocol (IP) and Media Access Control (MAC); the Address Resolution Protocol (ARP) plays a pivotal role in associating the IP address with the MAC address. Indeed MAC address serves as the identifier for a host within the confines of the Local Area Network (LAN), for an efficient packet delivery, it is crucial that the packet reaches its intended destination IP. If the designated IP corresponds to an address on the Internet, the host requires the MAC address of the gateway (router). Nevertheless, if the destination IP resides within the same LAN, the host still necessitates the MAC address, but this time for the local host. Every entry in the ARP table is assigned an expiration time, prompting the host to issue an ARP request for each entry. Notably, the Address Resolution Protocol (ARP) operates as a stateless protocol.[43]

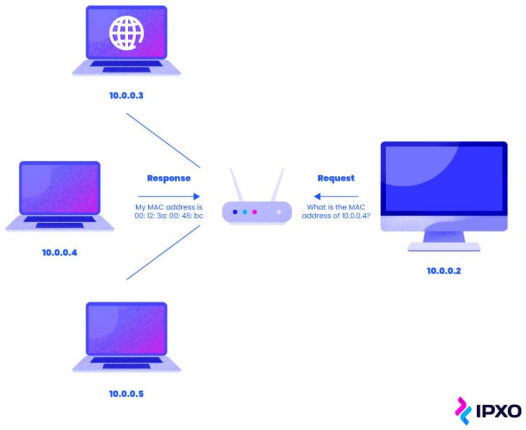

Figure 3.21: ARP communication, source: [44]

Whenever a source device intends to transmit an IPv4 packet to another device, the ARP protocol stack examines the ARP cache table, which is a repository of mappings between IPv4 addresses and corresponding MAC addresses. If the ARP cache lookup fails to yield a matching MAC address, the second task comes into play. In this scenario, the source server generates an ARP message, which is subsequently broadcast across the local area network (LAN). The ARP table serves as a repository containing records of both the IP addresses and MAC addresses of devices interconnected within the same network, the ARP protocol autonomously supplements the table with new entries upon receiving an ARP response.
The types of Address Resolution Protocol that are mostly used in Ethernet are:

1. **Proxy ARP**: The Proxy ARP protocol is designed to manage requests originating from IP addresses outside the local area network. In scenarios where the request packet originates from a system beyond the host's network, the router configured with this protocol takes on the responsibility of responding to the ARP request packet. Instead of furnishing the MAC address of the target host, the proxy ARP responds by assuming the identity of the destination providing its own MAC address, and enabling the router to act as an intermediary, facilitating communication between systems on disparate networks.

2. **Gratuitous ARP**: functions as an unsolicited ARP response, its purpose is to assist when a host needs to broadcast or update its IP address to MAC address mapping to the entire network. This proactive announcement helps ensure that other devices on the network are aware of the host's updated or current IP-to-MAC mapping, contributing to network efficiency and accuracy.

Furthermore, one of the main problems concerning ARP is "Spoofing" which is a malicious attack due to the transmission of counterfeit ARP messages to a targeted Local Area Network (LAN). The primary objective for the attackers is to associate their own MAC address with the IP address of a genuine device or server within the network, enabling the redirection of data from the victim's computer to the attacker's computer rather than its intended destination. ARP spoofing attacks present substantial risks, as they enable the covert transmission of sensitive information between computers without the awareness of the victims. The deceptive manipulation of ARP tables in this manner undermines the integrity of network communication, potentially resulting in unauthorized access and the interception of data. [45]

Chapter 4

IEEE standards for Automotive Ethernet and 10BASE-T1S Ethernet

Ethernet attained standardization within IEEE 802.3, delineating comprehensively both the physical layer (PHY) and the data link layer (DLL). Nonetheless, this depiction finds augmentation through additional standards such as IEEE 802.1 which defines the deployment of Ethernet-based communication systems, and IEEE 802.2 which standardizes the Logical Link Control layer. [46] IEEE 802.1 working group develops standards focusing on 802 LAN/MAN architecture, security and management, interfacing with higher layer protocols and internetworking routing and bridging within 802 LANs. [47]

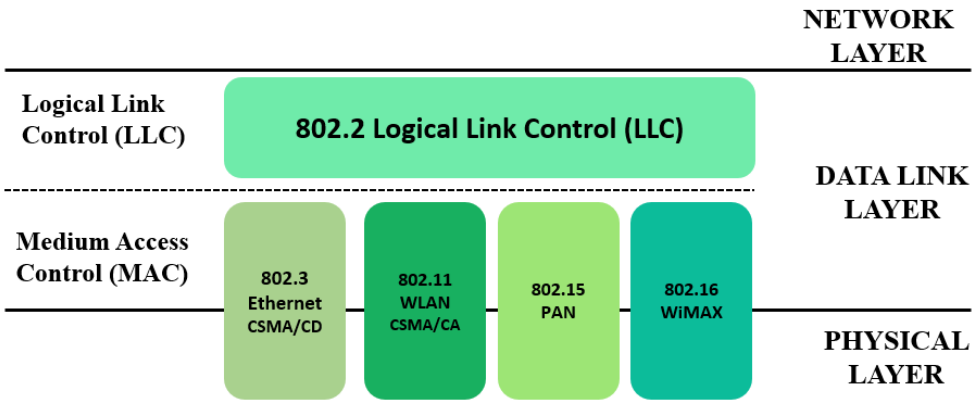

Figure 4.1: IEEE standards regarding Physical and Data Link layers, source: [48]

IEEE 802.3 defines a fieldbus architecture based on coaxial cable, nevertheless in recent times technologies based on UTP cables and optical fibers have gained ground. This standard involves the first OSI layer and the MAC sublayer defining, for the latter, a specific protocol called CSMA/CD.

4.1 Ethernet Standards

Ethernet standards are defined by the IEEE working group, and each one of them is distinguished by unique attributes regarding properties, implementations, and media types. Notable parameters include transmission speed and type, and the corresponding cabling specifications.

- **10Base2** Also known as ThinNet, based on coaxial cable, offering 10 Mbps speed. Its maximum length is 185 meters.
- **10Base5** Also known as ThickNet, based on coaxial cable, offering 10 Mbps speed. Its maximum length is 500 meters.
- **10BaseT** Based on UTP (unshielded twisted pairs, best solution to reduce electromagnetic interference) cables and Hubs (star + logical bus topology). According to this standard, the maximum number of Hubs in a network between workstations is four to guarantee that every station within the network is capable of detecting a collision.
- **10BaseF** Similar to 10BaseT, implemented over fiber optic cabling. It is commonly used to connect hubs to each other and to workstations.
- **100BaseT4** Enhancement of the 10BaseT, based on Cat3 wiring, providing 100 Mbps speed. Composed of four twisted pairs of wires; two of them implement a half-duplex transmission (flow of data only in one direction at time), and the other two implement a simplex one (flow of data always in one direction only).
- **100BaseTX** Also known as Fast Ethernet. Has the same topology as the 10BaseT but provides 100 Mbps data transmission speed. Composed of one wire pair that transmits data and another one that receives data.
- **100BaseFX** Fast Ethernet over fiber. Implemented by means of multimode fiber cables, due to dispersion its length is quite limited.
- **1000BaseT** Gigabit Ethernet, composed of Cat5 and other UTP cables, has physical star topology and a logical bus. It is compatible with both full-duplex and half-duplex data transmission.

- **10GBaseT** 10 Gigabit Ethernet, constituted of Cat6 and higher grade UTP cables. Exclusively operates in full-duplex mode, providing 10 Gbps speed. [49]

4.2 10BASE5 and 10BASE2

The introduction of 10BASE5 Ethernet occurred in 1980. However, the employed coaxial cable is characterized by its limited flexibility, hence the nomenclature "thick Ethernet". To address this limitation, the year 1985 witnessed the introduction of 10BASE2, distinguished by the utilization of a slender coaxial cable. As a result of the surging demand for high-speed networks, the affordability of Category 5 cable, and the widespread adoption of 802.11 wireless networks, the gradual phase-out of both 10BASE2 and 10BASE5 is underway.[50]
The term BASE stands for baseband, this term indicates that the transmission signal traverses the network using a carrier wave, employing Manchester encoding, characterized by a frequency of 20 MHz. This frequency serves as the pathway through which binary information is conveyed as it transitions between nodes within the network. In Manchester encoding, a binary 1 is denoted by a high-to-low transition in the middle of the bit period, and a binary 0 is signified by a low-to-high transition midway through the bit period. This encoding method facilitates clock recovery from the signal. Nevertheless, the extra transitions inherent in Manchester encoding result in a doubling of the signal bandwidth. [51]

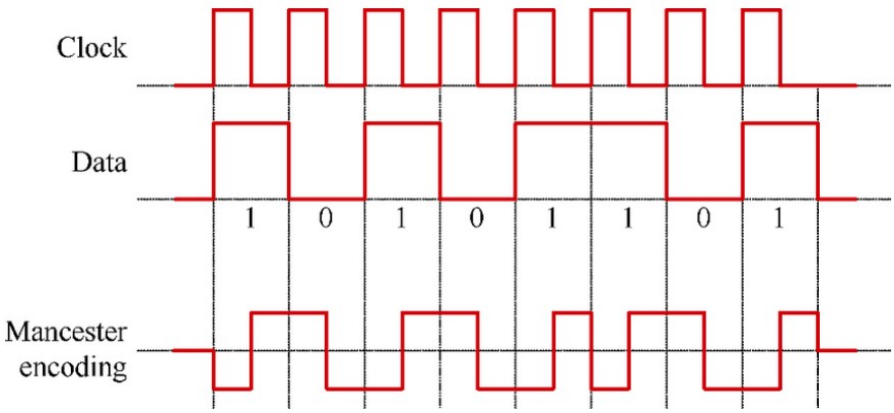

Figure 4.2: Manchester encoding principles, source: [52]

4.2.1 10BASE5

10BASE5 was standardized in 1982 and was the first available variant of Ethernet. A 10BASE5 network supports a coaxial cable with a maximum length of 500 m and can accommodate up to 100 nodes. The nodes are connected using vampire connectors collocated at a minimum distance of 2.5 meters apart. The chosen distance was deliberately set to avoid alignment with the wavelength of the signal. This precaution ensures that reflections from multiple taps do not synchronize or align in phase. A terminator, also known as a termination plug, must be mandatory installed at both ends of this segment. This terminator is a specific type of connector designed to prevent the reflection of an incoming signal, thereby avoiding collisions with other transmitted impulses. The link between the transceiver and workstation is established through an 8-pair cable, commonly referred to as a drop cable. The reason for this additional branch cable is that the rigid coaxial cable used in 10Base-5 networks can only make very wide turns and is difficult to shape in a way that approaches all the existing nodes. [53]

4.2.2 10BASE2

10BASE2 can connect a maximum of 30 nodes; within this type of network, every cable segment links to a transceiver, typically integrated into a network adapter, utilizing a BNC T connector. Each female T connector connects to one section of the cable. Like other high-speed buses, proper termination of the Ethernet segment is crucial, involving the placement of a 50-ohm resistor on both ends. Typically, this resistor is incorporated into a male BNC connector, connected to the final device on the bus. In the absence of proper termination or in the occurrence of a broken cable, the AC signal on the bus undergoes reflection rather than dissipation upon reaching the termination point. This reflected signal mirrors a collision, rendering communication impossible. Grounding one end of the cable is essential; however, grounding both terminators can lead to a ground loop, potentially causing network outages and data corruption. When configuring a 10BASE2 network, it is required to ensure proper connections to all T-connectors. Detecting issues like poor contacts and short circuits can be particularly challenging. A failure at any point in the network cabling tends to disrupt overall communications. Consequently, 10BASE2 networks are usually replaced by more manageable 10BASE-T networks. [51]

4.3 10BASE-T

During the 1980s, cost-effective twisted-pair wires gained extensive use for implementing 10BASE-T, offering easier installation compared to the bulkier thick and thin coaxial cables. Notably, 10BASE-T Ethernet pioneered the adoption of a star architecture. It facilitated 10-megabit-per-second transmission speeds over twisted-pair cabling, with a maximum length of 100 meters. Typically, 10BASE-T

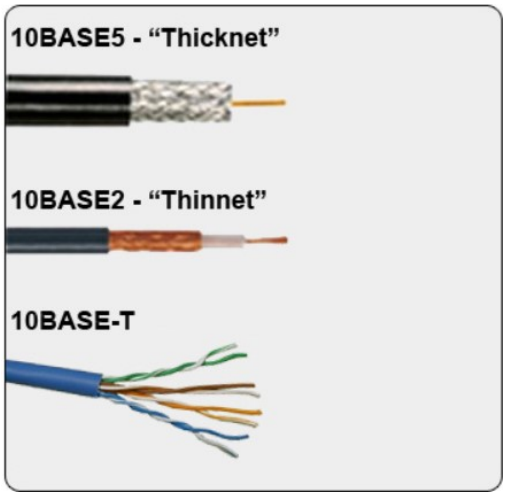

Figure 4.3: Comparison between coaxial cables used in 10BASE2 and 10BASE5 and twisted pair in 10BASE-T

networks employ a star topology, with connections centralized to a hub. Wiring is organized and terminated at patch panels while patch cables link each panel port to the central hub. The wiring is neatly concealed within a dedicated cabinet or arranged on a rack for convenient accessibility. Given the 100-meter maximum cable length in a 10BASE-T network, repeaters become necessary for extended distances. Connecting two or more segments, each with a minimum length of 2.5 meters, requires repeaters. Regular or stackable hubs for cascading support up to 1,024 nodes, but optimal performance and collision prevention are achieved with 200 to 300 nodes. Termination of 10BASE-T cables occurs through a Mod-Tap or a similar wall jack to establish connections inside the wall.[54]
Expanding 10BASE-T is feasible by linking new hubs, and the failure of one hub connection doesn't inherently jeopardize other connections to the hub. Nevertheless, the hardware costs for 10BASE-T tend to be higher owing to the demand for multiple hubs. Simultaneously, 10BASE5 and 10BASE2 exhibit simpler wiring requirements since only a single wire is needed, contrasting with the more complex setup of 10BASE-T. [51]

4.4 100BASE-T

Fast Ethernet has been widely adopted in local area networks (LANs) to provide higher data rates for tasks such as file transfers, video streaming, and other bandwidth-intensive applications. In November 1992, a high-speed study group was established by the IEEE 802.3 to assess proposals for 100Mbps Ethernet. Serving as the foundational layers within the OSI reference model, 100BASE-T is designed to connect end systems to a medium, such as an unshielded twisted pair wire. To form a network, multiple media are interlinked through a repeater.
The Media Access Control (MAC) protocol dictates the rules governing how stations access the shared network. The Reconciliation sublayer acts as the standard equivalent of "glue logic," primarily tasked with translating the interface specifications of the existing MAC to the newly defined Media Independent Interface (MII).

The Physical Layer (PHY) plays a crucial role in translating signals from the Media Independent Interface (MII) to the Medium Dependent Interface (MDI) in network stations. A repeater, positioned atop two or more PHY connections, provides multiple ports for connecting end stations, enabling communication over shared media in a network architecture. The development of media interface technologies to achieve 100 megabits per second (Mbps) data rates posed a significant challenge for 100BASE-T. Two distinct approaches were taken, resulting in standards supporting three media types collectively known as 100BASE-X. In the first approach, utilizing 100BASE-X, fiber and unshielded twisted pair (UTP) PHYs were employed. Specifically, 100BASE-TX operates on high-grade Category 5 cable with a range of up to 100 meters, while 100BASE-FX offers a fiber interface compatible with 100BASE-T. Both interfaces support full-duplex signaling. In the second approach, known as 100BASE-T4, the focus is on data-grade Category 3 cable common in 10BASE-T networks. To achieve 100 Mbps, 100BASE-T4 uses four cable pairs, reducing signaling bandwidth. An 8B/6T block code further limits signaling requirements. The 100BASE-T4 PHY operates on a half-duplex signaling scheme.

The 100BASE-T standard utilizes a repeater function to connect devices within a shared network segment. This repeater enables all connections to collectively utilize the total bandwidth of 100 megabits per second (Mbps) in the local area network (LAN). The standard defines two types of repeaters: CLASS I and CLASS II. Networks with CLASS I devices are limited to a single repeater per segment, while those with CLASS II repeaters have a stricter delay budget, allowing for two devices on a segment and an additional 5 meters of unshielded twisted pair (UTP). Maintaining Ethernet's packet format and access protocol during the transition to higher speeds provides an advantage by simplifying the migration process and

contributing to the ease of upgrading network infrastructure.
It's important to note that while Fast Ethernet provided a significant speed boost compared to its predecessor, it has since been surpassed by Gigabit Ethernet (1000BASE-T) and higher-speed Ethernet standards as the demand for faster network connections has continued to grow.[55]

4.5 1000BASE-T

In the realm of computer networking, Gigabit Ethernet (GbE or 1 GigE) denotes the transmission of Ethernet frames at gigabit-per-second rates. The prevalent 1000BASE-T, as defined by the IEEE 802.3ab standard, was introduced in 1999 and brought about a substantial enhancement in local networks, surpassing the speed of Fast Ethernet. Notably, it made effective use of existing cables and equipment reminiscent of earlier standards, effectively replacing traditional Ethernet. The 1000BASE-T standard operates on copper wiring, recommending a maximum segment length of 100 meters and the use of Category 5 or superior cables. Auto-negotiation is a requisite for 1000BASE-T, involving one endpoint as the master and the other as the slave. Distinguishing itself from its predecessors, 1000BASE-T employs echo cancellation and adaptive equalization through a hybrid circuit. This enables all pairs to concurrently transmit in both directions using four lanes and 5-level pulse amplitude modulation (PAM-5). Negotiation takes place on two pairs, ensuring the successful establishment of a "gigabit" connection even with only two pairs connected between gigabit interfaces. Diagnostic registers in most gigabit physical devices address potential link establishment issues.
The transmission process involves sending data over four copper pairs, with eight bits transmitted at a time. Initially, the eight bits of data undergo expansion into four three-bit symbols through a complex scrambling procedure, utilizing a linear-feedback shift register. Subsequently, these three-bit symbols are mapped to continuously varying voltage levels during the transmission process. An illustrative example of this mapping is provided below: The optional feature of Automatic

Symbol	000	001	010	011	100	101	110	111
Line Signal level	0	+1	+2	-1	0	+1	-2	-1

Table 4.1: Example of mapping between symbols and signal levels in 1000BASE-T, source:[56]

MDI/MDI-X configuration simplifies cable connections, eliminating the need for crossover cables or manual selector switches. To optimize the use of existing Cat-5e and Cat-6 cabling, next-generation standards like 2.5GBASE-T and 5GBASE-T

operate at 2.5 Gbit/s and 5.0 Gbit/s, respectively. These standards, derived from 10GBASE-T but with lower signal frequencies, complement the capabilities of 1000BASE-T.[56]

4.6 100BASE-T1

The IEEE 802.3bw standard, also known as 100BASE-T1 (formerly BroadR-Reach), is a 100 Mbps automotive Ethernet standard designed to enhance data throughput, adhere to stringent automotive emissions standards, and reduce cabling weight and cost in automotive networking. Leveraging principles like superposition, along with specific encoding and scrambling schemes, 100BASE-T1 achieves notable reductions in electromagnetic interference (EMI), cabling weight, cost, and footprint size when compared to Ethernet standards such as 10BASE-T and 100BASE-TX. Tailored to meet automotive system requirements, 100BASE-T1 efficiently transmits and receives data using only an unshielded single twisted-pair cable. It achieves speeds of 100 Mbps over communication distances of at least 15 meters, facilitating the communication of various data types within vehicles, including audio, video, connected car features, firmware/software, and calibration data, particularly through the use of audio video bridging (AVB).
100BASE-T1 utilizes a unique encoding scheme, namely 4B3B, 3B2T, and PAM3, to minimize emissions compared to Fast Ethernet. The 100BASE-T1 PHY handles encoding, scrambling, and serialization tasks before transmitting data over a single unshielded twisted-pair cable. Importantly, this standard maintains transparency at the MAC level, keeping the existing Media Independent Interface (MII) unchanged.

There are four main xMIIs used for 100BASE-T1, each with specific characteristics:

1. **MII:** 4-bit-wide data interface, with receive and transmit controls and clocks.
2. **RMII:** 2-bit-wide data interface, with receive and transmit controls and a single clock reference.
3. **RGMII:** 4-bit-wide data interface, with receive and transmit controls and clocks.
4. **SGMII:** 2-pin LVDS for both receive and transmit paths.

After receiving data from the MAC, the Ethernet PHY performs encoding, scrambling, and serialization, preparing the data for transmission over the unshielded single twisted-pair cable to a linked partner.
When a 100BASE-T1 PHY communicates with a MAC via RGMII, it receives four parallel bits clocked at 25 MHz, resulting in a total of 100 Mbps. The PHY

then converts these four bits to three bits while increasing the clock frequency to 33 1/3 MHz, ensuring a constant 100 Mbps bit rate. Using each set of three bits, the PHY generates a ternary pair (2T). Finally, the ternary pair vector (TA, TB) is transmitted through three-level pulse amplitude modulation (PAM3) at a fundamental frequency of 66 2/3 MHz. This signal employs three voltage levels (+1 V, 0 V, and -1 V) with a peak-to-peak amplitude of less than 2.2 V. 100BASE-T1, as

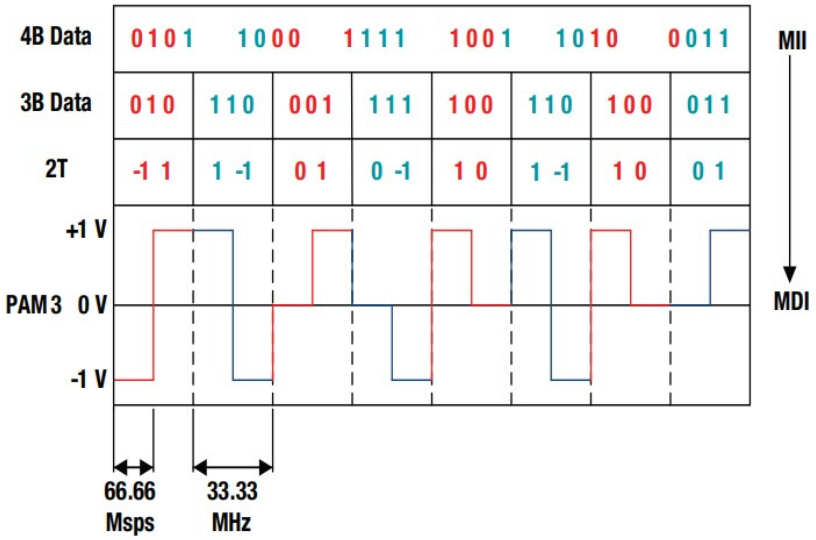

Figure 4.4: GPHY data conversion from MII to MDI, source:[57]

a physical full-duplex interface, allows simultaneous transmission and reception on the same pair, distinguishing it from 10BASE-T and 100BASE-TX where dedicated pairs handle transmission and reception. This shared media approach reduces overall cable weight in vehicles, offering not only material cost savings but also contributing to improved fuel efficiency. The achievement of physical full duplex is rooted in the principles of superposition.
In 100BASE-T1 PHYs, integrated hybrids and echo cancellation techniques are employed to eliminate the transmitted signal's impact and extract received information from a link partner. To facilitate this process, one PHY operates as a master, and the other functions as a slave. [57]

4.7 10BASE-T1S

As the paradigm of zonal-based architectures advanced, it became apparent that extending Ethernet connectivity to encompass edge sensors and actuators was necessary to fully exploit the benefits of this innovative framework. Traditional connectivity technologies, such as FlexRay and CAN, necessitated protocol translation typically executed through gateways, thereby introducing cost, complexity, and latency into the system. Even existing automotive Ethernet technologies, exemplified by 100BASE-T1, proved inadequate in terms of system costs for supporting the migration of edge connectivity applications to Ethernet. This limitation arose from the technology's reliance on point-to-point switched connectivity.
In response to this challenge, the IEEE issued a call for interest, seeking a resolution to these issues. Several key requirements emerged, including the imperative for faster communication compared to existing technologies like CAN(FD), the need to supplant legacy in-vehicle networking technologies like FlexRay, an alternative to 100BASE-T1 in instances where cost and energy efficiency were paramount, and the capacity to support connectivity for both simple and redundant sensor networks. This concerted effort aimed to address the evolving demands of automotive architectures and foster a more seamless integration of Ethernet into the broader automotive ecosystem. [58]

10BASE-T1S, denoting 10 Mbps Single Pair Ethernet, represents a contemporary technological advancement sanctioned by IEEE under the 802.3cg standard. The 'S' in 10BASE-T1S designates its suitability for short-reach applications. This innovative iteration embraces a multidrop topology, where each node establishes a connection with a singular cable. This strategic approach obviates the necessity for switches, leading to a reduction in cable infrastructure. Diverging from the conventional Ethernet cabling configuration with four pairs of wires, 10BASE-T1S cables streamline the process by employing a single pair of twisted wires. The IEEE guidelines specify a maximum of eight nodes per cable, with scalability allowing for additional connections. The primary objective of 10BASE-T1S revolves around deterministic transmission within a collision-free multidrop network. Widely integrated into applications utilizing Controller Area Network (CAN) Flexible Data Rate or 100BASE-T1, it adheres to the Physical Layer Collision Avoidance protocol, ensuring the optimal utilization of the entire 10 Mbps bandwidth. Noteworthy is its support for an arbitration scheme, guaranteeing seamless node access to the media within predefined time parameters. This refined architecture underscores the commitment of 10BASE-T1S to reliable and collision-free communication, making it an instrumental technology in contemporary networking landscapes. [54]

4.7.1 The 10BASE-T1S frame

10BASE-T1S has implemented the 802.3 Ethernet frame, as outlined below:

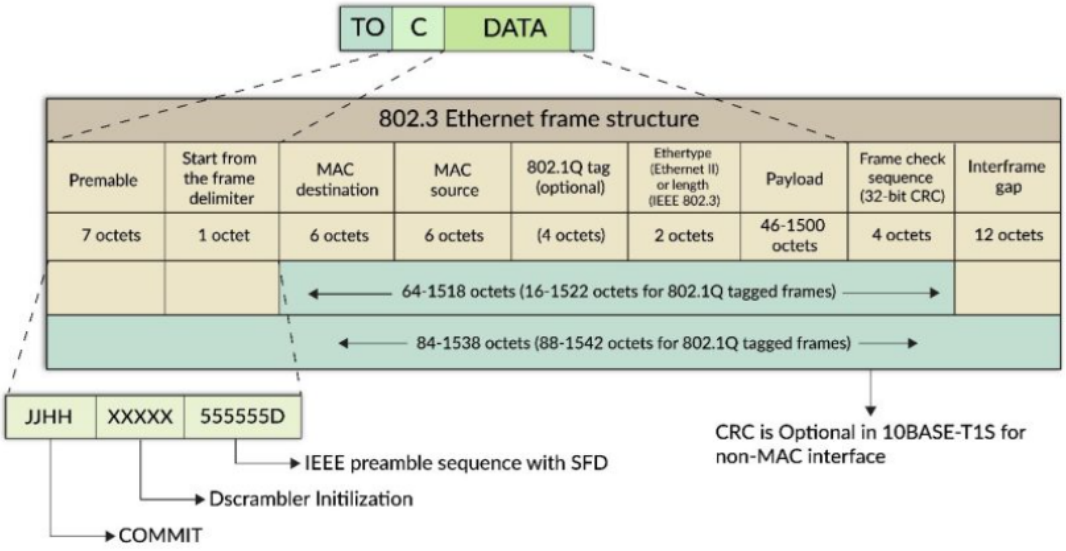

Figure 4.5: 10Base-T1S frame, source:[59]

- **Preamble:** composed of 7 octets
  - The commit comprises 5B symbols, 20 bits.
  - Descrambler Initializer is transmitted to enable the receiver node on the 10BASE-T1S bus to decode the data frame.
- **SFD**: composed of 1 octet 0xD5.
- **Data:** might be composed of 60-1514 octets.
- **CRC-Frame check sequence:** composed of 4 octets.
- **Inter-Packet Gap:**composed of a minimum of 12 octets and is inserted between frames.[60]

4.7.2 Advantages

Earlier automotive technologies, such as 100/1000BASE-T1, were characterized by high performance, albeit at a considerable cost. While other automotive specifications use Pulse Amplitude Modification (PAM) signaling, 10BASE-T1S

uses Differential Manchester Encoding (DME) to save complexity and cost. The Single Pair Ethernet (SPE) protocols, as outlined in IEEE 802.cg, offer improved bandwidth capabilities. This enhanced bandwidth can be leveraged to achieve reduced latency communication on the bus line. As a result, In-Vehicle Network (IVN) applications can operate with higher-quality data when compared to legacy protocols like MOST, CAN, and LIN. [61]

4.7.3 Working principles

Signaling and Encoding

The 10Base-T1S employs the Differential Manchester Encoding (DME) technique. An enhancement to Manchester coding, which falls under binary phase-shift keying, eliminates the need to determine the initial polarity of the transmitted message signal. This is because information isn't encoded based on absolute voltage levels, but rather on their transitions. Within the system, two clock ticks occur per bit period. During each second clock tick, a potential level transition takes place based on the data. Conversely, during the remaining ticks, the line state shifts unconditionally to facilitate clock recovery. Two versions of the code exist: one initiates a transition for a binary 0 and no transition for a binary 1, while the other triggers a transition for a binary 1 and no transition for a binary 0. Differential Manchester encoding offers several advantages:

1. Robust Clock Recovery: It ensures at least one transition per bit, aiding in reliable clock recovery, even in challenging conditions.

2. Improved Noise Immunity: Detecting transitions is less susceptible to errors in noisy environments compared to comparing signal levels against a threshold.

3. Polarity Insensitivity: Unlike Manchester encoding, the polarity of the signal doesn't affect the decoding process. This means the scheme functions the same even if the signal is inverted (e.g., wires swapped). Other encoding schemes with this property include NRZI, bipolar encoding, coded mark inversion, and MLT-3 encoding.

4. Zero DC Bias: When the high and low signal levels have equal magnitude but opposite polarity, the average voltage around each unconditional transition becomes zero. This absence of DC bias reduces necessary transmitting power, minimizes electromagnetic noise generated by the transmission line, and simplifies the use of isolating transformers.

These benefits come at the cost of doubling the clock frequency of the encoded data stream. [62]

Multidrop Topology

The 10BASE-T1S exhibits versatility by supporting both half-duplex and full-duplex communication. This feature enables the establishment of either a direct point-to-point connection between two nodes or the utilization of a multidrop topology, where up to eight nodes can be seamlessly interconnected along a single 25-meter bus segment. The implementation of multidrop cabling on a single bus line offers practical advantages, allowing for an extension and scalability with a reduced reliance on physical wires and diminished overall weight compared to point-to-point topologies. The minimal connector space required at the Engine Control Unit (ECU) facilitates effortless expansion by the addition of sensor units. An illustrative example of this scalability is evident in a bus line accommodating extra sensor units, such as those for ultrasonic and short-range radar applications. The multidrop cabling approach thus exemplifies a streamlined and efficient method for extending the network's capabilities. [63] 10BASE-T1S incorporates a mixing

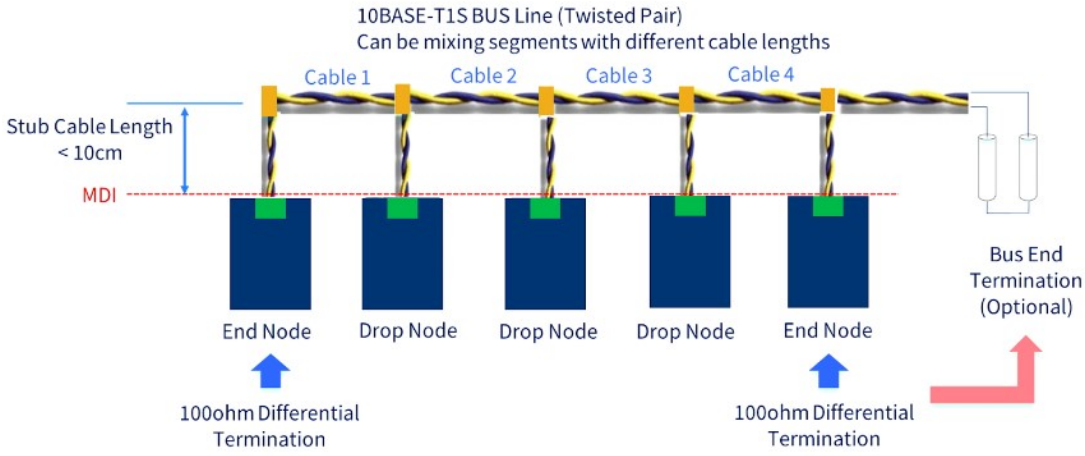

Figure 4.6: 10Base-T1S Multidrop topology, source:[61]

segment, enabling the connection of multiple devices to a common channel. The differential resistance, set at 100Ω for end node devices and 50Ω for drop nodes, facilitates precise timing for communication over the shared channel. An alternative configuration involves utilizing a 100Ω terminal for the end node, and a single 100Ω impedance termination, as a substitute for traditional cabling, can achieve a similar outcome. [61]

Chapter 5

Ethernet Arbitration Mechanisms

5.1 CSMA/CD Protocol

The original Ethernet standard outlined the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access method for data communications (IEEE 802.3, 2009). It is essential to emphasize that the development and evolution of the CSMA/CD protocol in Ethernet were primarily influenced by the existence of a shared-access communications medium. In this shared-medium scenario, all devices can attempt to access the communication medium when they have data to send, but only one device can transmit at any given time. This results in a competitive environment where devices contend for bandwidth on a shared communications medium.
Additionally, CSMA/CD enforces half-duplex communications, allowing only one device to transmit data on the medium at a specific moment. In a moderate-sized CSMA/CD Ethernet network, collision recovery can use a significant portion of the available bandwidth, ranging from 40% to 50%.[64]

Ethernet networks are characterized by a bus topology based on coaxial cable and 10Mb/s data rate transmission. CSMA/CD means Carrier Sense Multiple Access with Collision Detection and it's the method adopted to arrange the arbitration of the transmission channel. [17]. The CSMA/CD protocol is structured into three phases:

- **Carrier Sense**: each node transmits data only after checking whether the bus is free
- **Multiple Access**: despite the contribution of the carrier sense, occasionally

happens that multiple nodes start transmitting at the same time, this is due to the propagation time required by the signal to travel throughout the network, hence one node can decide to transmit before receiving the information that the bus is no longer idle.

- **Collision Detection**: whenever two or more nodes transmit simultaneously there is a Collision, each node, while transmitting, listens to the signals coming from the bus and checks if there is correspondence (listen while talking).

The measure taken in the event of a collision is the following:

1. The transmitting node sends a 32-bit jamming sequence that informs the remaining nodes about the collision that took place.
2. The nodes in listening mode receive the jamming sequence and discard the bits that previously received
3. The transmitting node, after some time, takes another run up to 16 times. The rescheduling happens in a pseudo-random range of time in order to avoid another collision.

5.1.1 Ethernet Constraints on CSMA/CD

The characteristics and topology of a network's physical medium play a crucial role in determining parameters like slot time, inter-frame spacing, and physical segment lengths. In LAN design, collision detection and the subsequent generation of a jam signal impose constraints. Stations must identify a corrupted signal and broadcast a jam signal before completing their transmission. The smallest acceptable Ethernet frame is 64 octets (64 bytes), ensuring reliable collision detection and jamming. Collision detection (CD) and the subsequent 32-bit jam signal generation limit transmission before generating a minimally sized Ethernet frame, resulting in a runt frame. Runt frames, failing Ethernet's minimum length and Frame Check Sequence (FCS) tests, are discarded by devices with minimal processing overhead. A station's random back-off timing is measured in intervals of the slot frame length, which is determined by the network's design and physical characteristics. This back-off mechanism is crucial for avoiding collisions and ensuring efficient data transmission. In the event of a collision, it's essential for a device to detect it before completing the transmission of a minimal-sized frame. This detection allows the device to promptly send a jam signal, notifying other stations about the collision and prompting them to perform their own random back-off and retransmission procedures. The slot time is then defined as twice the duration it takes for a signal to travel between the network's two most distant stations, creating an additional delay. This extended period caters to inter-frame spacing, facilitating the dissipation of signals following successful transmission.

Ethernet and CSMA/CD impose both physical and logical constraints on network segments. In terms of the physical medium, the signal degrades over distance due to line noise and signal attenuation. To overcome this limitation, layer one repeaters or hubs can be employed, allowing more devices to access shared media but introducing higher network latency.
Extending an Ethernet LAN with layer one devices may result in reduced throughput, as additional devices increase contention for shared media, potentially requiring longer slot times due to greater geographical separation. Linear bus topologies present a vulnerability with a single point of failure, and although layer one hubs can create a physical tree topology, they don't logically segment the collision domain. Logical collision domains face scalability challenges with more devices, frequent transmissions, increased loads, or extended cable distances, leading to a higher collision rate.
When implementing a tree topology, network designers should adhere to the industry-standard "5-4-3" rule. This guideline dictates that no two nodes on a network should be separated by more than five network segments, four repeaters, or three backbone segments. Following this rule ensures that the network's layout maintains propagation latency within acceptable levels.[64]

5.2 PLCA

CSMA/CD demonstrates unpredictable and unrestricted latencies, mainly due to potential collisions, making it unsuitable for numerous applications in Industrial, Automotive, and Automation Control where determinism is crucial. In contrast, PLCA ensures a maximum guaranteed latency, enhances throughput and ensures fair access, effectively addressing these constraints.

PLCA, situated as a reconciliation sublayer in 802.3 clause 148, aims to bring deterministic performance to CSMA/CD within half-duplex, mixing-segment (multidrop) networks featuring a constrained number of nodes. Even though the media access is still managed by the established CSMA/CD functions outlined in Clause 4, PLCA successfully prevents physical collisions on the media.

The fundamental operation of PLCA involves dynamically generating transmit opportunities, ensuring that only one node can send a packet over the media at any given time.

1. Each node is assigned a unique ID, with the node possessing ID = 0 serving as the PLCA coordinator.

2. This coordinator initiates a cycle by transmitting a BEACON signal, followed by its own designated transmit opportunity.

3. In cases where no data is available, the transmit opportunity is relinquished after 20 bit times, allowing the next node in line to have its turn. Conversely, if data is ready, the transmit opportunity persists until a packet has been successfully transmitted.

4. The coordinator commences a new cycle when the last node, regardless of whether it yielded its opportunity or used it to transmit data, has been granted a transmit opportunity.

It is crucial to emphasize that PLCA ensures the avoidance of physical collisions on the media, ensuring uninterrupted reception. The reconciliation sublayer autonomously identifies if a node has data to transmit. In the case where the RS detects this, it delays transmission until a suitable opportunity arises, either through a short delay mechanism or by recognizing a "logical" (local) collision (which in fact don't cause data corruption). Logical collisions impact only the local node before any physical transmission occurs. Following a logical collision, the MAC refrains from initiating a new transmission attempt while the line is reported as busy (Carrier Sense asserted). After reporting a logical collision, PLCA awaits a new transmit opportunity before reasserting Carrier Sense. In the initial transmit

attempt, the back-off time can be either 0 or 512 bit times. Since logical collisions can only occur at the transmission's outset, and 512 bit times are shorter than the minimum packet size being received, the MAC is always ready for a new transmit attempt before the next transmit opportunity arises. Moreover, the subsequent transmit attempt is guaranteed to succeed, preventing the back-off time from exceeding 512 bit times.

The duration of the BEACON cycle depends on the total data length sent by all nodes and is entirely asynchronous concerning MAC processes. This implies that any node, regardless of its assigned ID, could obtain a transmit opportunity at any time.

PLCA ensures packet fairness in a round-robin manner, granting each PHY exactly one transmit opportunity at each BEACON, irrespective of packet sizes. This aligns with CSMA/CD's fairness policy. Although the current PLCA definition doesn't address data fairness, ongoing discussions and proposals aim to incorporate this feature if necessary. [65]

5.2.1 Use of PLCA within Multidrop Topology in 10BASE-T1S

A PLCA transmit cycle sets the opportunities for communication and the sequence in which nodes are heard. PLCA operates by assigning each node, or PHY, a unique PHY ID [0...N], allowing only the designated PHY with the transmit opportunity to send data. Transmit opportunities are distributed through a round-robin algorithm, starting with the Master node at PHY ID = 0. Nodes can initiate transmissions only during their assigned transmit opportunity based on their individual node ID. The Master node initiates a new cycle by transmitting the BEACON synchronization pattern, signaling the beginning of the PLCA cycle.

The PLCA cycle is comprised of the BEACON followed by N+1 time slots, enabling the transmission of N+1 variable-sized DATA packets. During their designated transmit opportunity, a PHY has the option to promptly transmit a packet or is required to transmit a COMMIT pattern consisting of SYNC symbols. This compensates for any MAC latency and allows for additional time before the actual packet transmission. Nodes have the flexibility to extend the time slot to accommodate larger transmissions and can prioritize high-priority messages for burst transmission. While the node with the current transmit opportunity is transmitting, other nodes patiently wait for its completion before the cycle progresses to the next node with a transmit opportunity. A new time slot initiates either when nothing is transmitted within a specified time (TO_TIMER) or at the conclusion of any

packet transmission. [63]

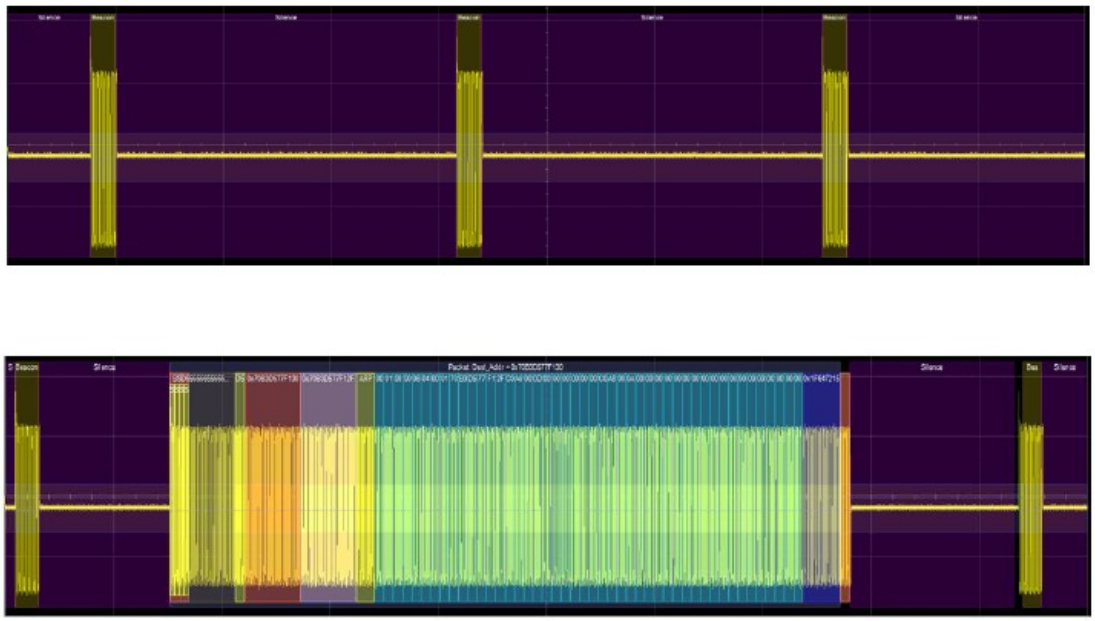

Figure 5.1: Data Transmission if there is no data traffic (top) and when data from a node will expand the time between two BEACONs (bottom), source: [63]

PLCA's operational principle involves synchronizing TO_TIMERs to maintain a maximum latency consistently below one PLCA cycle. The TO_TIMER, typically 20 bits, results in minimal throughput loss when waiting for idle PHYs. At the start of each transmission cycle, Node 1 is initially granted the transmit opportunity. If Node 1 lacks DATA and cannot COMMIT, it relinquishes the opportunity to the next node on the bus. This system's advantage lies in nodes independently tracking TO_TIMER post-BEACON. Nodes without data yield their opportunity, ensuring minimal throughput loss and latency increase within the short TO_TIMER window. Unlike TDMA, PLCA isn't a fixed reference; it dynamically adjusts based on each node's transmit needs.

The illustration below depicts a scenario of PLCA cycles with minimum latency. In cycle 1, as no node has data to transmit, the total latency is solely determined by the number of nodes multiplied by the TO_TIMER. In cycle 2, Nodes 1 and 3 have transmissions planned, leading other nodes to relinquish their transmit opportunities. Node 1 promptly transmits DATA, while Node 3 initiates with a COMMIT followed by DATA.

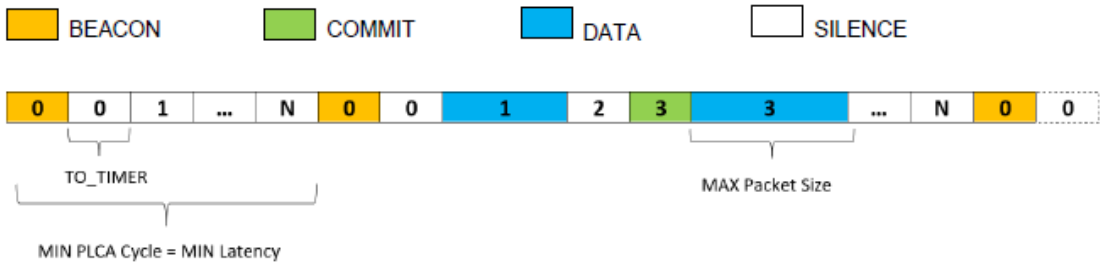

Figure 5.2: Minimum latency PLCA, source: [63]

In the scenario of a Maximum latency PLCA cycle, each node possesses a packet of maximum size and transmits a COMMIT while awaiting the MAC. [63]

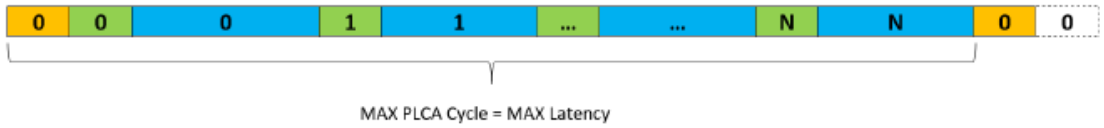

Figure 5.3: Maximum latency PLCA, source: [63]

Chapter 6

Testing Environment

The objective of this work is to implement and simulate an Ethernet network based on 10BASE-T1S. Additionally, further considerations will be undertaken. The working environment is a network composed of a few tools: RAD-Meteor, RAD-Comet, and the software Vehicle Spy. All these tools are provided by Intrepid Control Systems.

6.1 Vehicle Spy 3

Vehicle Spy is an all-encompassing tool that serves various purposes, including diagnostics, node and ECU simulation, data acquisition, automated testing, and monitoring of vehicle network buses. It has been specifically crafted with an emphasis on user-friendly operation and maximizing user productivity.
Vehicle Spy can be employed as a bus analyzer for monitoring network and message transmission, utilized as a "data logger" or "flight recorder", furthermore it can automatically capture bus events and store trace files, conduct simultaneous operations on multiple networks, each potentially employing a different protocol, simulate nodes, gateways, or even entire vehicles. The main features of Vehicle Spy are the following:

- Bus Monitor: used to view messages as they appear on the network, in bus traffic monitor applications, message buffers are typically collected and then saved.

- Signal Monitoring and Recording: collect, whenever are already installed, sensor data, along with vehicle network data, such as RPM, Coolant Temperature, Engine Load, Throttle Position, MAP, etc. without performing any sensor wiring on a vehicle.

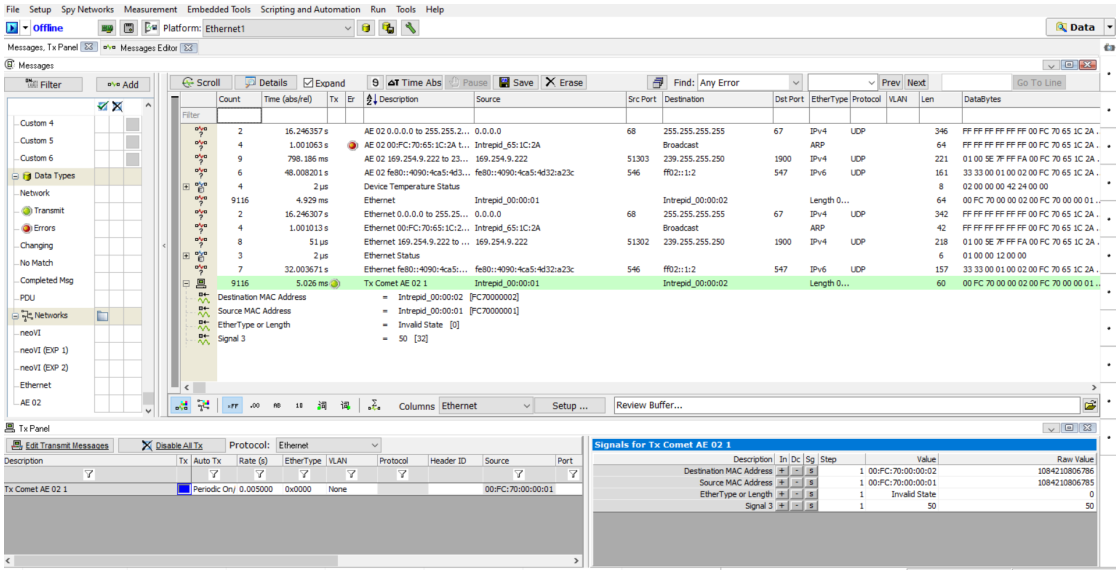

Figure 6.1: Vehicle Spy 3 interface

- Flight Recording: continuous monitoring of the network in a vehicle or on a vehicle bench for the purpose of observing or capturing issues over a period of time.
- Node/Vehicle Simulation: simulating an electronic module or car on a network. Vehicle Spy's playback feature simulates vehicle traffic by collecting and storing data, enabling the repetition of captured problems and facilitating tests.
- Test Automation: involves being able to send messages, receive messages, capture buffers, and react to incoming data.[66]

6.2 RAD-Comet

The RAD-Comet 2 serves as a versatile tool for developing and testing 10BASE-T1S communication in automotive systems. It can function as a simulated node on a 10BASE-T1S network, allowing simultaneous capture and analysis of 10BASE-T1S traffic alongside other vehicle networks. When utilized as a media converter, the RAD Comet 2 can bridge across its three Ethernet physical layers. This proves beneficial for tasks such as connecting a PC to an Automotive Ethernet device for monitoring network activity using Wireshark or Vehicle Spy. Additionally, it facilitates the integration of a 10BASE-T1S device or network into a 100/1000BASE-T1 network.

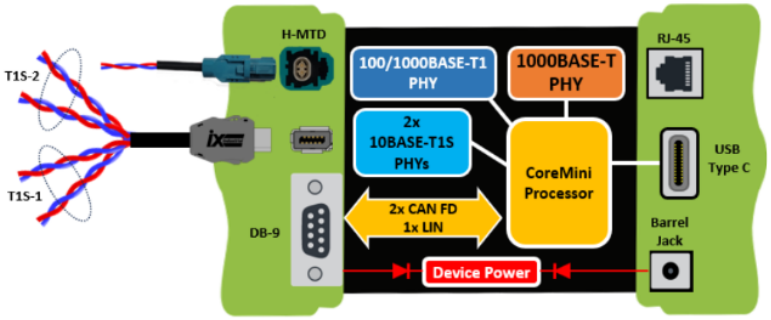

Figure 6.2: RAD-Comet Interfaces, source:[67]

Network Interfaces:

- 1x 10BASE-T1S (LAN8671)
- 1x 100/1000BASE-T1 (88Q2221M)
- 1x 10/100/1000BASE-T
- 2x ISO CAN FD channels with selectable on-board termination

Frames can arrive on one Ethernet network and be promptly forwarded to another network, either unaltered or with adjustments to the header or payload. This feature creates possibilities for incorporating diverse functions that prove advantageous in vehicle testing. Examples include altering the frame destination, MAC spoofing, layer 3/4 address translation, payload scaling, and payload manipulation (for activities like boundary testing, fault injection, penetration testing, etc...).[67]

6.3 RAD-Meteor

Intrepid's RAD-Meteor provides an economical solution for incorporating a 10BASE-T1S (IEEE Std 802.3cg) port into your computer via USB. Apart from monitoring 10BASE-T1S traffic like a standard Ethernet port, the RAD-Meteor network adapter is equipped to support the requisite test modes for testing the 10BASE-T1S network. Additionally, the RAD-Meteor features a built-in full-color display for observing network activity and monitoring the status registers of the 10BASE-T1S. Its main

Figure 6.3: RAD-Meteor, source:[68]

features are: a USB Network Adapter equipped with 10BASE-T1S Physical Layer (PHY), enumerates as a standard Ethernet Network Interface Controller (NIC) in Linux or Windows environments, supports transmission and reception with socket applications, utilizes the Microchip LAN867X 10BASE-T1S PHY, has a terminal Block with optional termination for T1S connections, field updatable firmware, device configuration and status register reporting over USB through Ethernet frames.[68]

6.4 Setup

The initial phase of this experimental procedure involved setting up a 10BASE-T1S network for testing purposes. The simulation environment comprised a computer equipped with Vehicle Spy software, along with a RAD-Comet and two RAD-Meteors. The network connections were established according to the configuration illustrated below.

Figure 6.4: Complete Setup

The computer served as both an endpoint for communication and a monitoring station. The RAD-Comet, functioning as a media converter tailored for automotive applications, facilitated connections among multiple devices within the network. It was linked to the PC and powered via a USB cable. Communication between the PC and hardware occurs through an USB cable.
The two RAD-Meteors likely operated as Ethernet nodes or devices communicating over the 10BASE-T1S network. Both were powered through a USB Type-C cable connected to the PC. The pivotal connection point lay in the unshielded twisted pair (UTP) cabling linking them to the RAD-Comet. This cable, depicted in the image, featured three endpoints, one corresponding to the 10BASE-T1S port on the Comet.
In a 10BASE-T1S network, termination was essential at the two end nodes to prevent signal reflections and ensure signal integrity. The end panel of the RAD-Meteor device featured silkscreen markings indicating the polarity of the bus wires, with "P", denoting positive and "N" indicating negative. This aided in accurately connecting the network wires to the RAD-Meteor.
Each end of the cable destined for the RAD-Meteors featured a removable screw terminal block for easy attachment of network wires to the RAD-Meteor.

Vehicle Spy offers a versatile platform for conducting simulations across various network types, including CAN, LIN, FlexRay, and more. One standout feature

is its compatibility with different interfaces, notably CAN and Ethernet. The RAD-Comet module offers access to two distinct networks: CAN and Ethernet. Delving into its specifics, the RAD-Comet module boasts a range of channels. For CAN, it provides DW CAN 01 and DW CAN 02 channels, while for Ethernet, it offers three channel options: AE01 (compatible with 100BASE-T1), AE02 (compatible with 10BASE-T1S), and ETH01 (compatible with 10/100/1000BASE-T PHY). Among these, the AE02 channel is particularly pertinent to our study. Moreover, the Vehicle Spy interface enhances usability by allowing users to choose between PLCA or CSMA/CD protocols when working with the AE02 channel. This flexibility enables tailored simulations to suit specific research or testing requirements.

The PC serves as the central point of the network, enabling the definition of all simulation parameters. Upon launching the software, from the initial interface users can create a file with a .vs3 extension, wherein they specify all simulation conditions. As soon as the new setup file is generated, users can establish the volume of traffic to be directed through the bus. Specifically, it is possible to define both Transmit (Tx) and Receive (Rx) messages. For both types of messages defined, users can:

- Specify the network through which the message is transmitted (e.g., AE 02).
- Provide a brief description for easy identification.
- Indicate both the source and destination MAC addresses.
- Define the EtherType (e.g., IPV4, ARP, PTP).
- If the message is periodic, set the default period within a range of 0.005 seconds to 5 seconds.
- Define the length of the Ethernet Payload, including the option to select the type of signal to transmit (analog, digital, state-encoded, text).

The two RAD-Meteors serve as versatile components within the system, capable of operating in multiple modes. They can function as passive receivers, adeptly capturing data frames from the network. Alternatively, they can actively transmit messages at different rates. Beyond their role in data transmission and reception, these RAD-Meteors play a pivotal role in enhancing network integrity. Furthermore, the RAD-Meteors play a crucial role in maintaining a comprehensive record of both received and transmitted messages, providing valuable insights into the communication dynamics of the network.

6.5 The .vsb and .pcap files

After each simulation, it's crucial to gather all transmitted data across the bus into a file for comprehensive data analysis. Vehicle Spy offers two file extensions for capturing this data: .vsb and .pcap.
The .vsb file, specific to Vehicle Spy, encompasses:

- Information on transmitted messages, including message IDs, signal names, and their corresponding values.
- Settings pertaining to hardware interfaces like CAN, LIN, FlexRay, and Ethernet used for connecting to the vehicle's network.
- Configuration parameters for vehicle communication protocols such as OBD-II, UDS.

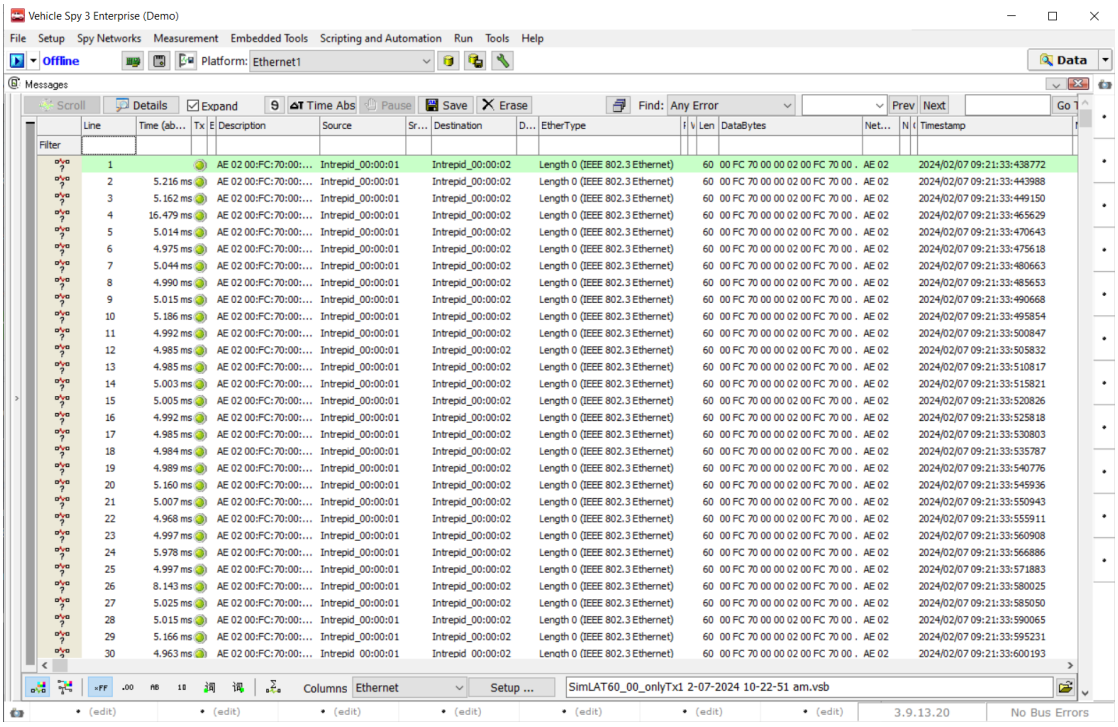

Figure 6.5: .vsb file open on Vehicle Spy

The .vsb file provides a convenient overview of the transmitted data, facilitating quick inspection. However, for in-depth analysis, the .pcap file proves more versatile. It can be read by both Matlab and Wireshark, allowing for comprehensive data examination and manipulation. This flexibility makes the .pcap file particularly

useful when detailed analysis of transmitted data is required.

The .pcap file extension, short for Packet Capture, represents a widely used file format employed in storing network packet data collected during network traffic analysis. It serves as a standard means to preserve network traffic for subsequent analysis, troubleshooting, or forensic examination. Various packet capture tools like Wireshark, tcpdump, or tshark are capable of generating .pcap files. These tools intercept the DL frames as they traverse through a network interface and store them into a .pcap file. Each packet within the file typically contains essential information such as source and destination IP addresses, port numbers, protocol type, packet size, and the actual payload data. Data traffic was monitored using Wireshark, installed on the PC, which serves as the central node for data collection. The network point under observation was typically one of the two RAD-meteors, specifically the passive one when the other was in transmitting mode. This capability was facilitated by Wireshark, which enables the observation of data flux within the Ethernet network associated with the specific RAD-Meteor being monitored.
Wireshark, in particular, stands out as an invaluable tool in this domain, offering comprehensive capabilities for data evaluation. In contrast to Vehicle Spy, Wireshark allows for the direct assessment of captured data. Displaying a wealth of information pertaining to individual data frames, Wireshark enables the creation of flow diagrams and I/O graphs based on the captured data. Additionally, Wireshark provides immediate insight into delta time, facilitating an understanding of latency trends.
The ability of Wireshark to offer detailed packet analysis, coupled with its visualization features and latency trend monitoring, makes it an indispensable resource for network administrators, security analysts, and researchers alike. Its versatility and functionality empower users to delve deep into network traffic data, unraveling insights crucial for optimizing network performance and addressing security concerns.

6.6 Matlab script

To facilitate the evaluation of received data, it has been chosen to incorporate a MATLAB script. The complete script is detailed in Appendix A. The primary goal of this script is to extract raw data from the .pcap file, isolating the timestamps from other information present in each element of the file.
Here's a summary of the steps outlined:

1. The first step involves uploading and reading the .pcap file on MATLAB by using a pcapReader object, named pcapReaderObj.

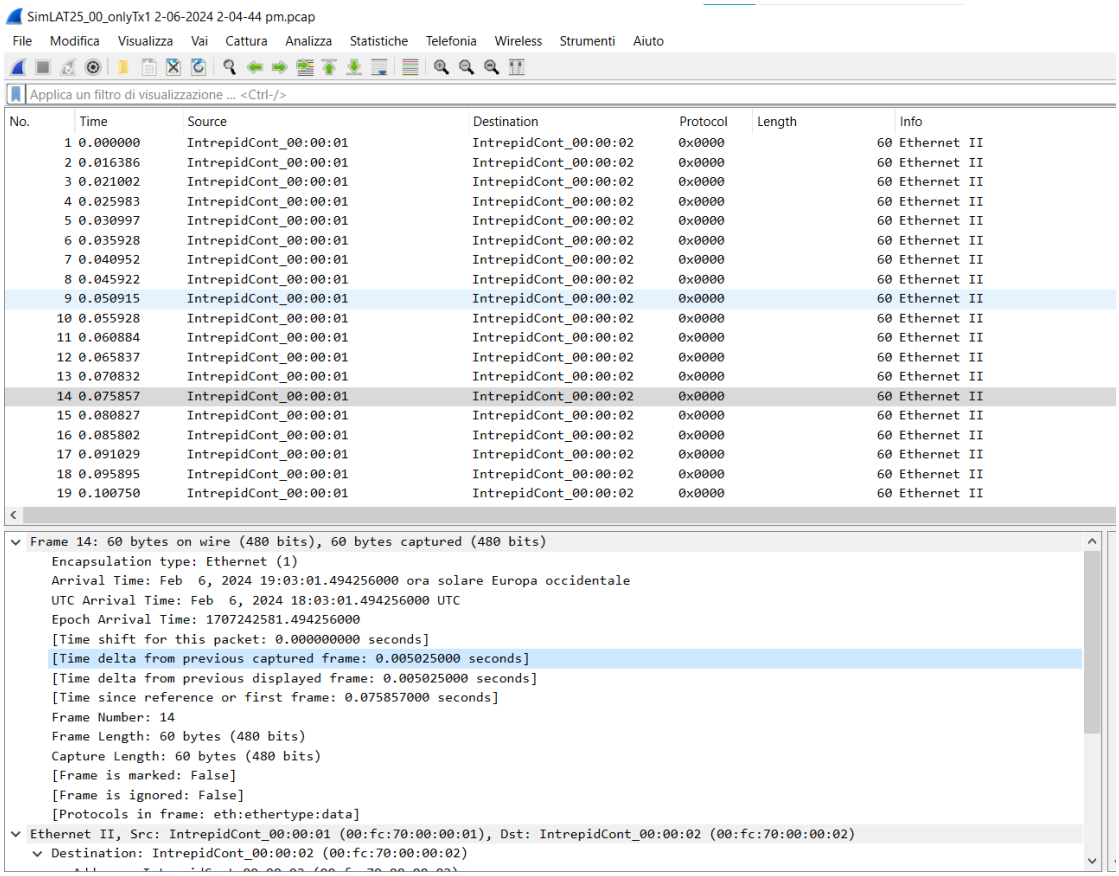

Figure 6.6: .pcap file open on Wireshark

2. Subsequently, the readAll function is employed to read all packets from the pcap file and gather them into the `decodedPackets` struct.

3. The line
`pcapReader('SimLAT60_00a.pcap', OutputTimestampFormat= 'datetime')`
is utilized to update `pcapReaderObj`, enabling the use of datetime format for timestamps during packet reading.

4. Additionally, a filter string is applied to select packets with a specific source address, in this case, '00FC70000001' corresponding to the PC.

5. Within the for loop, the script, while applying the filter, retrieves the timestamp of each Ethernet packet and stores it in the `time_vector` array. The structure of a single Ethernet packet is depicted in the figure 6.7.

6. The script proceeds to evaluate latencies. It iterates through each pair of consecutive packets in the `time_vector` array, calculating the latency between

```
ethPacket =

  struct with fields:

            SNo: 70
      Timestamp: 07-Feb-2024 15:31:54.346705
       LinkType: 1
       Protocol: 'eth'
   PacketLength: 60
         Packet: [1×1 struct]
       RawBytes: [1×0 double]
   TimestampSec: 1.707319914346706e+09
```

Figure 6.7: Data contained inside the EthPacket

them using the between function. The latencies are initially stored as strings in the array `latencies_str`. However, these strings contain the data in the format of hours (h), minutes (m), and seconds (s). In order to be sure that the data evaluated is exactly from a certain source it can also be applied a filter during this operation. In addition to this, all the frames contained in the data were preventively filtered on Wireshark on Vehicle Spy depending on the data point in which the data was supposed to be collected.

7. An array, `latencies_numeric`, is initialized to store the numeric values corresponding to the latency in seconds.

8. It iterates through each element in the `latencies_str` array, extracting the numeric value (in seconds) from the string and converting it to a numeric value. This is achieved by splitting the string at spaces and extracting the last element, which should contain the numeric value with the 's' character indicating seconds. The 's' character is then removed, and the resulting string is converted to a numeric value using str2double.

9. If the extracted string is empty, indicating an error or missing value, it assigns NaN to the corresponding element in `latencies_numeric`.

10. The last lines of the code involve evaluating the mean value of the latency. Additionally, the jitter and its respective mean are calculated. Jitter refers to the variability in latency measurements over time, often indicating the consistency or stability of the network performance. Calculating these metrics provides insights into the overall performance and reliability of the network communication.

Chapter 7

Experimental Results

This chapter provides insights into the experiments and simulations conducted utilizing the setup described in the preceding chapter. The primary objective of these experiments within the company environment was to assess the efficacy of the instruments employed in implementing the network. The bus network underwent various conditions during the experiments to observe and analyze its behavior.
A common procedure adopted in most experiments involved progressively increasing the number of data frames transmitted through the network to evaluate its performance. The network's overall performance was assessed based on the following criteria:

- Latency;
- Jitter;
- Packets transmitted;

While latency and jitter are related, they are not always directly correlated. The use of jitter buffers to mitigate jitter can contribute to overall latency by introducing additional playout delay. These metrics collectively provide valuable insights into the performance and reliability of the network communication. By analyzing latency and jitter under different experimental conditions, it becomes possible to optimize network configurations and identify potential areas for improvement.

Moreover, additional considerations were made regarding the bus's reaction when one or both of the two RAD-Meteors were transmitting messages. Further experiments were conducted to analyze the network's behavior under different network access methods, namely PLCA (Priority-based Logical Collision Avoidance) and CSMA/CD (Carrier Sense Multiple Access with Collision Detection).

7.1 Examining the performance of RAD-Meteor

7.1.1 Latency

The first object of observation is the RAD-Meteor, which, as said before is a peripheral device. Specifically, each RAD-Meteor can send messages with different dimensions and rate. In this thesis work, for each frame, those values were set at 1500 bytes with a rate of 1 millisecond. The goal is to monitor how latency and jitter are affected by the simultaneous increase in data transmission requests and to make observations regarding the network's reliability under these conditions. The simulations are carried out using RAD-Meteors and Vehicle Spy. The initial experiment involved a single transmitting RAD-Meteor to observe how the increasing concurrency of messages sent from the PC affects network performance. While the data rate from the RAD-Meteor remains constant, Vehicle Spy can send messages each one might have a minimum rate of 5 ms, allowing for an increase in the quantity of messages from the PC. It is possible to set quite high values for the length of the latter, in this case it is chosen 64 bytes of length.

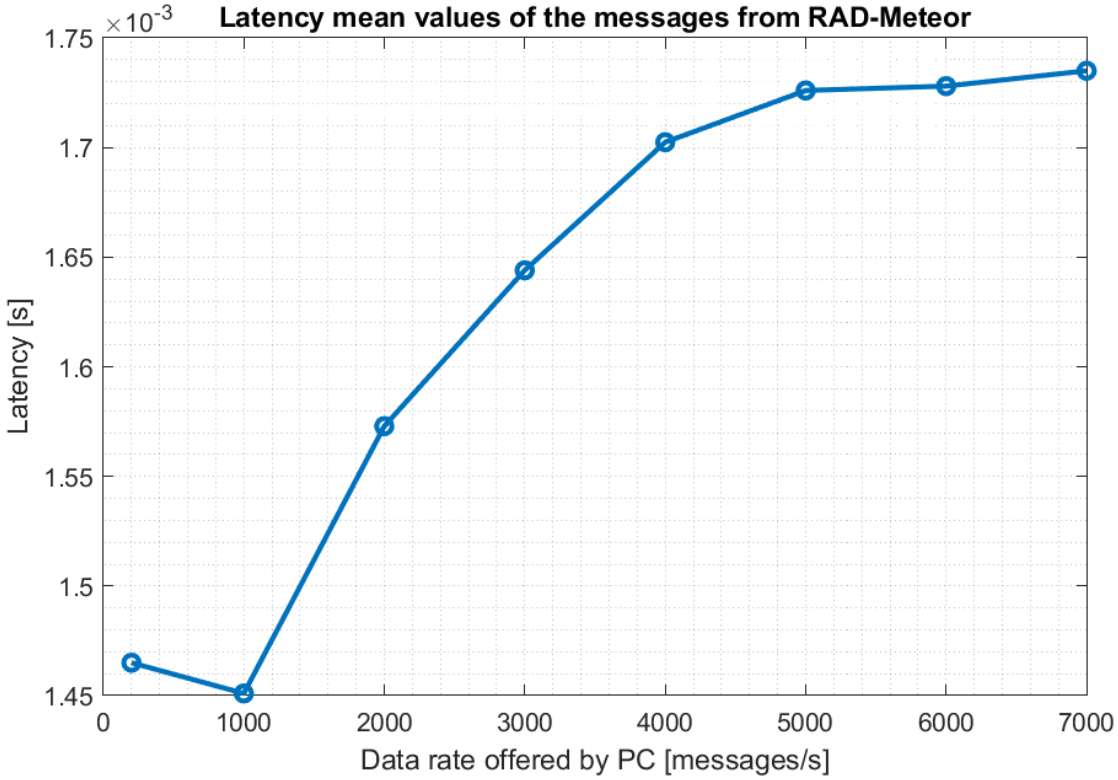

Figure 7.1: Latency mean values of the messages from RAD-Meteor

Figure 7.1 illustrates the data rate of messages sent from the PC on the x-axis, perturbing the normal transmission from the RAD-Meteor. The first data point, at 0 messages/s, represents the scenario where only RAD-Meteor messages are present on the bus. The final data point corresponds to 7000 messages/s sent from the PC. The y-axis denotes the mean latency values. The plot's trend aligns with expectations for the most part. A quite fast increase in latency is noticeable between 1000 and 5000 messages/s. At 5000 messages/s, the trend seems to stabilize, settling around approximately 1.73e-3 seconds, this suggests that the network has reached a point where further increases in message rate have minimal impact on latency. This condition is favorable, allowing for consistent latency values within a predictable window across all conditions. However, a notable drawback is that even with a small message concurrency, latency appears to be three times larger than the period of messages sent from RAD-Meteor indicating potential inefficiencies or bottlenecks in the network.
The stationarity observed can be attributed to the fact that the network is handling a quite remarkable traffic. Beyond this point, additional message transmissions don't notably increase the network load because it's almost fully utilized. Moreover, at this point, all network resources like buffer space, and processing power are almost fully engaged. Consequently, contention for network medium access and queuing delays may already be high, leading to longer wait times for message transmissions. As a result, all messages contend for network medium access similarly, and further increases in message rate don't significantly impact latency.

Another intriguing aspect to investigate would be the impact of introducing another RAD-Meteor transmitting at the same rate as the first one.

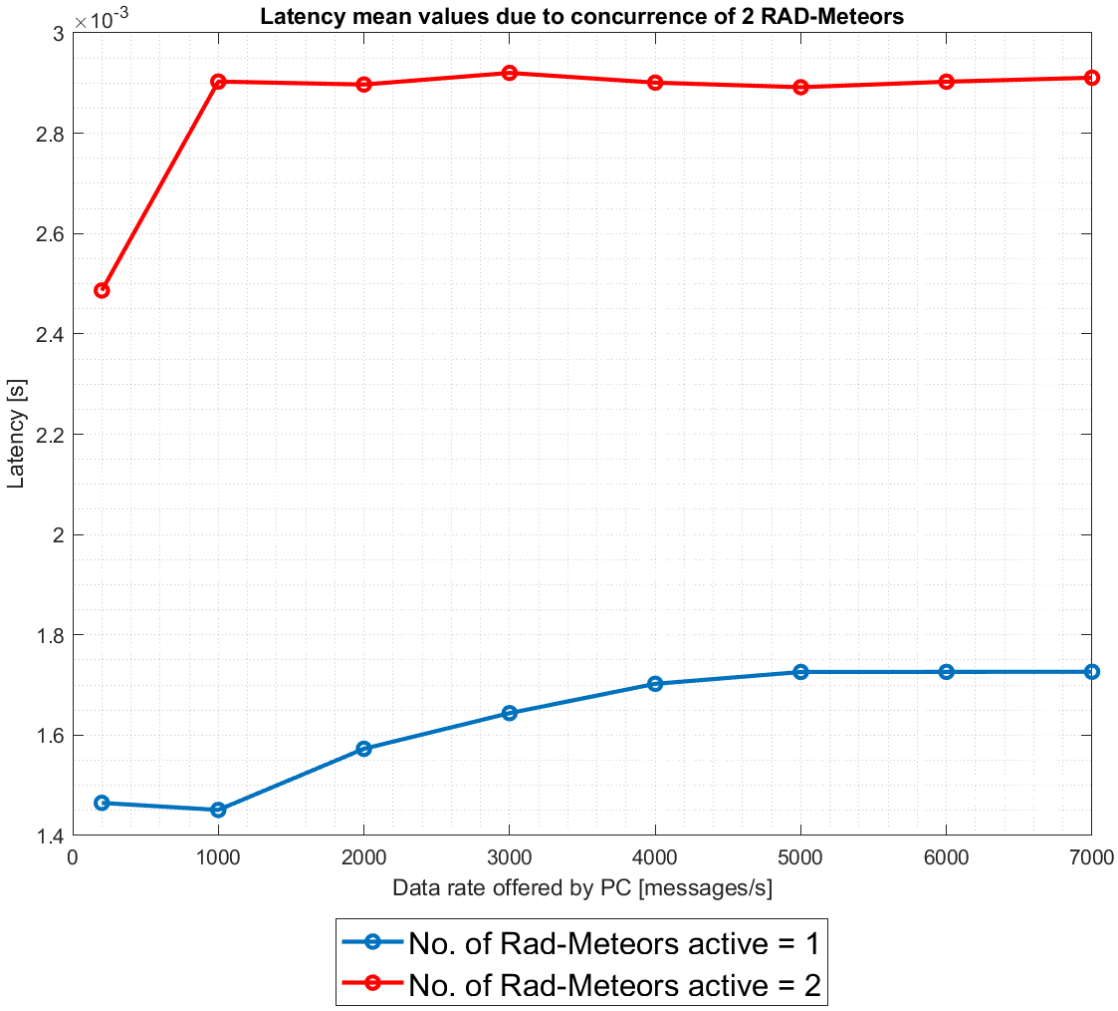

Figure 7.2: Latency mean values due to concurrence of 2 RAD-Meteors

As depicted in the Figure 7.2, there's a noticeable abrupt increase in latency when the data rate of messages sent from the PC transitions from 200 to 1000 messages/sec. This differs from the previous case, where the transition to the stationary latency value occurred gradually. Consequently, it can be inferred that the network reaches a stable state right at the second data point, much earlier than in the previous experiment (which occurred after five data points). Another interesting observation is that the average latency for the 2-RAD-Meteors case is around 2.90e-3, which is 1.7 times larger than in the first case.

7.1.2 Jitter

The next step is to evaluate performance in terms of jitter. Similar to previous observations, the jitter values are higher when both RAD-Meteors are active compared to when only one is transmitting. Specifically, when the plot stabilizes, the values are exactly double those of the single-transmission scenario.

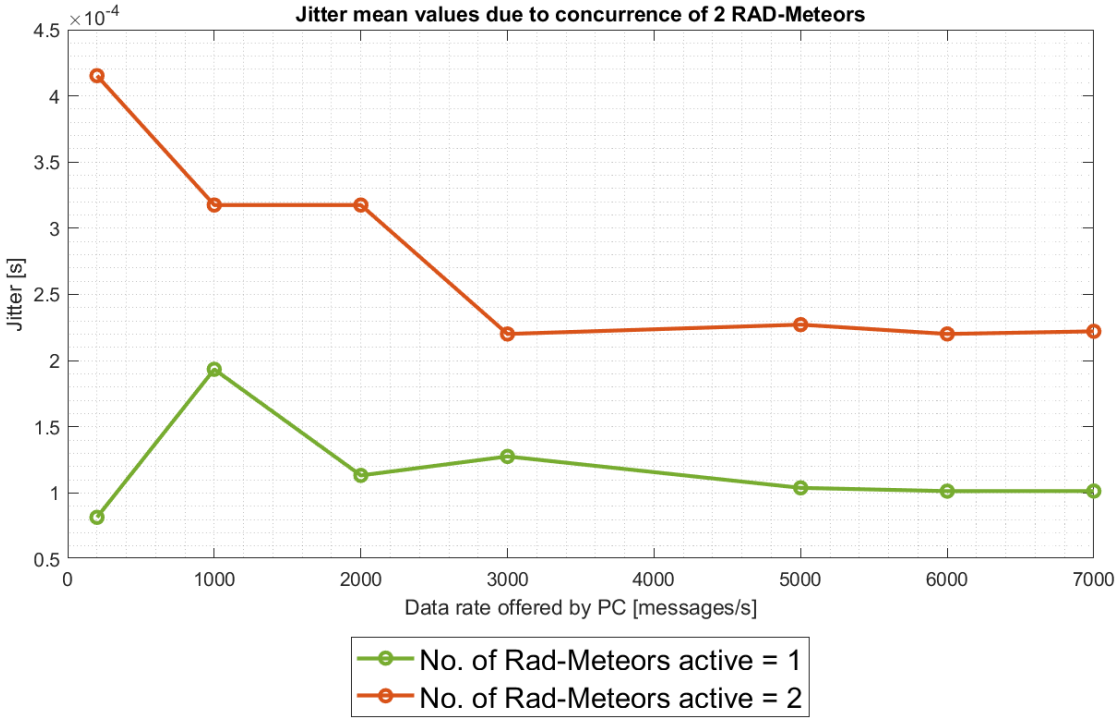

Figure 7.3: Jitter mean values due to concurrence of 2 RAD-Meteors

An interesting observation is that when only one RAD-Meteor is active, the jitter initially experiences a slight increase before stabilizing at its stationary value around 1.00e-4. Conversely, when both RAD-Meteors are active, the jitter starts from higher values and then abruptly decreases to its stationary value. Another notable pattern is that for a two RAD-Meteor transmitting, the first stationary point for latency was observed at 1000 messages/s, whereas for jitter, it was at 3000 messages/s.
As both RAD-Meteors become active simultaneously, there is a higher contention for network resources. This can result in elevated initial jitter values as the network struggles to manage the increased traffic and resolve conflicts between simultaneous transmissions. Nonetheless, as the system adapts and optimize resource allocation through PLCA algorithm, jitter may undergo a sudden decline towards its stationary value. This reduction might stem from the implementation of mechanisms aimed

at alleviating contention and prioritizing message transmissions, ultimately leading to a more stable and predictable variance in message delivery times.

7.1.3 Impact of PLCA and CSMA/CD on the packet transmission

All the procedures previously described were conducted when the PLCA arbitration method is active, however an interesting investigation might involve the comparison between PLCA and CSMA/CD. The experiment described in this section has been conducted on Wireshark. In contrast to previous cases, the metric under analysis is the number of packets per second transmitted by a single RAD-Meteor initially without the interference of data frames sent from other devices, and subsequently when the other RAD-Meteor is activated (at 14 second indicated by the red line). The first Figure 7.4 illustrates message transmission under the PLCA arbitration system, while the second Figure 7.5 depicts transmission under CSMA/CD. In the absence of any other interference factors, the packet-per-second transmission rates are identical. However, the intriguing aspect lies in how each arbitration system responds to the concurrency of the two RAD-Meteors.

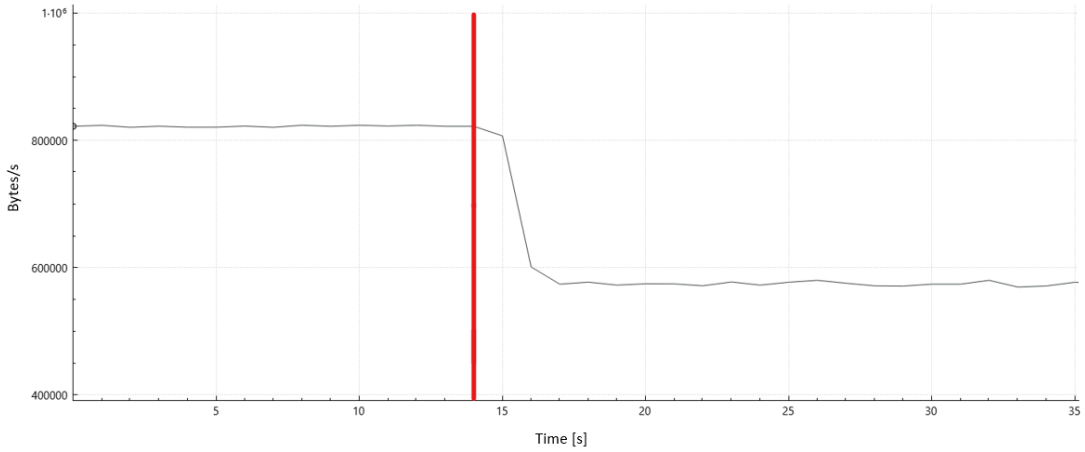

Figure 7.4: Packet transmission with PLCA arbitration system

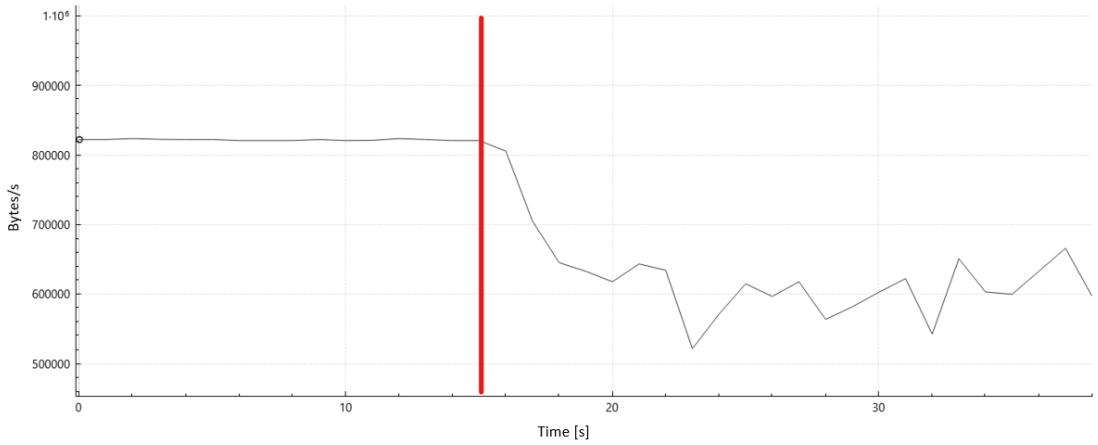

Figure 7.5: Packet transmission with CSMA/CD arbitration system

In the first case the packet quantity sharply declines upon activation of the second RAD-Meteor, while in the second plot is observable a smoother trend. Yet, under PLCA, the packet count remains consistently at 570000 bytes/sec. Conversely, with CSMA/CD, a more irregular and jagged trend is observed, despite the mean packet quantity being approximately equal to that of the PLCA case.

The difference in behavior stems from the inherent characteristics of the two arbitration methods. PLCA, prioritizing network access based on predetermined criteria like message importance or assigned priorities, ensures that upon activation of the second RAD-Meteor, the first RAD-Meteor maintains its designated priority level. This enables it to sustain a constant packet transmission rate. PLCA achieves this by dynamically adjusting access priorities or implementing mechanisms to minimize contention between the two RAD-Meteors, thereby ensuring a consistent transmission rate.
On the other hand, CSMA/CD operates on a contention-based access approach, where nodes listen for the carrier signal and attempt to transmit when the channel is idle. With the activation of the second RAD-Meteor, CSMA/CD faces heightened contention for channel access, resulting in collisions and packet retransmissions. Despite having a mean packet quantity similar to that of the PLCA case, the irregular and rugged trend observed in packet transmission under CSMA/CD reflects the dynamic nature of contention and collision resolution within the network. The PLCA coordinator by default is the RAD-Comet with ID=0, the two RAD-Meteor have respectively ID=1 and ID=2.

7.2 Examining the performance of overall network

In the first study case of this section, the focus is on the PLCA (Priority-based Logical Collision Avoidance) arbitration system, while the second study case will delineate a performance comparison between PLCA and CSMA/CD.

7.2.1 PLCA

The initial examination focuses solely on the behavior of the node constituted by the Vehicle Spy software. In this series of experiments, it has been chosen the minimum rate possible by software (5 ms), and a variable number of messages to send, from 1 message to 100 messages spaced with a step of 10. This means that the window of variation is comprised between 200 messages/s and 20000 messages/s. The experimental results of this process are depicted in the Figure 7.6.

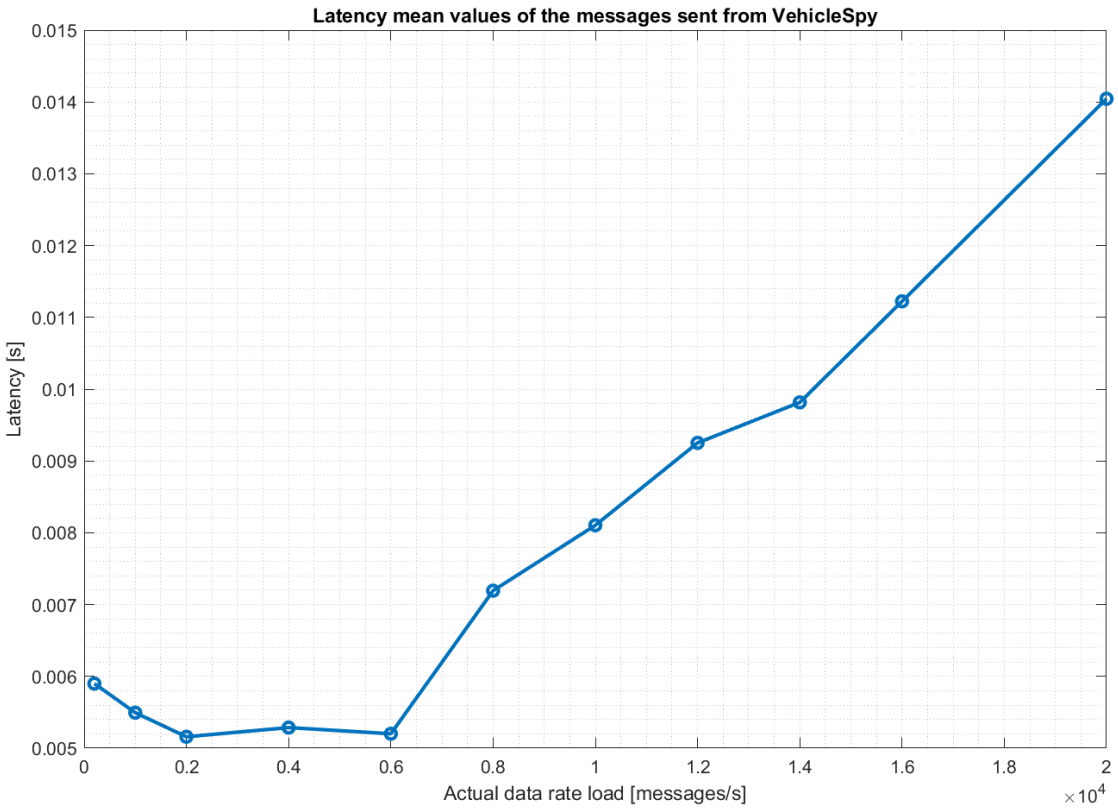

Figure 7.6: Latency mean values (Only the messages sent from Vehicle Spy)

The graph illustrates data rates on the x-axis and the corresponding median values

of latencies on the y-axis. Initially, up to the breakpoint of 6000 messages per second (mes/s), the network exhibits an almost constant behavior. However, the most notable observation is the significant increase in latencies beyond this point. Indeed, one would expect latency to increment gradually and at a steady pace as the data rate increases. However, the sudden rise in latencies beyond a certain threshold, is quite interesting compared to the RAD-Meteor latency plot in which there was a slower increase in the trend. This anomaly suggests the presence of underlying factors or conditions within the network environment that warrant further investigation and analysis. Understanding the root cause of this behavior is essential for optimizing network performance and ensuring reliable data transmission in similar scenarios. A possible reason arises from the hardware limitation leading to buffer overflows. If network devices or switches have limited buffer capacities, they may start dropping packets or experiencing increased queuing delays when the incoming message rate exceeds their processing capabilities. This factor highlights the complex interplay of various network parameters and conditions that can influence latency behavior. Identifying and addressing these issues is crucial for optimizing network performance and ensuring reliable data transmission in high-demand scenarios.

A noteworthy observation concerns some unexpected latency values, which often exceed expectations even at relatively modest data rates. An important factor to consider in understanding this behavior is the handling of message transmission by tools like Vehicle Spy. Usually tools with same purposes as Vehicle Spy maintain a fixed message rate, regardless of fluctuations in bandwidth. To achieve this consistency, when confronted with increased bandwidth, these tools may resort to generating multicast or broadcast messages. This practice ensures that the specified message rate is maintained. However, it's worth noting that these additional message types might not be detected by tools like Wireshark, commonly used for message collection. The introduction of multicast or broadcast messages alongside unicast transmissions can have a substantial impact on latency. One explanation is the potential increase in network congestion, leading to delays in message delivery. In summary, while tools like Vehicle Spy play a crucial role in monitoring communication networks, their approach to message transmission under varying bandwidth conditions can significantly affect system performance, particularly in terms of latency.

Impact of the RAD-Meteor on the performance metrics

The scenario outlined focuses on analyzing latencies in the network when only one node is transmitting, while the two RAD-Meteors remain in a passive mode, solely receiving. Another relevant experiment might investigate how the activity of the two RAD-Meteors can affect network behavior. The experiments follow the same procedure as before, with data collected when either just one or both RAD-Meteors, along with Vehicle Spy, are transmitting. The results are illustrated below in Figure 7.7.

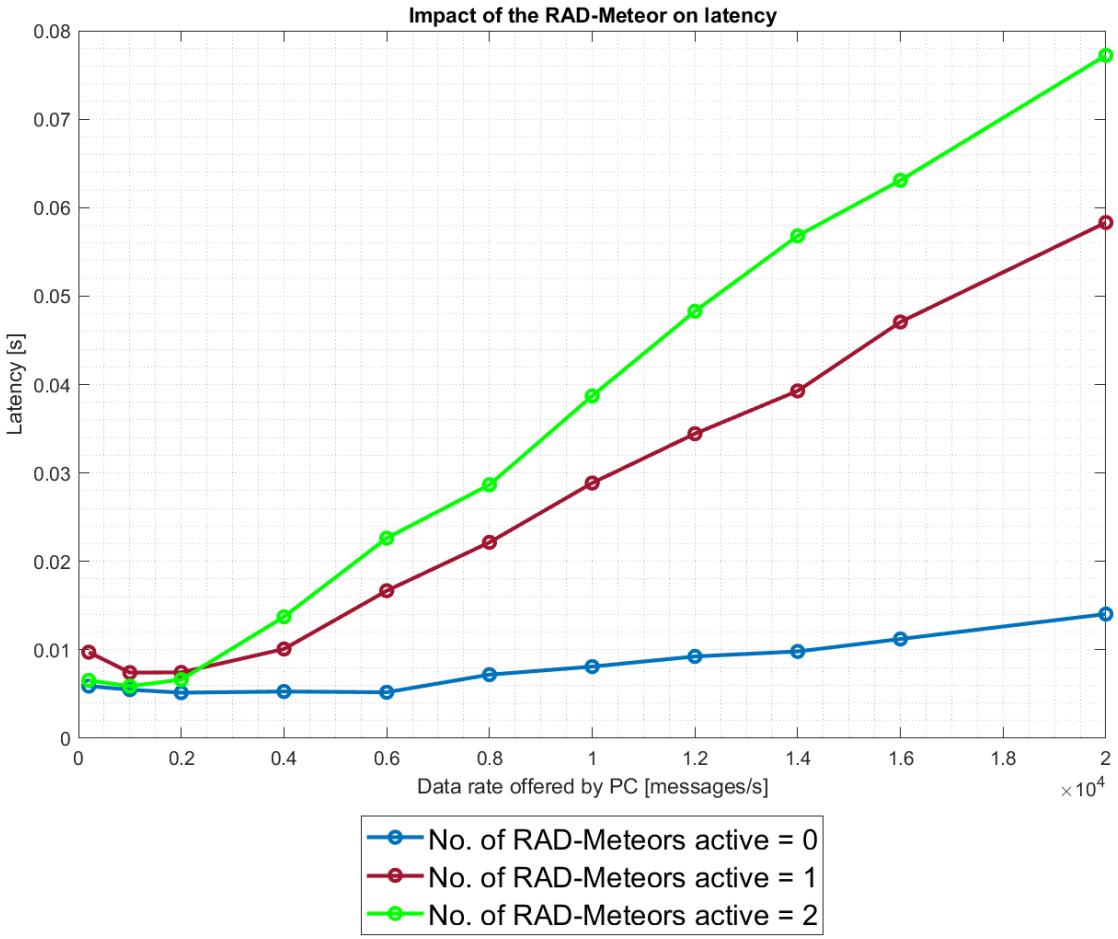

Figure 7.7: Impact of the RAD-Meteor on latency

It is evident from the plots that the introduction of two RAD-Meteors has a significant impact on latency compared to the original scenario with only one RAD-Meteor.
The plots display a steeper slope, indicating a higher rate of increase in latency as the system approaches its maximum traffic capacity. In the scenario with just one RAD-Meteor, the latency reaches 0.06 seconds when the maximum traffic capacity of the system is reached. However, in the scenario with two active RAD-Meteors, the latency increases even further, reaching 0.078 seconds at the same traffic capacity. This represents a significant increase in latency, with the latency being approximately 5 to 6 times larger than in the scenario where only the PC is transmitting. This substantial increase in latency can be attributed to the additional load introduced by the two RAD-Meteors. As more devices actively transmit data, the network experiences higher congestion and increased processing overhead, leading to delays in message transmission and reception. Consequently, the latency observed in the system is considerably higher when multiple devices are actively transmitting compared to when only one device is transmitting.
Overall, these findings highlight the critical impact of device load and network congestion on system latency, underscoring the importance of efficient resource management and network optimization strategies to mitigate latency issues in complex communication systems.

Moving forward with the analysis of jitter, it is evident that the results are quite remarkable. The observed trend aligns with expectations, as an increase in latency corresponds to an increase in jitter.

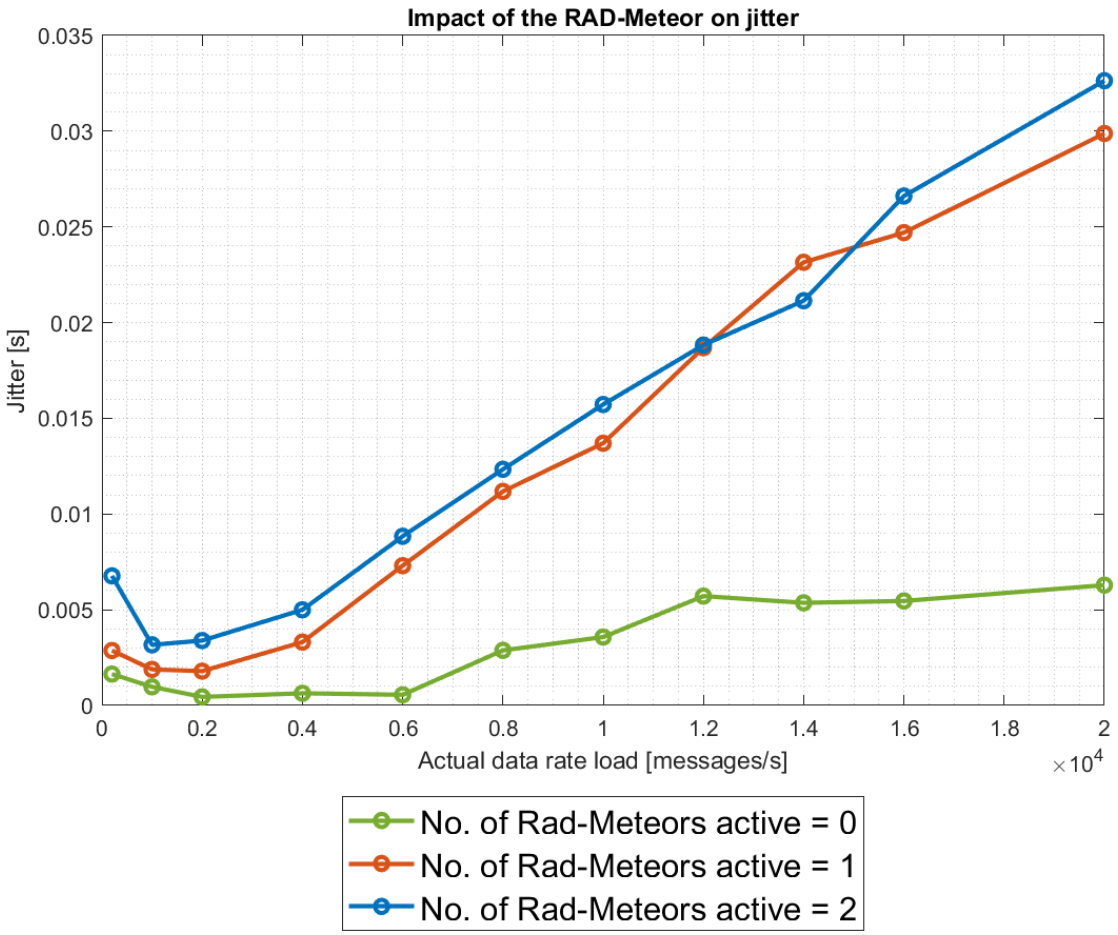

Figure 7.8: Impact of the RAD-Meteors on jitter

As the data rate increases, it becomes evident that both graphs, where the bus is also occupied by messages from the RAD-Meteor, exhibit notable trends. In the jitter plot for the scenario where only messages from Vehicle Spy occupy the bus. Furthermore, it's noticeable that the jitter increases more rapidly in scenarios where there is activity from RAD-Meteors compared to the only-Vehicle Spy case.

7.2.2 Comparison between PLCA and CSMA/CD

Latency

In this section, the investigation focuses on examining the network behavior under the influence of two distinct methods of message arbitration. While previous analyses solely relied on PLCA for arbitration, the exploration now extends to include considerations of how CSMA/CD may affect the observed trends. By introducing CSMA/CD alongside PLCA, valuable insights can be observed regarding the comparative impacts of differing arbitration methodologies on network performance. This comparative analysis aims to provide an objective evaluation of the strengths and weaknesses inherent in each arbitration method, shedding light on their implications for network stability and efficiency.

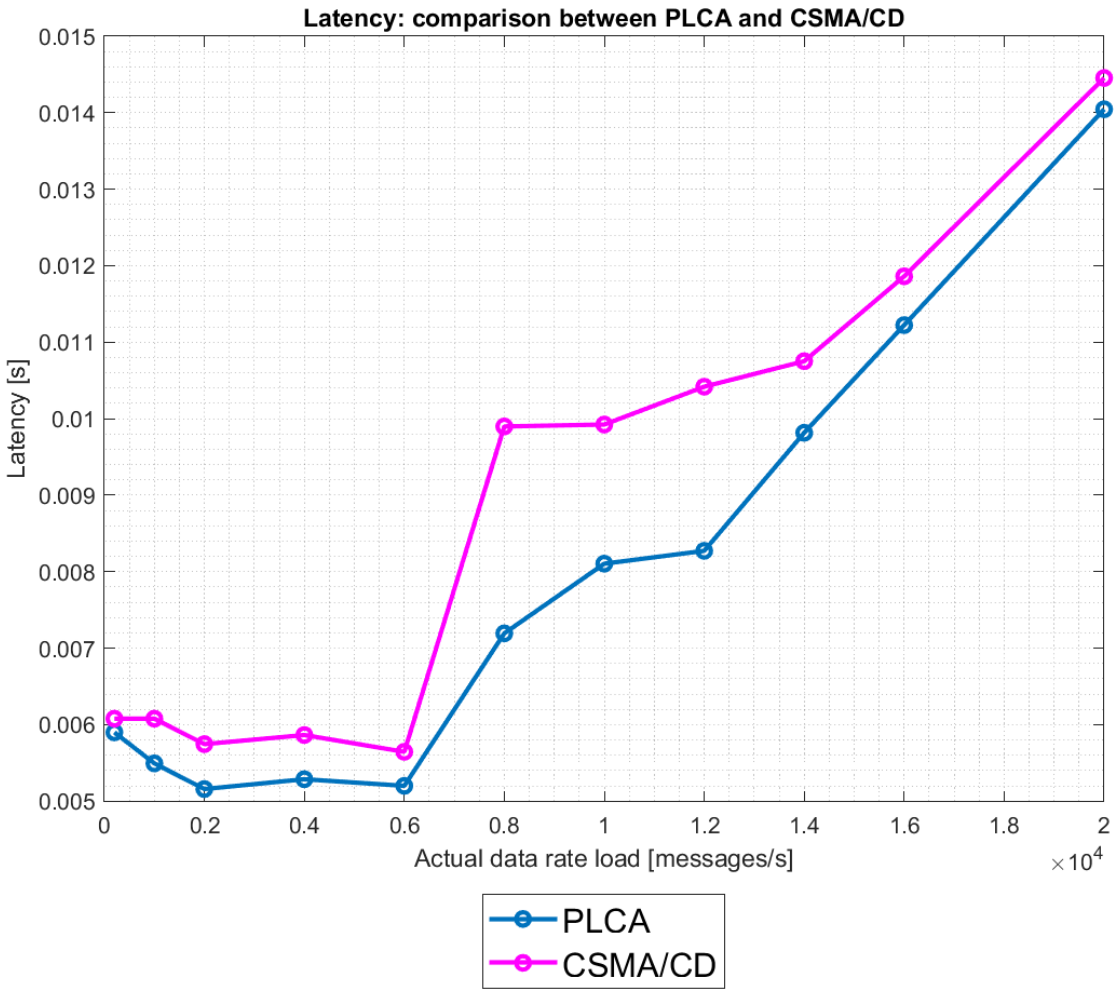

Figure 7.9: Latency: comparison between PLCA and CSMA/CD

The mean latency trends of the two systems exhibit striking similarities. Both systems maintain relatively constant latency values until reaching a message rate of 6000 messages/s. Beyond this threshold, a sudden increase in latency is observed. Following a period characterized by an increase in the data rate up to 20000 messages/s. However, an overall analysis suggests that the performance of the CSMA/CD-arbitrated system is marginally inferior to that of PLCA. In the message rate range from 200 to 4000 messages/s and from 16000 messages/s onward, the difference in latency is approximately 0.1 milliseconds. Notably, there exists an operational window where a significant discrepancy arises; particularly at message rates from 8000 to 14000, where the difference increases up to 3 milliseconds for 8000 messages/s and 2 milliseconds for 10000 and 12000 messages/s.

The observation that PLCA behavior is significantly better than CSMA/CD only within a narrow operational window could indeed be attributed to the way message transmission is handled. Particularly, as traffic volume increases, the effectiveness of PLCA may diminish, leading to higher latencies. This phenomenon could occur because the system becomes overwhelmed with the volume of data to be transmitted, causing delays in message dispatch and consequently increasing latency. One of the contributing factors to this behavior could be the firmware of the devices involved, such as RAD-Meteors, RAD-Comet, and the Vehicle Spy software. Since these components are still under development, there is room for improvement in their efficiency and performance. Enhancements to the firmware could optimize the transmission process, reducing latency and improving overall system performance. Furthermore, as the operational characteristics and requirements of the system become better understood, firmware updates can be tailored to address specific challenges, such as handling high traffic volumes more efficiently. Continuous development and refinement of the firmware can lead to better adaptation to varying network conditions and improved performance across a wider range of operational scenarios. In summary, while PLCA demonstrates superior performance over CSMA/CD within certain operational parameters, there is potential for further improvement, particularly in managing high traffic volumes and reducing latency. Continued development and optimization of firmware for the devices involved, alongside ongoing refinement of communication protocols and transmission strategies, can contribute to addressing these challenges and enhancing the overall efficiency and effectiveness of the system.

Comparing these latency graphs (Figure 7.9) with those obtained from the RAD-Meteor (Figure 7.7), it is possible to observe distinct patterns. While the RAD-Meteor's latency is quite stable until 1000 messages/sec followed by a progressive regular increase, in this scenario the plot exhibits an almost linear trend. A significant factor contributing to this disparity could be the variation in Transmission

Intervals between RAD-Meteor and Vehicle Spy. The fact that RAD-Meteor messages are transmitted every 0.0005 seconds, while those depicted in the latency plot are sent by Vehicle Spy every 0.005 seconds, implies a tenfold difference in transmission frequency. This divergence may lead to fluctuations in network congestion, buffer occupancy, and contention for channel access, ultimately impacting latency.
Furthermore, an essential consideration arises from the method of data collection. Since it wasn't feasible to directly collect information about the arrival time of data from the RAD-Meteor, as it lacks the capability to store data frames, all messages were detected and collected by Wireshark, which was monitoring data from the Ethernet network associated with the RAD-Meteor.
Various factors, such as network topology, routing paths, and device processing times, can introduce variable delays affecting latency measurements. The conditions of the Ethernet network, including traffic levels, congestion, and network utilization, can also influence latency measurements. Depending on the data collection point's location relative to network congestion points or bottlenecks, different latency patterns may be observed.

Jitter

Moving forward, the analysis will shift focus towards the performance in terms of jitter. The examination of both the PLCA and CSMA/CD cases has yielded some interesting results, which are depicted below.

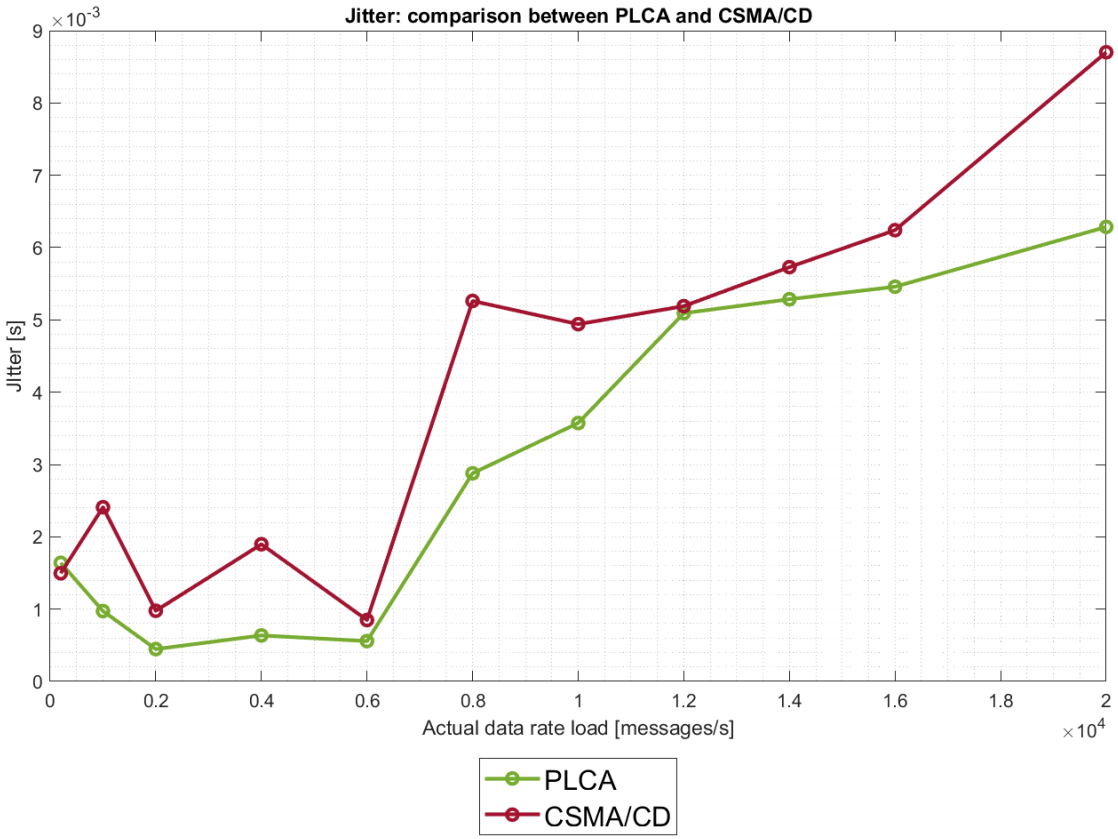

Figure 7.10: Jitter: comparison between PLCA and CSMA/CD

The results align with expectations: as latency increases, the mean jitter value also rises. A notable increase was observed when the data rate shifted from 5000 to 10000 messages per second. Subsequently, both plots continue to increase, albeit at a slower pace.
It's apparent that the jitter values are significantly higher when CSMA/CD is active compared to PLCA. Specifically, if this difference is around 0.001 seconds.

Similarly as the previous cases, the observed trend in the plot can be attributed to the factors mentioned earlier. In CSMA/CD networks, the interplay of collision detection, backoff mechanisms, contention for channel access, and network congestion typically leads to increased jitter compared to networks using PLCA. In PLCA

networks, transmissions are prioritized according to predetermined criteria rather than solely relying on channel sensing and collision detection, resulting in lower jitter.

Packet transmission

Another intriguing aspect regards the performance of the two network arbitration mechanisms in terms of bandwidth, and consequently, packet transmission efficiency.

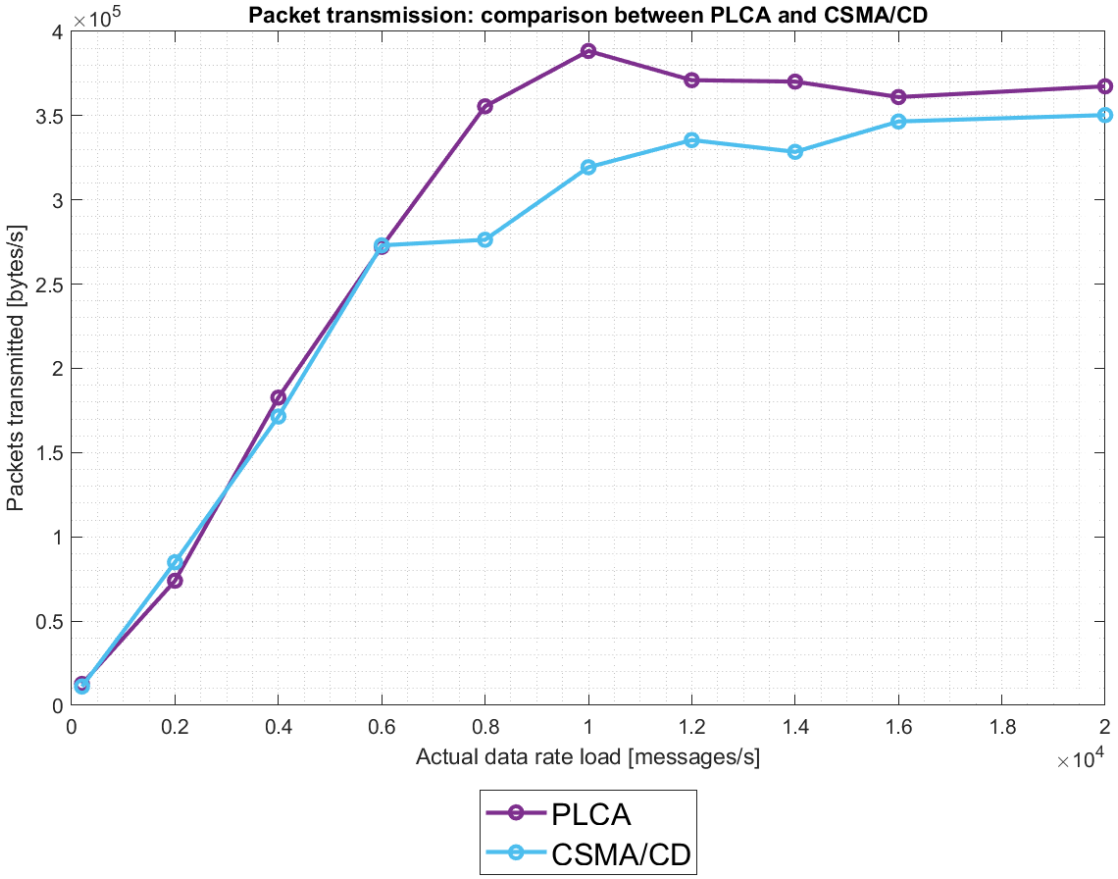

Figure 7.11: Packet transmission: comparison between PLCA and CSMA/CD

The trends for both PLCA and CSMA/CD show remarkable similarity up to 6000 messages/s. Beyond this threshold, a noticeable deviation emerges, indicating a clear performance advantage for PLCA as the message rate from Vehicle Spy increases. Between 6000 and 16000 messages/s, a substantial gap emerges between the two plots, with subsequent stabilization occurring thereafter. Even at this stabilization point, the difference between the two methods remains around 20000

bytes/s. Notably, the PLCA trend stabilizes at 12000 messages/s, three data points ahead of the CSMA/CD trend at 16000 messages/s. Overall, PLCA consistently demonstrates superior packet transmission rates, making it the preferred mechanism in this evaluation.

Chapter 8

Conclusions and further developments

The automotive industry undergoes constant change, driven by an unstoppable evolution of technologies. With the demand for increasingly sophisticated services from clients, car manufacturers must continually adapt and work on new solutions able to accommodate all the requests. It is anticipated that in the near future, new technologies will emerge to succeed the traditional CAN network, with Automotive Ethernet proving to be one of the most promising candidates. This technology promises a plethora of advanced features, including advanced driver assistance systems (ADAS), infotainment systems, cameras, and vehicle-to-vehicle communication systems. In the context of this thesis, the exploration of the 10BASE-T1S standard provides valuable insights into one of the latest Ethernet standards. It has demonstrated a commendable balance between speed, cost-effectiveness, and systemic synergy. Despite its recent introduction, 10BASE-T1S has already garnered interest from automotive manufacturers. For instance, BMW is already in the process of integrating 10BASE-T1S transceivers for ambient lighting systems in future vehicles. [69]
From this perspective, the experimental work has proven to be highly intriguing, particularly due to the innovative devices developed by IntrepidsCS, which offer a valuable demonstration of a 10BASE-T1S network. Furthermore, the experiments outlined in this thesis serve as a gateway to innovation, providing a pathway to delve deeper into this standard and its implications. Undoubtedly, they present an opportunity to delve deeper into this standard to effectively address future challenges.

Appendix A

MATLAB script

```
clc, clear all
format long

pcapReaderObj = pcapReader('MV10_03.pcap','OutputTimestampFormat','
    datetime');
decodedPackets = readAll(pcapReaderObj);

N=length(decodedPackets);

 for packetCount = 1:N
     time_vector(packetCount)=decodedPackets(packetCount).Timestamp;
 end

 %Calculate LATENCIES
 for i = 1:(N-1)
  latencies(i+1)=between(time_vector(i),time_vector(i+1));
 end
latencies_str = string(latencies);

% Initialize an array to store the numeric values corresponding to
    the seconds
latencies_numeric = zeros(size(latencies_str));

% Loop through each element in the array
for i = 1:numel(latencies_str)
    % Extract the substring containing the numeric value
    corresponding to the seconds
    temp_str = split(latencies_str(i), " ");
    seconds_str = temp_str(end);

    % Remove the 's' character from the end of the string
```

```
29      seconds_str = extractBefore(seconds_str, "s");
30
31      % Convert the extracted string to a numeric value
32      if ~isempty(seconds_str)
33          latencies_numeric(i) = str2double(seconds_str);
34      else
35          latencies_numeric(i) = NaN; % Assign NaN if the string is
        empty
36      end
37  end
38
39  % Display the numeric values corresponding to the seconds
40  % disp('Seconds:');
41  % disp(latencies_numeric);
42
43  % Compute the mean and standard deviation of latencies
44  mean_latency = mean(latencies_numeric);
45  std_latency = std(latencies_numeric);
46  disp(['Mean Latency: ', num2str(mean_latency)]);
47  disp(['Standard Deviation of Latencies: ', num2str(std_latency)]);
48  figure(1), plot(latencies_numeric, '-o')
49  xlabel('Index')
50  ylabel('Latency (seconds)')
51  title('Latency Plot')
52
53  %Compute the jitter
54  jitter = abs(latencies_numeric - mean_latency);
55  disp(['Jitter: ', num2str(mean(jitter))])
56  % figure(1), hold on, plot(jitter, '-o','Color', 'red')
57  % xlabel('Index')
58  % ylabel('Jitter (seconds)')
59  % title('Jitter Plot')
60
61  mean_latency, mean(jitter)
```

Bibliography

[1] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert. «Trends in Automotive Communication Systems». In: *Proceedings of the IEEE* 93.6 (2005), pp. 1204–1223. DOI: 10.1109/JPROC.2005.849725 (cit. on p. 2).

[2] T. Nolte, H. Hansson, and L. Lo Bello. «Automotive communications-past, current and future». In: *Proceedings of 10th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2005, September 19-22, 2005, Catania, Italy*. IEEE, 2005. DOI: 10.1109/ETFA.2005.1612631. URL: https://doi.org/10.1109/ETFA.2005.1612631 (cit. on pp. 2, 9, 14–16).

[3] L. Lo Bello, G. Patti, and L. Leonardi. «A Perspective on Ethernet in Automotive Communications—Current Status and Future Trends». In: *Applied Sciences* 13.3 (2005), p. 1278. URL: https://doi.org/10.3390/app13031278 (cit. on pp. 2, 14, 16, 42).

[4] F. Winters, C. Mielenz, and G. Hellestrand. «Design process changes enabling rapid development». In: (Jan. 2004). Society of Automotive Engineers, Warrendale, PA, pp. 21–85. URL: https://www.researchgate.net/publication/228782214_Design_process_changes_enabling_rapid_development (cit. on p. 3).

[5] M. Ashjaei, L. Lo Bello, M. Daneshtalab, G. Patti, S. Saponara, and S. Mubeen. «Time-Sensitive Networking in Automotive Embedded systems: State of the art and Research opportunities». In: *J. Syst. Archit.* 117 (2021), p. 102137. URL: https://api.semanticscholar.org/CorpusID:234814871 (cit. on pp. 3, 44).

[6] Ixia. *Automotive Ethernet: An Overview*. [Online; accessed on 25-10-2023]. 1999. URL: https://support.ixiacom.com/sites/default/%20_les/resources/whitepaper/ixia-automotive-ethernet-primerwhitepaper_1.pdf (cit. on pp. 3, 6, 8, 15).

[7] P. Hank, S. Müller, O. Vermesan, and J. Van Den Keybus. «Automotive Ethernet: In-vehicle networking and smart mobility». In: *2013 Design, Automation Test in Europe Conference Exhibition (DATE)*. 2013, pp. 1735–1739. DOI: 10.7873/DATE.2013.349 (cit. on pp. 4, 5).

[8] K. Matheus and BMW GROUP. «Evolution of Ethernet-Based Automotive Networks: Faster and Cheaper». [Online; accessed on 27-10-2023]. 2018. URL: https://standards.ieee.org/wp-content/uploads/import/documents/other/d1-03_matheus_evolution_of_ethernet_based_automotive_networks.pdf (cit. on pp. 6, 7, 40).

[9] K. Van Cleave and Raj Jain. *A Survey of Automotive Ethernet Technologies and Protocols*. [Online; accessed on 27-10-2023]. 2019. URL: https://www.cse.wustl.edu/~jain/cse570-19/ftp/auto_eth/index.html (cit. on pp. 9, 14).

[10] ASCENTEN. *Automotive*. [Online; accessed on 30-10-2023]. URL: https://ascenten.net/automotive-electronics-design.php (cit. on p. 10).

[11] T. Kouthon. «Automotive Architectures: Domain, Zonal and the Rise of Central». In: *EETimes* (2022) (cit. on pp. 10, 11).

[12] H. Askaripoor, M. Hashemi Farzaneh, and A. Knoll. «E/E Architecture Synthesis: Challenges and Technologies». In: *Electronics* 11.4 (2022). ISSN: 2079-9292. DOI: 10.3390/electronics11040518. URL: https://www.mdpi.com/2079-9292/11/4/518 (cit. on p. 11).

[13] P. Aberl, S. Haas, and A. Vemur. «How a Zone Architecture Paves the Way to a Fully Software-Defined Vehicle». In: Texas Instruments. URL: https://www.ti.com/lit/wp/spry345b/spry345b.pdf?ts=1701078139616&ref_url=https%253A%252F%252Fwww.google.com%252F (cit. on pp. 12, 13).

[14] J. Klaus-Wagenbrenner. *Zonal EE Architecture: Towards a Fully Automotive Ethernet–Based Vehicle Infrastructure*. [Online; accessed on 2023-11-27]. Visteon. Sept. 4, 2019. URL: https://standards.ieee.org/wp-content/uploads/import/documents/other/eipatd-presentations/2019/D1-04_KLAUS-Zonal_EE_Architecture.pdf (cit. on p. 13).

[15] CSS Electronics. *Lin Bus Explained - A simple intro*. [Online; accessed on 30-10-2023]. 2023. URL: https://www.csselectronics.com/pages/lin-bus-protocol-intro-basics (cit. on p. 14).

[16] Siemens. *Automotive Ethernet*. [Online; accessed on 18-11-2023]. URL: https://www.plm.automation.siemens.com/global/en/our-story/glossary/what-is-automotive-ethernet/109722 (cit. on p. 17).

[17] *La rete ethernet e gli standard IEEE 802.3*. [Online; accessed on 11-11-2023]. 2020. URL: https://studioreti.it/it/book_2020/Cap-6.pdf (cit. on pp. 17, 22, 60).

[18] C. Kozierok, C. Correa, R. Boatright, and J. Quesnelle. *Automotive Ethernet: The Definitive Guide*. Intrepid Control Systems, 2014 (cit. on pp. 18, 19).

[19] JavaTpoint. *Ethernet Frame Format*. [Online; accessed on 12-11-2023]. URL: https://www.javatpoint.com/ethernet-frame-format (cit. on pp. 19, 20).

[20] K. Matheus and T. Konigseder. *Auromotive Ethernet - Second Edition*. Cambridge University Press, 2017, p. 7 (cit. on p. 20).

[21] Ignas Anfalovas. *Logical Link Control (LLC) and Media Access Control (MAC) Sublayers Explained*. [Online; accessed on 16-11-2023]. 2022. URL: https://www.ipxo.com/blog/llc-and-mac/ (cit. on pp. 21, 23).

[22] *Il Progetto 802*. [Online; accessed on 16-11-2023]. URL: https://www.itimarconi.ct.it/sezioni/didatticaonline/informatica/lezionidisistemi/quinta/IEEE_802.htm#:~:text=Il%20progetto%20IEEE%20802%20definisce,ISO%20con%20la%20sigla%208802. (cit. on p. 21).

[23] CCNA. *Unicast, Multicast, and Broadcast Addresses*. [Online; accessed on 16-11-2023]. URL: https://study-ccna.com/unicast-multicast-and-broadcast-addresses/ (cit. on p. 22).

[24] Academic Acceleretor. *Logical Link Control*. [Online; accessed on 16-11-2023]. URL: https://academic-accelerator.com/encyclopedia/logical-link-control (cit. on p. 23).

[25] Wikipedia contributors. *Logical link control — Wikipedia, The Free Encyclopedia*. [Online; accessed 13-March-2024]. 2024. URL: https://en.wikipedia.org/w/index.php?title=Logical_link_control&oldid=1195556376 (cit. on p. 23).

[26] Profinet University. *Network Reference Model*. [Online; accessed on 18-11-2023]. URL: https://profinetuniversity.com/industrial-automation-ethernet/network-reference-model/ (cit. on p. 24).

[27] K. Matheus and T. Königseder. *Auromotive Ethernet - Second Edition*. Cambridge University Press, 2017. Chap. 5 (cit. on pp. 25–27, 36, 38, 39, 42).

[28] C. A. Shoniregun. «Internet Protocol Versions 4 (IPV4) and 6 (IPV6)». In: *Synchronizing Internet Protocol Security (SIPSec)*. Boston, MA: Springer US, 2007, pp. 75–106. ISBN: 978-0-387-68569-4. DOI: 10.1007/978-0-387-68569-4_3. URL: https://doi.org/10.1007/978-0-387-68569-4_3 (cit. on p. 25).

[29] The Open University. *Data networks and IP addresses*. [Online; accessed 19-November-2023]. 2023. URL: https://www.open.edu/openlearncreate/mod/oucontent/view.php?id=129584&printable=1 (cit. on p. 25).

[30] O. Babatunde and O. Al-Debagy. «A Comparative Review Of Internet Protocol Version 4 (IPv4) and Internet Protocol Version 6 (IPv6)». In: *ArXiv* abs/1407.2717 (2014). Department of Information Systems Engineering, Cyprus International University. URL: https://api.semanticscholar.org/CorpusID:8085884 (cit. on pp. 26, 27).

[31] M. Pramatarov. *What is IPv4? Everything you need to know — ClouDNS*. [Online; accessed 19-November-2023]. 2023. URL: https://www.cloudns.net/blog/what-is-ipv4-everything-you-need-to-know/ (cit. on p. 26).

[32] A. S. Gillis. *IPv6 address — TechTarget*. [Online; accessed 19-November-2023]. 2020. URL: https://www.techtarget.com/iotagenda/definition/IPv6-address#:~:text=The%20specific%20layout%20of%20an,ID%20and%20the%20interface%20ID. (cit. on p. 27).

[33] D. El Idrissi, N. Elkamoun, and R. Hilal. «Study of the impact of the transition from IPv4 to IPv6 based on the tunneling mechanism in mobile networks». In: *Procedia Computer Science* 191 (2021). The 18th International Conference on Mobile Systems and Pervasive Computing (MobiSPC), The 16th International Conference on Future Networks and Communications (FNC), The 11th International Conference on Sustainable Energy Information Technology, pp. 207–214. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2021.07.026. URL: https://www.sciencedirect.com/science/article/pii/S1877050921014216 (cit. on p. 27).

[34] Wikipedia contributors. *Network socket — Wikipedia, The Free Encyclopedia*. [Online; accessed 20-11-2023]. 2023. URL: https://en.wikipedia.org/w/index.php?title=Network_socket&oldid=1181923753 (cit. on p. 28).

[35] A. Pattavina. *Internet e Reti - Fondamenti*. Third edition. Pearson, 2022, pp. 271–282 (cit. on pp. 28–33).

[36] A. Pattavina. *Internet e Reti - Fondamenti*. Third edition. Pearson, 2022, pp. 223–262 (cit. on pp. 34, 35).

[37] I. Grigorik. *High Performance Browser Networking: What every web developer should know about networking and web performance*. First edition. O'Reilly Media, 2013. Chap. 9 (cit. on p. 35).

[38] *SOME/IP*. Protocol Specification. Document Identification No 696, Release 1.3.0. 2017 (cit. on pp. 36–39).

[39] L. Völker. «One for All; Interoperability from AUTOSAR to GENIVI». In: *1st Ethernet IP @ Automotive Technology Day* (2011) (cit. on p. 40).

[40] M. L. Han, B. Kwak, and H. K. Kim. *TOW-IDS: Automotive Ethernet Intrusion Dataset*. 2022. DOI: 10.21227/bz0w-zc12. URL: https://dx.doi.org/10.21227/bz0w-zc12 (cit. on p. 41).

[41] PreSonus. *An Introduction to AVB Networking*. [Online; accessed 26-November-2023]. URL: https://legacy.presonus.com/learn/technical-articles/an-introduction-to-avb-networking (cit. on p. 41).

[42] Boatright, R. *Understanding IEEE 1722 AVB Transport Protocol - AVBTP*. 2009. URL: https://www.ieee802.org/1/files/public/docs2009/avb-rboatright-p1722-explained-0903.pdf (cit. on p. 42).

[43] M. A. Hussain, H. Jin, Z. A. Hussien, Z. A. Abduljabbar, S. H. Abbdal, and A. Ibrahim. «ARP Enhancement to Stateful Protocol by Registering ARP Request». In: *2016 International Conference on Network and Information Systems for Computers (ICNISC)*. 2016, pp. 31–35. DOI: 10.1109/ICNISC.2016.017 (cit. on p. 45).

[44] Ignas Anfalovas. *What Is Address Resolution Protocol? A Beginner's Guide to ARP*. [Online; accessed on 26-11-2023]. 2022. URL: https://www.ipxo.com/blog/address-resolution-protocol/ (cit. on p. 45).

[45] *What is Address Resolution Protocol (ARP)?* [Online; accessed on 26-11-2023]. 2022. URL: https://www.fortinet.com/resources/cyberglossary/what-is-arp (cit. on p. 46).

[46] K. Matheus and T. Konigseder. *Auromotive Ethernet - Second Edition*. Cambridge University Press, 2017, p. 5 (cit. on p. 47).

[47] D. Law. «IEEE 802 Standards Overview». [Online; accessed on 18-11-2023]. June 2008. URL: https://www.itu.int/dms_pub/itu-t/oth/06/13/T06130000010005PDFE.pdf (cit. on p. 47).

[48] D. Quaglia. «Ethernet e la famiglia di protocolli IEEE 802». [Online; accessed on 17-11-2023]. 2018. URL: https://www.di.univr.it/documenti/OccorrenzaIns/matdid/matdid522226.pdf (cit. on p. 47).

[49] ComputerNetworkingNotes. *Ethernet Standards and Protocols Explained*. [Online; accessed on 16-11-2023]. 2023. URL: https://www.computernetworkingnotes.com/networking-tutorials/ethernet-standards-and-protocols-explained.html (cit. on p. 49).

[50] A. Quine. *A Brief Overview of Ethernet History*. [Online; accessed on 2023-11-30]. ITPRC. Jan. 8, 2008. URL: https://www.itprc.com/overview-of-ethernet-history/ (cit. on p. 49).

[51] *10BASE2, Encyclopedia, Science News Research Reviews*. [Online; accessed on 2023-11-30]. Academic Accelerator. URL: https://academic-accelerator.com/encyclopedia/10base2 (cit. on pp. 49–51).

[52] L. Zheng, J. Yu, Q. Yang, and F. Gao Y.and Sun. «Vibration wave downhole communication technique». In: *Petroleum Exploration and Development* 44 (Apr. 2017), pp. 321–327. DOI: 10.1016/S1876-3804(17)30037-X (cit. on p. 49).

[53] G. Mangioni. *Ethernet*. [Online; accessed on 2023-11-30]. Università di Catania. URL: http://www.diit.unict.it/users/gmangioni/teaching/architettura_2014_2015/materiale/ethernet_802.15.pdf (cit. on p. 50).

[54] Zola, A. *10BASE-T — TechTarget*. [Online; accessed 13-December-2023]. 2021. URL: https://www.techtarget.com/searchnetworking/definition/10BASE-T (cit. on pp. 51, 56).

[55] J. McCool. «100BASE-T: An Overview». In: Bay Networks. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=485300 (cit. on p. 53).

[56] Wikipedia contributors. *Gigabit Ethernet — Wikipedia, The Free Encyclopedia*. [Online; accessed 9-January-2024]. 2023. URL: https://en.wikipedia.org/w/index.php?title=Gigabit_Ethernet&oldid=1187491578 (cit. on pp. 53, 54).

[57] D. Porter. «100BASE-T1 Ethernet: the evolution of automotive networking». In: Texas Instruments. 2018. URL: https://www.ti.com/lit/wp/szzy009/szzy009.pdf?ts=1704781325452&ref_url=https%253A%252F%252Fwww.google.com%252F (cit. on p. 55).

[58] F. Hurley. *Why 10BASE-T1S Is the Missing Ethernet Link for Automotive Communications?* [Online; accessed on 2023-12-13]. Analog Devices. URL: https://www.analog.com/en/thought-leadership/why-10base-t1s-is-the-missing-ethernet-link.html (cit. on p. 56).

[59] *10BaseT1S Protocol-Overview of the Protocol and Decoding using Tektronix Oscilloscope and application software from Prodigy Technovations*. [Online; accessed on 2024-01-17]. Prodigy Technovations. Aug. 2023. URL: https://prodigytechno.com/10baset1s-protocol/ (cit. on p. 57).

[60] M. Tazebay J. Cordaro A. Chini. *New Preamble Proposal for 10BASE-T1S*. [Online; accessed on 2024-01-17]. Broadcom. Dec. 2017. URL: https://www.ieee802.org/3/cg/public/adhoc/cordaro_8023cg_short_reach_new_preamble_proposal_1220.pdf (cit. on p. 57).

[61] R. Eckelt and GRL Team. *Introduction to 10BASE-T1S Automotive Ethernet Test Standards*. [Online; accessed on 2023-12-13]. GRANITE RIVER LABS BLOG. Dec. 7, 2023. URL: https://www.graniteriverlabs.com/en-us/technical-blog/automotive-ethernet-10-base-t1s#:~:text=Multidrop%20topology,and%20over%20the%20same%20channel. (cit. on pp. 58, 59).

[62] Wikipedia contributors. *Differential Manchester encoding — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Differential_Manchester_encoding&oldid=1201657753. [Online; accessed 14-March-2024]. 2024 (cit. on p. 58).

[63] *What Is 10Base-T1S Automotive Ethernet?* Tech. rep. Teledyne LeCroy, Mar. 2023 (cit. on pp. 59, 65, 66).

[64] J. G. Looby. *Ethernet CSMA/CD*. [Online; accessed on 2024-01-11]. CISS 100 – Computing Information Sciences. URL: https://ciss100.com/?page_id=203/wp-admin/install.php (cit. on pp. 60, 62).

[65] IEEE 802. *PLCA Frequently Asked Questions (FAQ)*. [Online; accessed on 2024-01-11]. 2018. URL: https://www.ieee802.org/3/cg/public/July2018/PLCA%20FAQ.pdf (cit. on p. 64).

[66] *Vehicle Spy Help Documentation*. [Online; accessed on 2024-01-17]. Intrepid Control Systems. 2023. URL: https://docs.intrepidcs.com/vspy-3-documentation/ (cit. on p. 68).

[67] *RAD-Comet 2 - 10BASE-T1S Development Interface with 100/1000BASE-T1, 10/100/1000BASE-T, and CAN FD*. [Online; accessed on 2024-01-17]. Intrepid Control Systems. 2023. URL: https://intrepidcs.com/products/automotive-ethernet-tools/media-converters/rad-comet-10base-t1s-development-interface/ (cit. on p. 69).

[68] *RAD-Meteor - 10BASE-T1S*. [Online; accessed on 2024-01-17]. Intrepid Control Systems. 2023. URL: https://intrepidcs.com/products/automotive-ethernet-tools/rad-meteor-low-cost-10base-t1s-interface/ (cit. on p. 70).

[69] *ADI and the BMW Group Join Forces to Provide Industry-Leading 10MB Ethernet for Automotive, Enabling Software-Defined Vehicles*. [Online; accessed on 2024-03-19]. Analog Devices, Inc. URL: https://investor.analog.com/news-releases/news-release-details/adi-and-bmw-group-join-forces-provide-industry-leading-10mb (cit. on p. 95).