Tampere University

Jawad Ahmad

ASSESSMENT OF INDUSTRIAL NETWORKING TECHNOLOGIES FOR SUPPORT OF TIME-CRITICAL COMMUNICATION

Master's Thesis
Faculty of Information Technology and Communication Sciences
Dmitri Moltchanov
Evgeny Kucheryavy
May 2023

ABSTRACT

Jawad Ahmad: Assessment of Industrial Networking Technologies for Support of Time-Critical Communication
Master's Thesis
Tampere University
Advance Studies in Communication Systems and Networks
May 2023

Industry 4.0 is the current and future of the automation industry, and real-time systems play a vital role in its success. The automotive and automation industry has benefitted from the efficiency and productivity of embedded systems in both past and present and with more demanding schedules and time-critical applications taking their place in the modern world the importance of real-time systems cannot be overstated. For time-critical communication on an industrial level, there have been major improvements. The newest player is Time Sensitive Networks. It is packed with standards targeting time-critical transmissions. There is not much work done on the practical implementation of Time Sensitive Networks. A handful of research was available on Time Sensitive Networks, with most of them being software and simulation-based. The main focus of this study was data link layer comparisons of Ethernet and Time Sensitive Networks. The comparison-based analysis was done on Ethernet IEEE 802.3 ab and Time Sensitive Networks IEEE 802.1. The two main standards that will be used are IEEE 802.1 Qbu frame preemption and IEEE 802.1 Qbv enhancement for scheduled traffic. A series of tests were performed on Ethernet and Time Sensitive Networks to find the best possible configurations. The last step was to apply all the information gathered from the testing phase. This was achieved by preparing a real life scenario based network, and then running the Ethernet and Time Sensitive Networks settings on the same setup. The last test drew comparisons on parallel execution. The final verdict was on jitter and latency performances.

Keywords: Ethernet IEEE 802.3 ab, IEEE 802.1 Qbu, Time Sensitive Networks, IEEE 802.1 Qbv, NXP LS 1028ARDB, Industrial Ethernet, Latency, Jitter

PREFACE

A Master's degree holds great significance in my family's eyes. It was always a dream of my family to have a member with an international master's degree. I can say with immense pleasure that it would be me doing it.

First of all, I would like to thank Wapice Oy to provide me the opportunity to do my thesis with them. A special mention to Petri Karjanlahti, who was my supervisor and helped me throughout my thesis work. I would like to appreciate all the help provided by Aurelian Pop, Teemu Rantala, Konsta Boman, and Lauri Välimaa

I would thank Dmiti Moltchanov for providing guidance during the thesis work. Lastly, I would like to give special thanks to the university, my family, and all my friends who helped me through my journey.

Tampere, 12 May 2023

Jawad Ahmad

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

ASCII	American Standard Code for Information Interchange
ARM	Advanced RISC Machine
AVB	Audio Video Bridging
CAN	Controller Area Network
CSMA	Carrier Sense Multiple Access
CSMA-CD	CSMA Collision Detection
CSMA-CA	CSMA Collision Avoidance
EtherCAT	Ethernet for Control Automation Technology
eMAC	Express Media Access Control
IEC	International Electrotechnical Commission
Gbps	Giga bits per second
GLONASS	Globalnaya Navigazionnaya Sputnikovaya Sistem
HTTP	HyperText Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ISA	International Society of Automation
ICMP	Internet Control Message Protocol
Kbps	Kilo bits per second
LS1028ARDB	Layerscape LS1028A Reference Design Board
Mbps	Mega bits per seconds
MBps	Mega bytes per second
NXP	NXP Semiconductor Ltd
OSI	Open Systems Interconnection
pMAC	Preemptable Media Access Control
RJ	Registered Jack
RS	Recommended Standards
RTSP	Real-Time Streaming Protocol
RTU	Remote Terminal Unit
SD	Secure Digital
SSH	Secure Shell
TC	Traffic control
TCP	Transmission Control Protocol
UART	Universal Asynchronous Receiver / Transmitter
USB	Universal Serial Bus
UDP	User Datagram Protocol
WirelessHART	Wireless Highway Addressable Remote Transducer Protocol
WRM	Wapice Remote Monitoring

1. INTRODUCTION

TSN is a relatively new technology compared to Ethernet. It aims to provide real-time messaging to target the current and future of the industry. With a focus on automation and automotive, TSN packs just the right amount of power needed to fulfill the requirements. With features such as time scheduling and frame pre-emption, it reduces jitter and latency levels to an extent where timely delivery of messages is made possible. The current market is dominated by industrial Ethernet [5]. TSN looks like a possibility, but not in the immediate future. However, it is down to the availability of Ethernet than any issue with TSN. Therefore, a natural question is how Ethernet exactly compares with TSN. We cannot compare industrial Ethernet directly with TSN, as TSN is a data link layer protocol, whereas most of the standards or protocols generally known to be industrial Ethernet are using more than just the data link layer.

1.1 Motivation

Given the difference in nature of industrial Ethernet and TSN, it is only fair to compare Ethernet and TSN strictly on the data link layer levels. TSN is a flexible protocol, which can run industrial Ethernet protocols as well. The only question left to answer is how much better is TSN when compared to Ethernet on a data link layer basis. When dealing with real-time systems or time-critical communication, the two primary components are jitter and latency. Therefore, the logical comparison between Ethernet and TSN should be based on them.

Wapice Oy is a big company, and well aware of the technologies being used in the industry. As mentioned above, industrial Ethernet is still dominating the automation industry, but a TSN takeover seems inevitable. To be ahead of the competition as well as to spread the technology wings, TSN was a topic of interest for the company. However, the vast domain of TSN cannot be discussed under one thesis, so it was decided to make comparison-based research. The two standards IEEE 802.1 Qbv and IEEE 802.1 Qbu were the most desired of all the TSN standards, as a result, a study that included both of them was finalized by the company.

1.2 Overview of the Thesis

The whole thesis is divided into different chapters. The next chapter deals with all the background literature available related to communication systems. It further extends to discuss the term switched Ethernet and explains all the industrial Ethernet technologies standards and protocols in the market. A comparison is also drawn between all the well-known industrial Ethernet technology to provide an overview. Lastly, the chapter introduces the concept of TSN and the famous standards related to TSN, specifically IEEE 802.1 Qbv and IEEE 802.1 Qbu. Chapter 3 focuses on the research goal. It will describe the type of devices required and the reasoning behind the selection of the gadgets. It will then provide information regarding how to configure the devices in order to achieve the required testing. The next chapter is the chapter in which all the testing is performed as well as the analysis against each of them is provided. The limiting factors in each of the tests will be discussed. The tests will keep on improving the network, so all the enhancements will be explained. All the data gained from these tests will be used to optimize the network in the best possible way. Finally, the optimized TSN network and default Ethernet will run the same network configuration and a comparison will be drown. The last chapter is the conclusion, which will provide a summary of all the research. The future of the industry will be discussed briefly. In the end, the learning outcomes of the whole study will be provided.

1.3 Research Goal

The purpose of the thesis is to draw comparisons between Ethernet 802.3 ab and TSN 802.1, and how they fare against each other. As mentioned in the earlier sections the industry is currently heavily dominated by Ethernet and industrial Ethernet protocols when it comes to automation or automotive industries. Whereas, TSN is relatively a new player in this industry with long-term impact considering the power it packs. So it is only fair to estimate TSN prowess within a setup that can replicate an industrial communication system. For example, an environment where the system is handling information from multiple nodes, managing different types of information, and then forwarding the data to another node or server.

The study is based on five different tests, with each of them developed considering the dynamics of Ethernet and TSN. The whole testing is performed on hardware, whereas simulations are not part of this study. The testing setup consists of two layerscape 1028ARDB boards developed by NXP [51]. A single LS1028ARDB board packs 4 TSN capable switches that be enabled using Linux commands and Linux kernel options. The same switches can also be configured to use Ethernet. Rest of the the setup compromises an IP camera, a proprietary Wapice Oy WRM247+ device and lastly two host computers either running Windows or Linux. One of the host computers will be an endpoint for data streams coming from the IP camera and LS1028ARDB board. The other computer will be used in a few test cases to trigger ICMP packets for ping testing.

The two main research questions for this thesis topic are

1. How well can Ethernet 802.3ab (1000Base-T 1Gbps) transmit critical packets while best-effort traffic is being sent on the same link in terms of jitter and latency?

2. How well can TSN 802.1 standards transmit critical packets while best-effort is being sent on the same link in terms of jitter and latency?

3. How does an Ethernet 802.3ab connection compare with a connection running 802.1 Qbv and Qbu in terms of jitter and latency?

Throughout section 3 and 4 , Ethernet 802.3 ab will be referred as Ethernet.

2. LITERATURE REVIEW

The chapter contains information related to all the fields related to time-critical information. Starting from what makes a real-time system, to what type of communication standards and mediums are available to carry out all the information. Discussing the developments in the field of wired and wireless communication. Lastly, explanation regarding what is TSN and which standards are useful for time-critical systems.

2.1 Real-Time Systems

The main objective of real-time systems is to provide a response to an event in a given time, which means these systems require a physical clock to quantify time. The events can then be analyzed or actions can be taken accordingly. They provide a response within the required time slot to avoid any loss or miscommunication. However, these systems can further be connected to a controlling system or more advanced machinery, and it is of utmost importance that all the systems are synced to the same time scale. Mall R [1], defined a system as known to be real-time if the system's behavior is determined using a quantitative measure of time. These systems are designed to provide a specific type of response, which is different from general-purpose systems or machines like embedded systems, but this does not mean that embedded systems are also real-time.

According to Li, Q., and Yao, C [2], the relation between both is not mutually exclusive, which means regardless of them being separate entities, there is still overlapping between them. To conclude not all embedded systems are real-time and not all real-time systems are embedded systems. Real-time systems can further be divided into two different types, soft and hard real-time systems. There are no such systems by definition, however, the time constraints help differentiate between them. If the deadline to deliver a message or to perform a task has a very small or close to zero tolerance, the system is categorized as a hard real-time system. In case, if the action item is missed, the results could be very fatal as the required task is of no use. On the contrary, in a soft real-time system the required task is not invalid even if it misses the deadline, rather determines the usefulness of the performed action depending on how much time has elapsed. In short, the tolerance of a soft real-time system is not close to or absolute zero.

2.2 OSI Model

Before getting into more details regarding Ethernet and its place in the world of real-time systems, it is of utmost importance to discuss the Open System Interconnection model or more commonly known as the OSI model. OSI model is a network model that comprises seven layers. It set the tone for all the advancements in networking concepts [16]. It has helped in connecting different networks, lowered compatibility issues, and reduced the complex nature of networking devices. Each layer is independent of the other, which results in many different configurations and setups. It also means that all the layers share their responsibility.

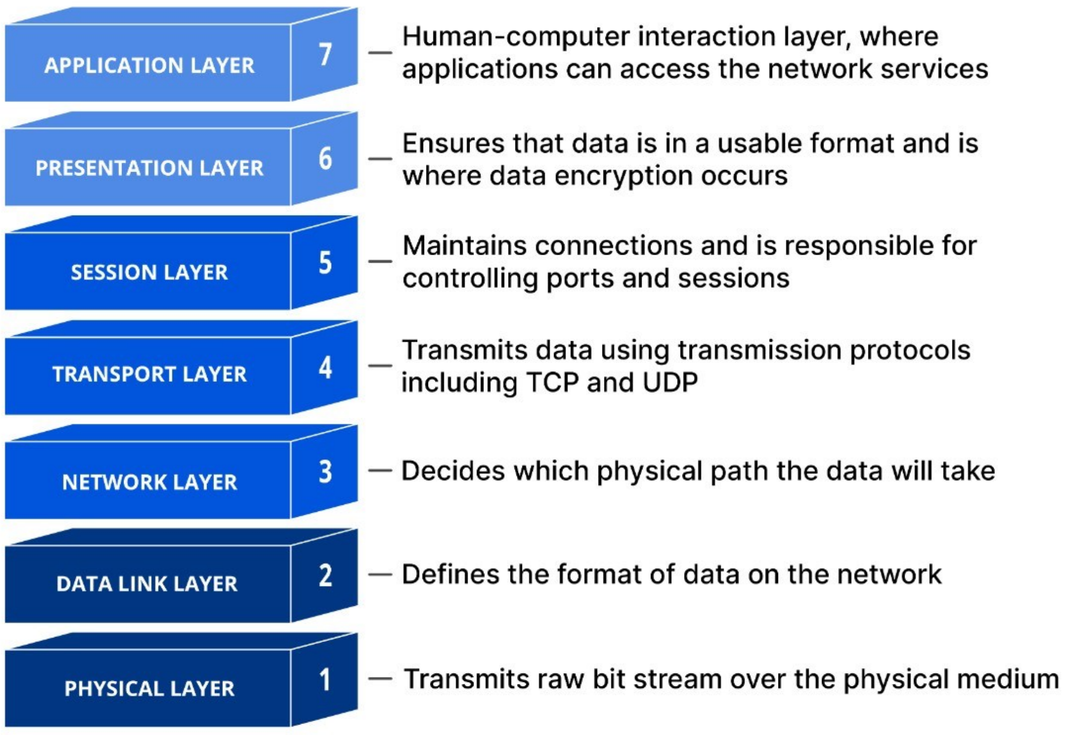

APPLICATION LAYER
7
Human-computer interaction layer, where applications can access the network services
PRESENTATION LAYER
6
Ensures that data is in a usable format and is where data encryption occurs
SESSION LAYER
5
Maintains connections and is responsible for controlling ports and sessions
TRANSPORT LAYER
4
Transmits data using transmission protocols including TCP and UDP
NETWORK LAYER
3
Decides which physical path the data will take
DATA LINK LAYER
2
Defines the format of data on the network
PHYSICAL LAYER
1
Transmits raw bit stream over the physical medium

Figure 1. *The OSI Model [17]*

The physical layer is the first layer, it revolves around wires, antennas, connectors, and other tools required to connect to endpoints. The messages are delivered using the physical layers by means of voltage difference [16]. The data is transmitted in the form of a sequence of bits. The main concerns of this layer are dealing with the type of data, the direction of communication, the integrity of both the connection and the communication, along with establishing the range of data transfer. The most common standards for physical layers are V.35, RS-449, and RJ-45 EIA/TIA RS-232.

The second layer in the architecture is the data link layer, it is the most interesting layer considering the type of research this thesis will be covering. This layer falls between the physical and network layer, so by default one of the main responsibilities of this layer is to make sure there is smooth communication between both of them. The information is passed on in the form of frames. Each of them includes crucial information which is passed on from the network layer, which consists of the addresses of the sender and receiver, as well as data related to network topologies, physical addresses, error correction, and control information [16]. Receiving of data is ensured by using a cyclic redundancy check mechanism, and retransmission is required if the codes do not match at the receiving end. However, it is also vulnerable to high-speed sending and low-speed receiving issues which should be kept in mind. High-level data link control, synchronous data link control, point-to-point, spanning tree protocol, and frame relay are well-known data link layer protocols

Network, transport, session, and presentation layers are the next layers in the OSI model, but regarding this research study, these layers will not be discussed any further. The last layer in the model is the application layer, this layer is the farthest away from the hardware, whereas it holds great importance when discussing Ethernet as an option for communication in real-time systems. The application level is more software oriented and provides means to configure or use the network, it facilitates file transfer, email data, and file management. The application layer has many different protocols such as telnet, file transfer protocol, simple network management protocol, hypertext transport protocol, and many others [16].

The communication medium used for real-time systems can either be wired or wireless, this research is mostly based on wired communication protocols, due to their popularity in the domain [5], but some light will be shed on the wireless industrial protocols.

2.3 Wireless Networking Protocols

There has already been much work done to use available Wireless industrial networking protocols, yet it faces many challenges [4]. Different stages in the automation process have different tolerance levels, which means that some of the phases are susceptible to retransmission. There are three phases monitoring, closed-loop, and interlocking control, and all of them have their timing requirements. Monitoring is not affected much by losses and delays, however, closed-loop and interlock operations are sensitive. Safety, availability, latency, and retransmission are areas where delays in communication can have a significant impact.

According to Jacob, R[3], wireless communication in real-time cyber-physical systems can be achieved by using a method called synchronous transmissions. Synchronous transmission enables meeting the necessary time deadlines to fulfill real-time constraints. Regardless of whether it is a promising solution for the wireless industrial networking protocols, there are limited resources available to the cyber-physical system engineers which hinders the use of such protocols, as the engineers are not known to be experts in the wireless domain. There is a gap to be filled when it comes to the proof of concept of these systems, as the research work and the available documentation is insufficient.

While there are protocols such as WirelessHART and ISA100.11a available to provide real-time guarantees for communication held between network interfaces [13], there is an issue with handling application schedules, as processes are considered periodic and there is no surety that message handling will be done within the time deadlines[3]. The solution to this problem lies in tackling shared resource schedules as well as using a co-scheduler to provide energy-efficient and seamless switching between nodes. However, the research work is limited as one solution is asynchronous by nature which makes it not suitable for latency-sensitive applications, and the other option is not scalable as it requires detailed scheduling information that can be taxing on the devices as huge computational power is needed to acquire the scheduling tables in the first place.

2.4 Wired Networking Protocols

The most common medium of communication in the industry is wired. There is a wide range of different protocols that are based on end-to-end connectivity using a wire. Ethernet which is still the most dominant technology used in the industry [5], has been around for a while, with yearly growth showing an expected rise in use as far as the future is concerned. Ethernet can not be used alone in the industry as it is not a real-time communication protocol, it has always been used with other technologies. This section will cover all the relevant wired protocols that are being used in the industry, and what will be their place in the near future.

Fieldbus

The research being done for this thesis study is mainly focused on the use of different Ethernet technologies in industrial automation, and the annual growth shows that the current and future most dominant industrial networking tool is industrial Ethernet [5]. However the fieldbuses still capture more than 25% of the market, and graphs show an increase in the total number of nodes connected to fieldbus. The idea of fieldbuses become famous in the late 1900s, with the advent of different medium access control protocols. The issue at the time was that Ethernet was not capable enough to provide determinism, which was essential for real-time systems, and thus for industrial automation the gap was filled by fieldbuses [6]. This also became the reason that there was not one solution, every company and solution provider went with want seemed to be the best option for them.

CAN protocol was created and released around 1986 by Bosch [7], keeping insight efficient solutions for the automotive industry. The fieldbus was designed to provide a data rate between 20 kbps and 1Mbps and was further divided into multiple standards depending on the range of speed. The protocol follows the concept of carrier sense multiple access. The bus setup allows the system to be in sync with all the nodes, due to the fact that the protocol is message based and not address-based [8]. CAN also have the ability to adapt if a faulty node is detected, and it keeps on delivering messages to fulfill real-time requirements.

As CAN was created keeping in mind the real-time messaging required in the vehicles and to limit the cabling issues that arose from the addition of micro-electronics, Profibus fieldbus was developed to be a generic solution, more focused on providing solutions in the automation industry [9]. Profibus does not use the same CSMA technique, rather it uses a token-based system, where the master issues a token that allows the nodes to use the bus for the transfer of information. The speed range of Profibus is from 9.6 kbps to 12Mbps which depends on the line length, it provides an option to connect 32 different nodes to a single bus, but repeaters can be used if the number of nodes is more than the maximum limit[10].

Modbus was introduced by Modicon in the mid-1970s in order to support their programmable logic controllers by using master-slave configurations. Later it was converted to open protocol by the same organization. The master controls the communication among the slave, and the slaves can only reply to the request or write command generated by the master. Considering the ease of implementation and its simple nature the protocol has variants from multiple companies [11]. There are three main serial Modbus types Modbus RTU, Modbus ASCII, and Modbus Plus, the difference is more towards the encoding used for each of them, which means the structure of the message is almost the same. There is also another variation called Modbus TCP, which uses a client and server technique rather than the master-slave used by Modbus serial. Modbus TCP is deployed on Ethernet[12].

Ethernet and Switched Ethernet

IEEE 802.3 standard or better known as Ethernet was first released in 1985, as a medium access control using CSMA collision detection [14]. The speed limit at that time was 10 MBps with bus topology operations. Soon after the standard was extended and new topologies as well as fiber optics cabling options were added. Later 802.3u and 802.3x improved the protocol by an increase in data rates up to 100Mbps while supporting full duplex communication on both twisted cables and fiber optics. In 1998 802.3z further enhanced the speed up to 1000Mbps, the next significant change came in 2002 with 802.3ae which pushed the data rates to 10Gbps. Currently, Ethernet technology is capable of achieving 400Gbps speeds[15].

One of the main features of Ethernet is the availability of multiple topologies settings, which are usually not available for fieldbuses[18]. Ethernet technologies such as Profinet can be established in line, star, tree, or ring topology, whereas fieldbus uses line most of the time, there is also a possibility of star or tree, which are not common. Each of them has different advantages and disadvantages, however, this study will not focus on what they are. The different networking topologies can be seen in figure 2 [19].

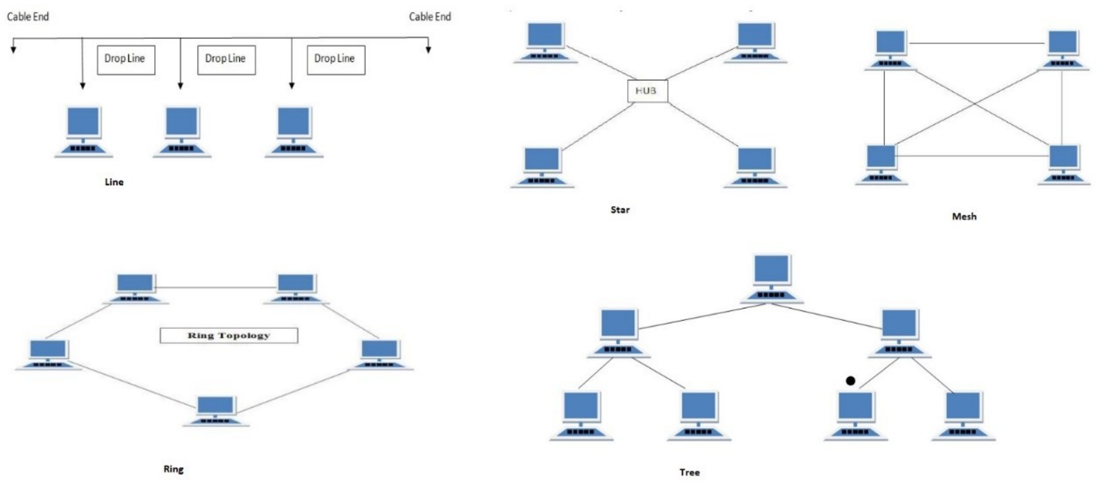


Figure 2. *Network Topologies [19]*

The switching mechanism was not part of the Ethernet when it first came, initially the hub was a repeater, which was used to broadcast the message that is received at the input. On the contrary, the hub in switched Ethernet is a device that acts upon the information it is passed on. In this manner, the hub in switched Ethernet knows exactly where to send the message, and it does not broadcast a message, rather it is a point-to-point communication. This mechanism lets the other idle devices communicate among themselves freely [24]. Switched Ethernet uses a full-duplex link, whereas the hub in Ethernet used a half-duplex link. With the addition of said features, switched Ethernet eliminated the problems caused by CSMA collision detection, as the occurrence of the collision was avoided altogether. The switch uses the store and forward technique when a packet arrives at the switch it first checks if the line is idle, if it finds the line to be idle, it sends the packet to the destination. However, if the line is busy, it stores the packets in the buffer and after the line is idle again, it sends the frames. It also manages the scenario where multiple senders want to reach the same destination, so it buffers all the frames and sends them one by one[24].

Ethernet has been widely using the IEEE 802.3 standard which uses common hardware with different plugs as well as multiple communication mediums. With performance higher than in typical fieldbuses, Ethernet works well with high-power systems [20]. Regardless of all the positives of Ethernet, the classic version of the standard is not enough to satisfy the level of determinism that is required in a real-time system. CSMA collision detection mechanism does not guarantee the transfer of data without interruption, due to the possibility of data colliding, when both ends of the network are sending the data at the same time. Queuing packets on switches, packet losses due to buffer overflow caused by multiple senders trying to reach the same destination and other delays caused by packet losses on switches do not result in deterministic information transfer.

The upper layers even worsen the case for Ethernet or the classic Ethernet. The user datagram protocol can make a case for real-time communication, given that it is capable to send independent packets, but due to no data ordering reliable communication is not possible. On the other hand, transmission control protocol helps in that manner that it can provide large data transfers, and also provides ordered delivery of data, however, the protocol is vulnerable to inconsistent transfer speeds. Issues such as error correction and overload control make transmission layer protocol a non-real-time-capable protocol [20]. This is where the concept of industrial Ethernet comes into play.

Industrial Ethernet

Considering all the shortcomings of Ethernet to deliver information in real-time, it is needless to say that the Ethernet standard 802.3 was not designed to ensure determinism [20]. As mentioned before, the Ethernet-based protocols also fail in this regard. Given the limitations in Ethernet, the OSI model layers provide enough room to adapt the Ethernet standard and make it deterministic to a level that would satisfy the real-time communication requirements. Virtual local area network standards are known to provide solutions in the field of multimedia traffic by prioritizing the flow of data, this results in reduced jitter and 10 milliseconds delivery times [21][22]. A token-based approach is also proposed [23], but a communication failure will result in retransmission from the last station to keep the system in check, otherwise, the communication fails. It uses a broadcast token message to keep the system in synchronization, which causes latency issues. As there is no single solution that makes Ethernet a reliable real-time communication medium, vendors use different mechanisms to improve the situation. The question arises which real-time industrial Ethernet is capable and to what extent, does the system has a single point of failure, how scalable is the system, can the system dynamically adapt to changing topologies, and does it need special hardware. Usually, vendors have their own answers to such questions, which leads to variation in the industrial Ethernet solution that is available.

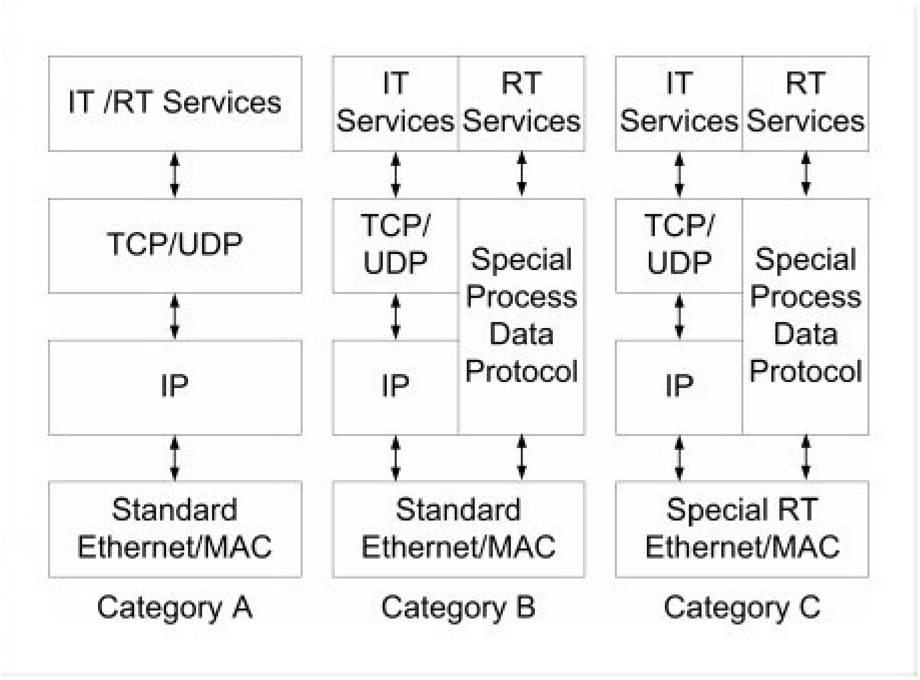


Figure 3. *Industrial Ethernet categories [27][28]*

Real-time capabilities of industrial Ethernet can be divided into three different types, it is defined by the time taken for a cycle[25][26]. The first class is also referred to as soft real-time, the time cycle is scalable and it is approximately 100 milliseconds. The second class is a hard real-time where the time cycle is between 1 and 10 milliseconds. Finally, the last class is isochronous real-time, the time cycle for this type is 250 microseconds to 1 millisecond and the jitter is less than 1 microsecond. Industrial Ethernet categorization based on software and hardware requirements for a device is shown in figure 3. Category A uses the same TCP/UDP and IP at the network layer scheme with standard

Ethernet for real-time and other information technology services. Category B uses a similar scheme at the medium access layer with standard Ethernet switches and controllers, but it uses a different protocol to send real-time information from Ethernet. The last category uses a special real-time-based Ethernet controller and switches mechanism along with a special protocol to transfer real-time information from Ethernet [20]. The class of a system relates to the time requirements whereas the category defines what changes to the Ethernet technology are required if any. Vendors have developed solutions that fall into different categories, so the next objective is to discuss how solutions differ from each other and what can they achieve.

Industrial Ethernet Technologies

Modbus-TCP was introduced for soft real-time systems, which refers to first class. It uses standard Ethernet as well as TCP/IP scheme on the network and transport layer. Such configurations make it a category A system. Modbus-TCP is implemented on the application layer, which makes it a flexible solution as it can be deployed on many communication systems. TCP port number 502 is used for the transfer of information[29], it can also be controlled using a remote setup. The devices use a client-server setup, and there is no limit to the number of devices that can be connected.

Ethernet Powerlink also follows the scheme used in Modbus-TCP for real-time communication, it relies on Ethernet switches and controllers, as it is a software-based solution if category B metrics are needed. For isochronous real-time systems [27], hardware changes are required for the Ethernet Powerlink protocol. It uses a master-slave setup, and the master is the key in this configuration as it permits the slaves to send the information in a particular cycle. The protocol uses an 8-bit addressing mechanism for the connected devices which limits the number of active connections.

EtherCAT has recently seen growth in industrial networking, with 11% of the total market share [5]. It was introduced by Beckhoff Automation and standardized under IEC 61158. Both soft and hard real-time applications can be implemented using this solution. It focuses on providing time cycles of less than 100 microseconds and jittering as low as 1 microsecond to achieve synchronization. EtherCAT functions in a manner where a telegram is generated from the EtherCAT master which is sent to all the slave devices. In fact, the single telegram is passed on to all the nodes. Every slave device in the system reads the information pertaining to it and adds its information to the packet and passes the packet to the next node. The last node in the scheme is identified when it finds an open port, by using a full-duplex mechanism the information is passed back to the master. In this protocol, the EtherCAT master is the only node that can initiate the frames, which results in predictability and real-time guarantees. The implementation is based on Ethernet, therefore master devices can be deployed on any system with Ethernet. The slave devices use a hardware-based EtherCAT slave controller removing the unpredictability of the network. It supports tree, star, line, and daisy-chain topologies, along with 65,656 devices that can be connected to a single EtherCAT segment[30].

CC link IE field is a protocol developed by Mitsubishi Electric, and it is capable of providing a real-time class 2 [20]. It is equipped with a 1 Gbps communication speed. The maximum amount of stations that can be connected is 254 and the device addresses are between 1 and 254. The scheme supports line, star, and ring topologies, as well as the distance constraint from station to station is 100m. It uses a master-slave setup as do most Industrial Ethernet networks. The flow of data, states of all in and out, and initialization of the whole system are managed by the master device. The master is limited in memory and in terms of bits the amount is 32768 or 16384 in words. The token-passing technique is used to transmit data cyclically [31]. Only the device given the token from the master can send packets, that is started by the master and then passed on to other devices. A special Mitsubishi chip or seamless message protocol changes are needed to use CC-link IE Field.

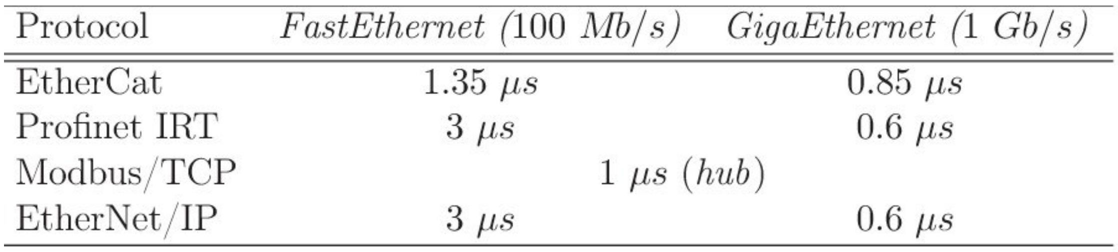
Common Industrial Protocol forms the bases of Ethernet/Industrial Protocol or Ethernet/IP, and it is placed on top of the networking and transport layer [32], which makes it a category A type of industrial Ethernet solution. It uses TCP/Internet Protocol or TCP/IP to encapsulate explicit messages and UDP/IP for input and output messages. This feature makes it easier for other devices to communicate as TCP/IP and UDP/IP are found in many applications [32]. The amount of devices that can be connected to the system while using Ethernet/IP is not given. But this is not actually the case, as in order to achieve isochronous real-time device limitations are a necessity[20], and hardware-based switches with IEEE 1588 support are also required. Multicasting and broadcasting features at routers are a must, but there are currently no standards or configurations available to implement them [20].

Profinet is one of the most used industrial automation standards [33]. Its system uses a cyclic communication mechanism, and it consists of many phases. It starts every cycle with the transmission of isochronous real-time frames, which are basically configured when the network is being installed [20]. Isochronous real-time frames are scheduled with precision on devices using synchronized time clocks, with the help of a master. The routing takes place with the help of switches, and it does not depend on medium access layer addressing rather it relies on transmission time. As a result, communication is not possible without special switches or Profinet switches in this case. This phase is followed by a real-time communication phase and the last part is for not real-time related information transfer [20].

Comparison Between Industrial Ethernet Technologies

Industrial Ethernet systems tend to follow the master-slave scheme, as well as shared memory solutions which are the controlling entity in the network. In order to achieve isochronous real-time capabilities, the master device in the system requires a token or polling technique to recognize and allocate appropriate time slots to the new devices, which results in the master device limiting the overall performance and reliability of the system. Given the limitations, it still requires expensive hardware. Industrial Ethernet has not yet reached the stage where devices can communicate autonomously without the help of a master device. All of the solutions mentioned above provide real-time communication in the form of small frames or packets and do not support large data transfers[20].

Famous industrial Ethernet systems comparisons where two different bandwidths of 100Mbps and 1Gbps were tested on a linear topology [34]. The two payloads used were 16 bytes and 100 bytes. The latencies were assumed to be as shown in Figure 4. The propagation delay was 50 nanoseconds and the distance was 10 meters between the devices. The result was measured in a minimum number of cycle times per millisecond with respect to slave devices with constant payload and Modbus-TCP was found to be the worst. It was also noted that with a small payload, the best results were achieved by EtherCAT while using FastEthernet mode. Ethernet/IP and Profinet results on 100Mbps and 1Gbps were drastically different. However, it is safe to say that all the systems were able to achieve communication with 60 slaves in less than 1 millisecond cycle time, except FastEthernet mode and Modbus-TCP[34]. The change in payload did not reflect much information.

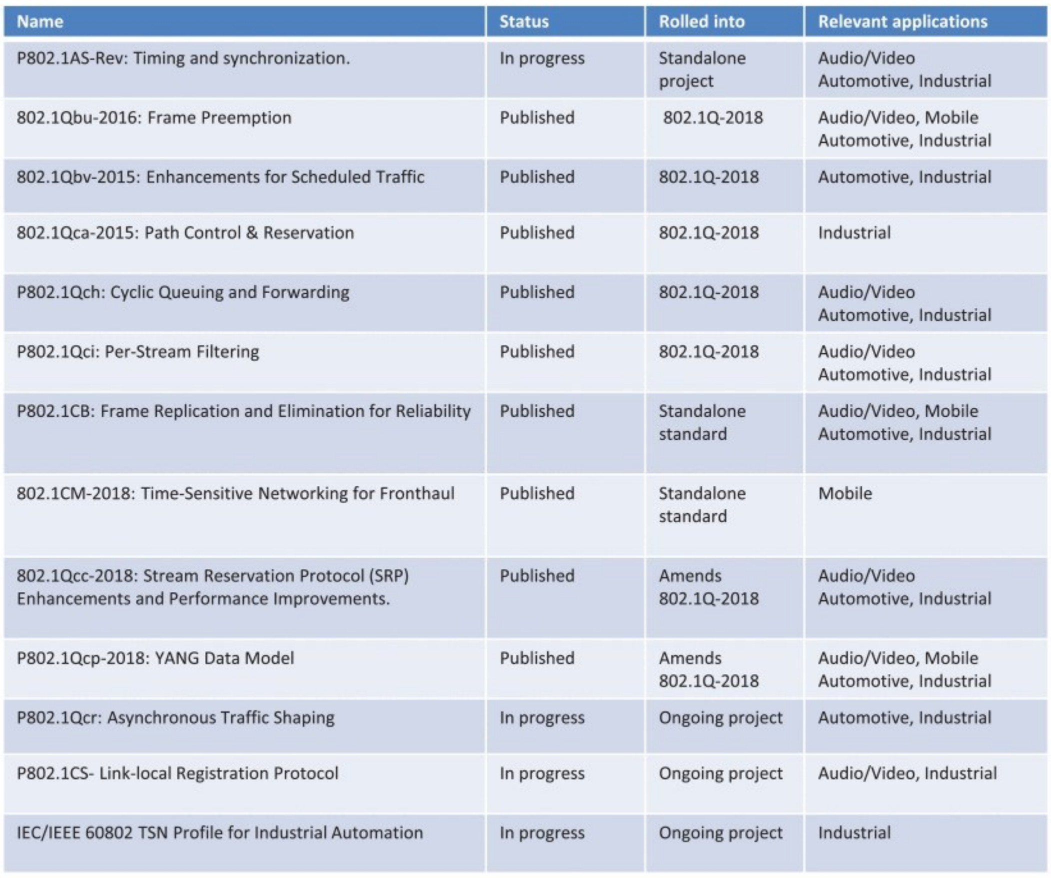


Protocol	*FastEthernet (100 Mb/s)*	*GigaEthernet (1 Gb/s)*
EtherCat	1.35 μs	0.85 μs
Profinet IRT	3 μs	0.6 μs
Modbus/TCP	1 μs (*hub*)	
EtherNet/IP	3 μs	0.6 μs

Figure 4. *Latency Comparison of Industrial Ethernet Systems[34]*

2.5 Time Sensitive Network

Time Sensitive network or TSN development is done under IEEE 802.1 task group, it contains all the work related to the standard and the extension or amendments to the original scripts. The concept of TSN is to support time-sensitive information and to add determinism to IEEE 802 technologies, which involves the transfer of information within the latency requirements. TSN targets time synchronization, high reliability, low latency, and resource management. All these objectives are carried out with different standards, and most of them fulfill more than one purpose [35]. The standards will be discussed in more detail in this section.



Name	Status	Rolled into	Relevant applications
P802.1AS-Rev: Timing and synchronization.	In progress	Standalone project	Audio/Video Automotive, Industrial
802.1Qbu-2016: Frame Preemption	Published	802.1Q-2018	Audio/Video, Mobile Automotive, Industrial
802.1Qbv-2015: Enhancements for Scheduled Traffic	Published	802.1Q-2018	Automotive, Industrial
802.1Qca-2015: Path Control & Reservation	Published	802.1Q-2018	Industrial
P802.1Qch: Cyclic Queuing and Forwarding	Published	802.1Q-2018	Audio/Video Automotive, Industrial
P802.1Qci: Per-Stream Filtering	Published	802.1Q-2018	Audio/Video Automotive, Industrial
P802.1CB: Frame Replication and Elimination for Reliability	Published	Standalone standard	Audio/Video, Mobile Automotive, Industrial
802.1CM-2018: Time-Sensitive Networking for Fronthaul	Published	Standalone standard	Mobile
802.1Qcc-2018: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements.	Published	Amends 802.1Q-2018	Audio/Video Automotive, Industrial
P802.1Qcp-2018: YANG Data Model	Published	Amends 802.1Q-2018	Audio/Video, Mobile Automotive, Industrial
P802.1Qcr: Asynchronous Traffic Shaping	In progress	Ongoing project	Automotive, Industrial
P802.1CS- Link-local Registration Protocol	In progress	Ongoing project	Audio/Video, Industrial
IEC/IEEE 60802 TSN Profile for Industrial Automation	In progress	Ongoing project	Industrial

Figure 5. *TSN Standards [35]*

With the time synchronization feature all the host machines and nodes on the network are able to synchronize the clocks as accurately as 1 to 10 microseconds. Precision Time Protocol can help achieve this goal, which is based on IEEE 1588. Losses due to buffer overflow at the nodes are one of the most common causes of packet drops. TSN deals with network congestion, by delivering information at such a speed that eliminates the possibility of losses. TSN flows also guarantee a worst-case scenario latency for the transfer of information [36]. A duplication mechanism is used by TSN to add reliability to the communication. Duplication of sequenced data packets is done at the origin and then sent through multiple paths. When the packets reach their destination the copies of the packets are then discarded. TSN has many use cases such as automation, audio and video streaming, automotive applications, and many others [36].

Time Synchronization

Time synchronization is of vital importance when considering professional audio and video setups, and automotive or time-sensitive automation. As the risk of losing communication links is fatal in such cases, the system should still be active in case the grandmaster or link is lost [37]. The two most interesting standards are 802.1AS-2011 and 802.1AS-Rev both published by IEEE. The first standard is one of the 802.1 Audio Video Bridging standards, which deals with audio and video. It also contains a Precision Time Protocol profile, that adds sharing of time-related information such as synchronized time, selection of source, phase, and frequency irregularity notifications. With the help of IEEE 802.1AS-2011-time synchronization between the nodes is precise. While having a time reference accuracy of less than 1 microsecond, the recovery mechanism takes time and causes time jumps when an active grandmaster is lost. Given the importance of this protocol, it will be discussed in more detail in this section.

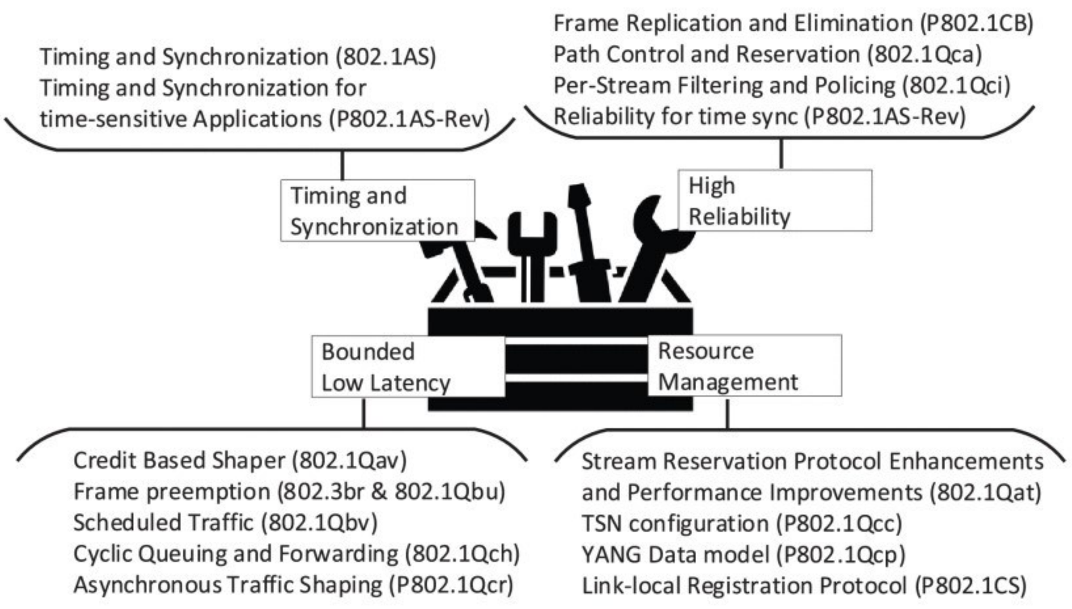

Timing and Synchronization (802.1AS)
Timing and Synchronization for time-sensitive Applications (P802.1AS-Rev)
Frame Replication and Elimination (P802.1CB)
Path Control and Reservation (802.1Qca)
Per-Stream Filtering and Policing (802.1Qci)
Reliability for time sync (P802.1AS-Rev)
Timing and Synchronization
High Reliability
Bounded Low Latency
Resource Management
Credit Based Shaper (802.1Qav)
Frame preemption (802.3br & 802.1Qbu)
Scheduled Traffic (802.1Qbv)
Cyclic Queuing and Forwarding (802.1Qch)
Asynchronous Traffic Shaping (P802.1Qcr)
Stream Reservation Protocol Enhancements and Performance Improvements (802.1Qat)
TSN configuration (P802.1Qcc)
YANG Data model (P802.1Qcp)
Link-local Registration Protocol (P802.1CS)

Figure 6. *Key Features of TSN [35]*

Low Latency

AVB and TSN have improved conventional Ethernet networks by adding determinism and guaranteed message delivery functionality. AVB includes the standard which is related to queuing and forwarding messages to ensure the timely transfer of information. Forwarding and Queuing Enhancement for time-sensitive streams the IEEE 802.1 Qav standard provides the real-time guarantee even if the network stations are not globally in synchronization. IEEE 802.1Qbv and Qbu are TSN standards that further increase its ability to transfer real-time data. With the schedule-based communication approach of 802.1 Qbv, the necessity to synchronize and the time needed to decide which packet to send next is saved [35]. On the other hand, 802.1 Qbu uses a pre-emption-based technique, which prioritizes real-time communication over non-time critical information. These two standards are the most relevant standards related to this thesis study, so these will be discussed further in this section.

Reliability

With all the similarities between AVB and TSN, the high reliability of packet transmission sets them apart. These applications are usually very sensitive in nature and loss of communication results in disasters, for example, automotive vehicles. The flow of communication between the nodes within the network cannot be exposed at any cost, and error detection and reporting are equally important. A delay or loss of any information may cause an accident or even worse. For this reason, multiple TSN standards are published, with each of the standards bringing different attributes. Duplication of information, detecting irregularities in stream flows, reserving bandwidth, and path control [35]. Frame Replication and Elimination for reliability standard IEEE 802.1CB uses the duplication technique, to send copies of the same data using different paths. The duplication is then removed before or at reaching the destination. This helps the data to reach the destination with less chance of loss of information. The protocol can send data with sequence numbers as well as add identities to different streams. Path control and reservation standard also known as IEEE 802.1Qca is another protocol that adds reliability to the communication system. It is basically an extended version of an intermediate-system-to-intermediate-system, that can provide forwarding paths with explicit controls. Other than that, it supports bandwidth reservation features. Another TSN standard is Per-Stream Filtering and Policing, the 802.1Qci. Stream misbehavior can be detected using this protocol, and further actions can then be taken accordingly [35]. These protocols are not of interest considering this thesis, so we will leave the discussion here.

Resource Management

Determinism in multiple TSN standards is achieved by resource management and bandwidth or resource reservation. IEEE 802.1 Qat Stream reservation protocol uses a mechanism where it limits the TSN flows by accounting for maximum packet size as well as transmitted packets for a given interval. By using this technique, the network resource is made available for all types of traffic. Queues, buffers, and other networking resources can be reserved or registered. This means the quality of service can be managed between endpoints, which enables real-time guarantees for latency and bandwidth. 802.1Qcc stream reservation protocol enhancement and performance improvement standard which is an extension of 802.1Qat targeting the industrial and professional market by adding stream configuration support. The last standard under discussion for resource management is IEEE 802.1CS link local registration protocol which is an improvement on the 802.1Q multiple registration protocol. These protocols make copies of databases in order to use the whole network for the distribution of information. However, 802.1Q tends to slow down the system when the size increases from 1500 bytes databases whereas 802.1CS can handle databases of size 1 megabyte which is almost 700 times more. These protocols are also not important given the scope of this research study.

2.6 IEEE 802.1 AS Clock Synchronization

The grandmaster is the reference time or master clock for all the other nodes and devices in the system. The selection of the grandmaster is done using an algorithm, the best master algorithm is capable of selecting a device as grandmaster device among many possible options. The next step is to allocate every node its specific states, which can be passive, slave, or master. This is achieved by the spanning tree synchronization mechanism within the network. If any of the devices were to be not used or disabled at any point, the state of the device can be set to disable. A device can be a master and a slave to another device, if the device is neither of them it is connected as passive. Moreover, these devices have end-to-end connections with other devices, so if one port is master the other device connected to the same port should be a slave and vice versa.

With the AS revision addition, the protocol can configure grandmaster as well as spanning tree synchronization multiple times to make the system more fault tolerant. Another feature was added, where the spanning tree synchronization can be established without using the best master algorithm [35]. Once the spanning tree synchronization is done, the next step is to periodically send fsync starting from the grandmaster. The fsync message is sent to all the devices where the grandmaster's port state is master. When this message is received at the slave device, it notes the timestamp and initiates the fysnc to all the ports where this device state was set to master, the same steps are repeated until the spanning tree hierarchy reaches the end node. Then another message is generated by the grandmaster which is ffulp. The message is passed on to the spanning tree in the same manner as the fsync frame. Ffulp contains the origin timestamp when the first fsync message was generated by the grandmaster. The devices can synchronize time by keeping track of residence time, which is the device's fsync transmission timestamp minus the receiving timestamp. Furthermore, the standard adds two additional features, the first one is to keep track of latencies of the network link and the second is to calculate the clock drifts among the neighbors to normalize the value of time used in the ffulp message.

The performance analysis of the standard depends heavily on its ability to synchronize the clock with the system. This starts to become an issue once industrial applications are considered on a large scale. Simulations have shown that physical parameters can affect the system’s time synchronization abilities [38]. The protocol is also susceptible to a security attack, which hinders its performance as a result. Although most of the attacks can be prevented using encryption cybersecurity techniques, there is still a threat when the attacker uses a delay attack. By using this mechanism, the forwarding of fsync and ffulp through the spanning tree is compromised [35].

Algorithms used for clock synchronization are passed through different assessments, to make sure that they are tolerant to the device and connection failures. These can be either software or hardware. The standard does not define how to establish a system by assuming a given failure is tolerated, as well as it does not provide a mechanism for whole system reset or local clock synchronizations in startup phases. Nevertheless, protocols such as SAE AS6802 known to be failure tolerant can be used on top of IEEE 802.1Asrev.

2.7 IEEE 802.1 Qbv Schedule-Driven Communication

In time-triggered communication, the information is transferred in a scheduled format, with the help of synchronized clocks. Communication in Time triggered is done by estimation of a schedule that provides the information to the network nodes when to send, and what to send. This is communicated to the devices in the network during the configuration phase, whereas the scheduling function handles the execution of the schedule in the devices [35]. There are many standards and protocols based on time-triggered communication such as Profinet, Time-Triggered Ethernet, Flexible Time-Triggered Ethernet, Time-Triggered Protocol, and FlexRay. 802.1Qbv is the time-triggered standard for TSN [35].

802.1Qbv does not exactly operate as time triggered when it comes to scheduling of frames, rather it schedules the deactivation and activation of the queues. These queues can be transmitted queues related to end stations or traffic class queues for bridges. This is made possible by using a gate for each queue to manage the deactivation and activation process. Activation of the queue requires the gate to be in the open state and similarly, the gate is to be in the closed state in order to deactivate the queue. The transfer process selects the frames that are from active queues at the bridge ports, and the same process is followed at the end stations as well. However, the frame transmission at the end stations is assumed to be scheduled, as the end stations' communication is not available to the system [35].

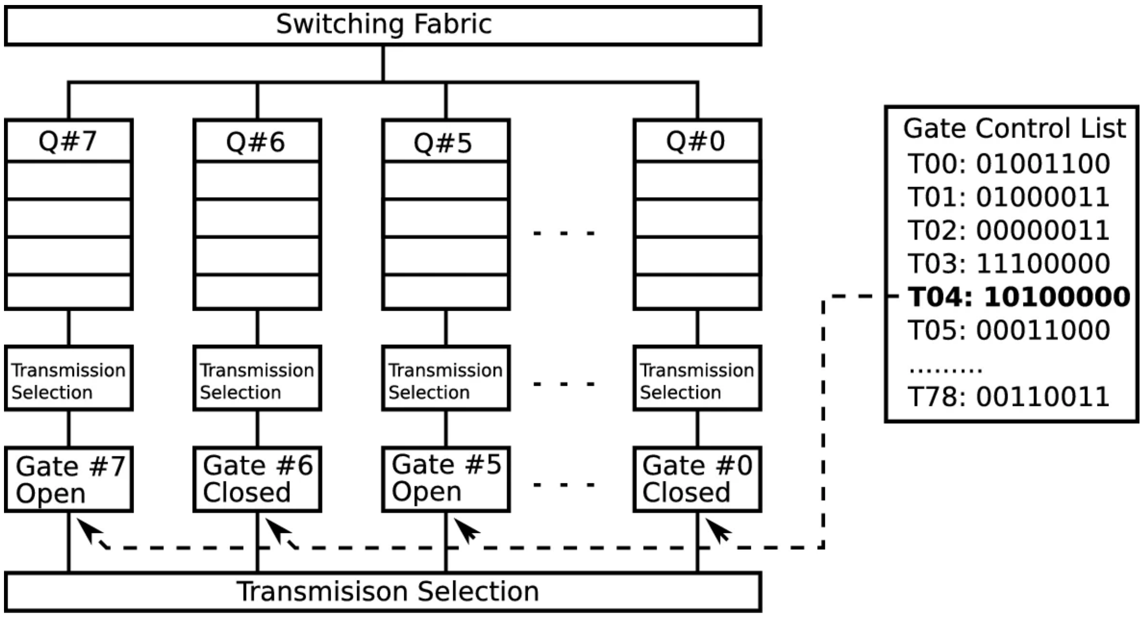


Figure 7. *Gate Control List and Queue [44]*

The main entity in this mechanism is the control list known as the gate control list, which holds all the data regarding states, classes, ports, and the time at which to send the information. Any entry in the list has a time value that indicates when to send the frames, a parameter to set the initiating bridge, the traffic queue that will be used, and finally the receiving end station. The most critical point is that the closing of the gate for a given queue is done way before the next traffic queue gates are opened with the help of a guardband. This makes sure that transmission of information from the early queue is completed before the gate is closed, as well as guarantees that the next traffic queue frames can be sent as soon as the gate opens as there will be no delay from earlier communication [35]. Long-duration wait or protected transmission window can be implemented by waiting for the time it takes to send the maximum-sized message for a given queue, which means the wait time is always using the worst-case scenario. The other approach for this can be to dynamically adjust the time depending on the size of the frames. The hold and release feature of 802.1Qbv can be used when frame preemption is needed.

When there are synchronized as well as not synchronized end stations in the system. In this case, if both the end stations have data to send at the same time, the synchronized end station will send the information, which is correct. However, if the unsynchronized end stations have queued the data slightly earlier than the synchronized one, the synchronized one will have to wait. As a result, time-critical information is lost due to unsynchronized data. This type of issue is solved using the Qbv standard, where the gate control list can allocate class priorities to different traffic queues.

Although the mechanism of the standard is easy to understand, making a gate control list on the other hand is a challenging task. There are different approaches when it comes to scheduling communication, by using specialized search algorithms or tools such as integer linear programming that are considered general purpose. It is a field of research, and some of the algorithms can be used for 802.1Qbv.

2.8 IEEE 802.1 Qbu Frame Preemption

This standard provides a mechanism to suspend the transmission of less critical frames in order to send more critical frames for bridges and end stations in a network. It also has the functionality to resume the less critical frame transmission once the more critical data is sent. This standard is linked with the Specification and management parameter for interspersing express traffic standard 802.3br. 802.3br has two different medium access control interfaces, pMAC and eMAC [35]. However, there is another option where both preemptable and express media access control can be attached to a single physical layer, this is also called the merge sublayer. It has the ability to provide a frame preemption feature as required in 802.1Qbu. This merge sublayer can block the preemptable data over the express traffic in two different ways, either by interruption or by completely stopping the transmission of preemptable traffic [35].

The interruption method has two conditions, the first one is that at least 60 octets of the initiated frame are sent. The other is that there are a minimum of 64 octets left which are yet to be sent, this also includes the cyclic redundancy check part of the frame. The choice can be 0, 64, 128, or 192 but 64 is the common packet size. The small packet of the frame each carries a cyclic redundancy check specific to the packet in order to rearrange the frame at the receiver, except the last packet which holds the 4 octet value of media access control that was preempted [35].

The second method where preemptable traffic can completely be stopped is achieved by a primitive interface introduced by 802.3br, known as the media access control service interface. The primitive can have either hold or release, the preemptable traffic is held at the merge sublayer, or the preemptable traffic is allowed when there is no express traffic respectively [35]. The hold and release feature helps in reduced latency for express traffic. With the conditions mentioned above, there are only two cases, one where preemptable traffic is allowed and the other where it is blocked. In the first scenario, if express and preemptable traffic both have a packet lined up to be sent, and there is no ongoing transmission then express traffic is sent. However, in the other case if preemptable traffic is allowed and it is currently being transmitted then the express traffic will have to wait for the completion of the transmission of preemptable traffic before it can start transferring frames. This is another additional constraint on preemptable traffic, that should be agreed upon on both ends, otherwise, it is not possible.

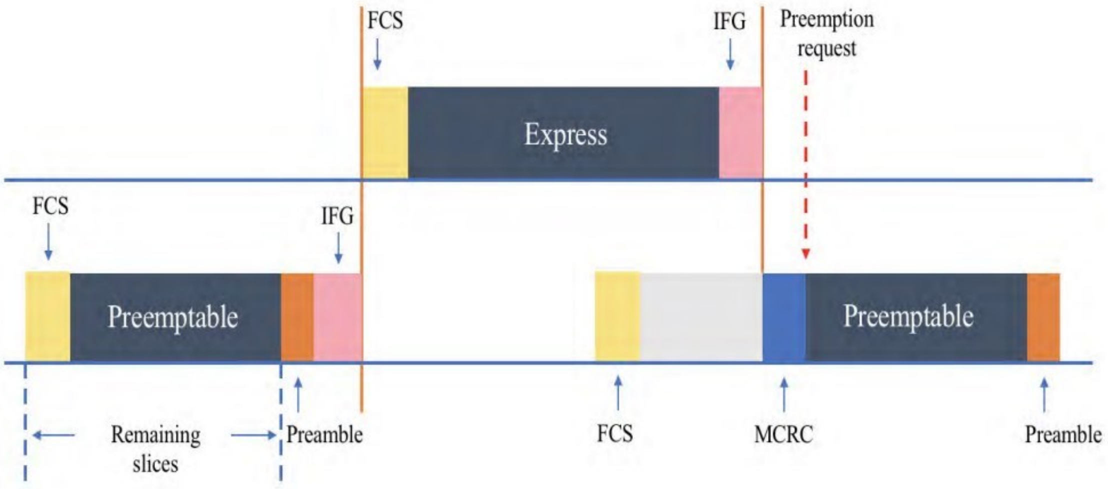


Figure 8. *Frame Preemption simplified view [45]*

802.3br introduces the frame preemption features in the 802.1Q TSN standards. The traffic queue of any port in any end station or bridge is allocated a status value regarding its preemption. This value can either be express or preemptable, and it is stored in a table within the same bridge or end station [35]. eMAC transmits expressed traffic and pMAC sends preemptable traffic. The preemption status of all similar traffic queues has equal priority. Change to the status value of preemption is possible, but it can only be done by the management.

Frame preemption helps transmit the information which is critical for the operation of the system, as it hinders the non-sensitive instead. The advantage of sending express data over preemptable has great value considering a low-speed system, whereas this is significantly less important considering a high-speed network [35]. At times express traffic from different traffic queues can cause latency in the transmission of express traffic. For this reason, scheduling along with preemption seems to be the best solution [35]. The preemptable traffic is usually sent on a gate that is open always, in this manner, the delay between for next express traffic frame is almost 124 octets. If the given time delay is still not enough for the system, the Qbv standard additionally contains set and hold media access control or set and release media access control gates. These gates are scheduling information considering the synchronized time of opening and closing of gates. By scheduling set and hold media access control gate 124 octets before express traffic gates open can fix the issue, and there will be no delay in the transmission of time-sensitive information.

3. EVALUATION PROCESS

The purpose of this chapter is to establish a testing setup that will be used to evaluate the practical limitations of classic Ethernet and how TSN compares with it. A combination of electrical and networking components will be selected and reasoning will be provided for the selection of the specific device. Moreover, this chapter will explain the testing environment that will be run once for Ethernet and then for different TSN standards.

3.1 Evaluation Setup

The network defined in Figure 9 consists of two NXP LS1028ARDB boards, one internet protocol camera, a server or computer, and most importantly a WRM 247+ device which is a proprietary Wapice Oy product. The network is connected in such a way that one LS1028ARDB board is connected with devices that are generating information. One is an IP camera that generates video stream which is a typical industrial as well as network device. Then there is a WRM 247+ board that collects information from the onboard or external sensors or actuators and forwards this information to a specific HTTP server, which also gives the impression of a device generally used in an industrial environment.

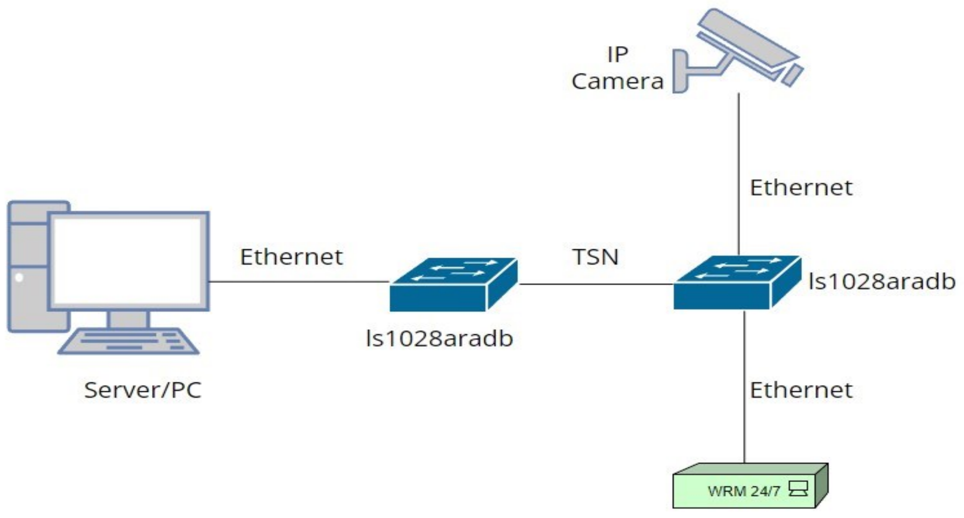


Figure 9. *The setup used to compare Ethernet and TSN*

This the second LS1028ARDB board is connected to another LS1028ARDB board or the first board and the communication between them can be varied between Ethernet or TSN, better explanation will be given when the configuration and features of the components will be discussed later in this chapter. The first board is then connected to a server or computer, this again is a pretty common configuration considering the network settings used in industry.

3.2 Network Components

The network components used in this setup have great significance considering Wapice Oy's industrial's interests. As it can be seen that only a few devices are connected in the whole network, which is atypical of an industrial network where there are 100s of nodes. Creating a network equivalent to or similar to those in the industry is not a viable option for this thesis topic. Also, it is a comparison-based thesis, so a small setup has many advantages such as easier debugging, packet sniffing, and configuration changes. So the components were selected after discussing things over great length especially the choice for a device that generates or transmits information periodically, and finally the WRM 247+ a Wapice Oy based device was selected.

3.3 LS1028ARDB Board

This is by far the most important component of the network. The setup consists of two of these boards. Both of them serve different purposes. One is connected to the server and the other is used as a connection to actuators and sensors. These boards are developed by NXP semiconductors. They target industry 4.0 as well as provide TSN to enable deterministic and effective flow of industrial and information traffic. It is equipped with high tolerance components with support up to 125 degrees of junction temperature and a 15 years guarantee which sits well with the industrial requirements.

The board provides four RJ45 switches which also support TSN. This feature is the most interesting for the research at hand. These switches are configurable through software. LS1028ARDB can be configured with proprietary commands available in the layerscape software development kit embedded in Linux operating systems via the terminal which is accessible through the UART port. The board also provides different Linux builds as well as custom builds to configure your board accordingly. These custom or prebuilds can be installed on a USB or SD card and then can be connected to USB A type connector.

The board has many other features but it is out of the scope of this thesis so next the configuration of the board will be discussed. To operate the board as shown in Figure 9, the latest firmware update was required. It was done by downloading the latest firmware available on the next website, then the firmware is burned or installed on the SD card using the DD tool and flex-build library. The next step was to customize the Linux kernel and Ubuntu mainland build to support TSN features which are not supported on the default kernel and Linux build.

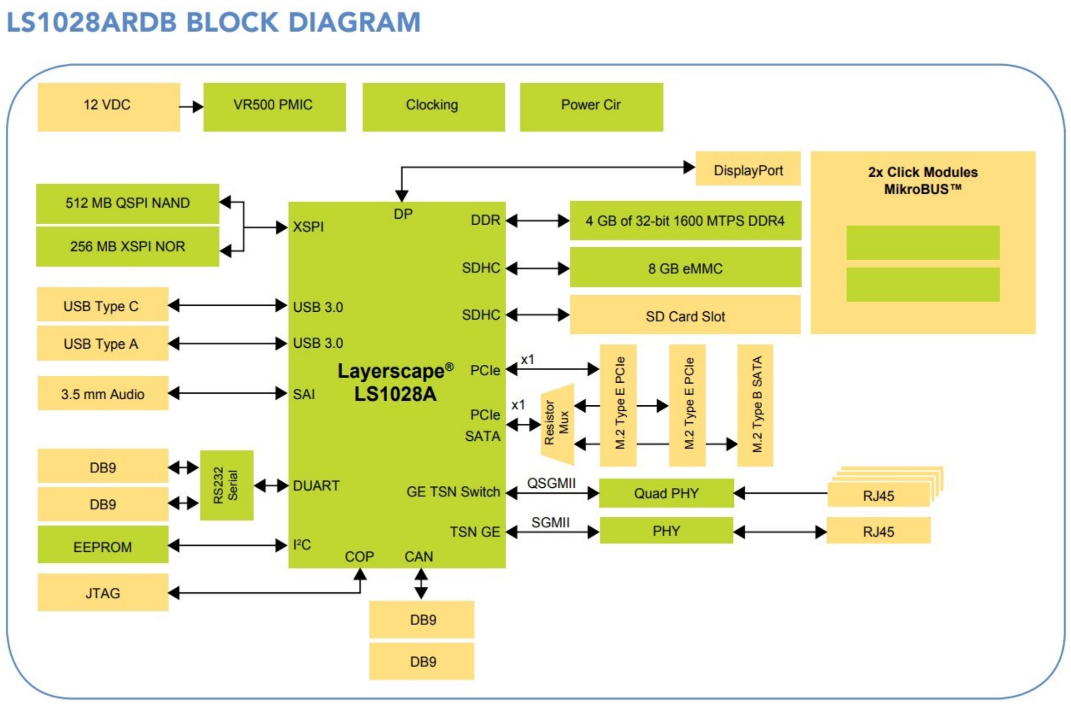


Figure 10. *Block Diagram of LS1028ARDB board [40]*

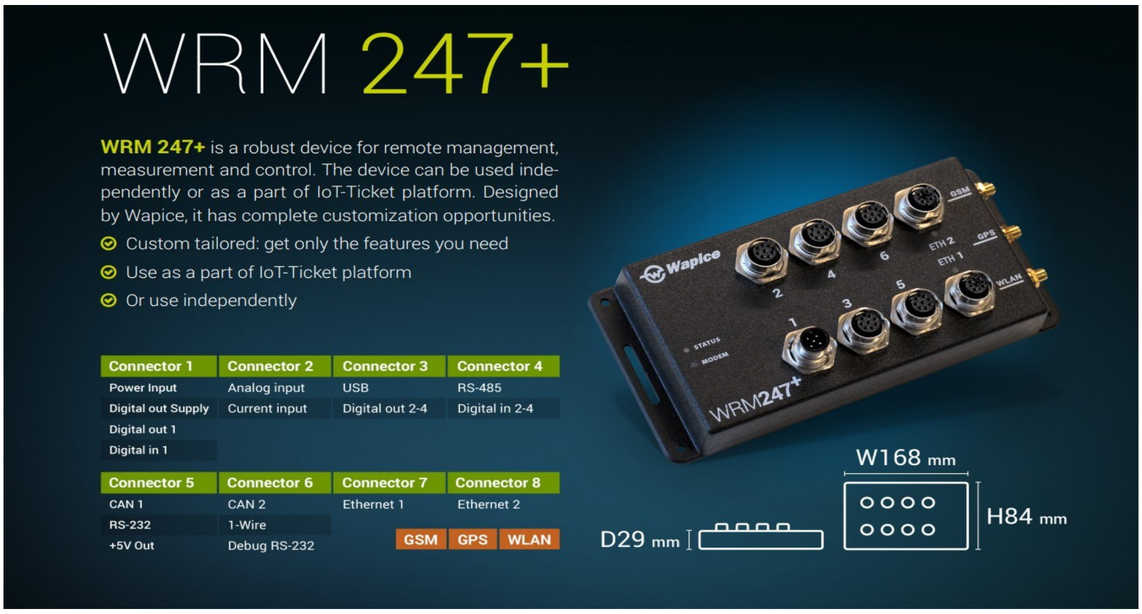
After these setting the device is booted using these builds and the tsntool library is now packed in the Ubuntu mainland, along with TSN felix switch settings enabled in the Linux kernel. To perform all these configurations layerscape software development kit user guide is used.

3.4 IP Camera

The choice of camera was the easiest decision to make, as there were several cameras that could have been selected. So, it was decided to use one of the cameras that were not difficult to set up. A Dahua IP camera was selected, which provides a web interface with secure login and H.265 coding which provides high-definition video output as well as high compression and low bit rate [41]. It can be powered by a direct current 12V constant power and it can be connected to the LS1028ARDB board using an Ethernet cable. Along with all the features it can operate from -30 degrees to 50 degrees Celsius and is graded Ingress Protection 67 which makes it a suitable camera for domestic or industrial use. The camera uses HTTP, TCP, or UDP communication protocols to transfer information and it can use either IPv4 or IPv6.

3.5 WRM 247+

WRM 247+ is a Wapice proprietary device that can be configured for different industrial purposes. It has an ARM cortex A5 process with 256MB of random access memory and flash memory. It runs real-time Linux with wired and wireless connectivity. It comes with two Ethernet and CAN ports. Cellular, wireless local area networks and position chips such as Global Positioning System and GLONASS are all available. It can be used standalone without any other Wapice Oy products, as the customization is flexible. It provides management, control, and measurement services. Other than the built-in position chip, accelerometer, and temperature sensor, different analog and digital inputs can be connected.


WRM 247+

WRM 247+ is a robust device for remote management, measurement and control. The device can be used independently or as a part of IoT-Ticket platform. Designed by Wapice, it has complete customization opportunities.

- ✓ Custom tailored: get only the features you need
- ✓ Use as a part of IoT-Ticket platform
- ✓ Or use independently

Connector 1	Connector 2	Connector 3	Connector 4
Power Input	Analog input	USB	RS-485
Digital out Supply	Current input	Digital out 2-4	Digital in 2-4
Digital out 1			
Digital in 1			

Connector 5	Connector 6	Connector 7	Connector 8
CAN 1	CAN 2	Ethernet 1	Ethernet 2
RS-232	1-Wire		
+5V Out	Debug RS-232		

GSM GPS WLAN

Figure 11. Wapice Oy WRM 247+ Device [42]

The selection of this device has many advantages considering it has many use cases and most importantly it is a Wapice Oy device. It can collect data from the sensors or actuators and then forward it to the HTTP server using any of the wired or wireless connections available. For this thesis, an Ethernet connection will be used which will directly be connected to the second LS1028ARDB board. Moreover, the board has an operating temperature of -30 to 60 degree Celsius with an ingress protection rated as class 65 making it appropriate for industrial use.

3.6 Server or Computer

This component of the setup was added in the end, it works as an endpoint to the whole system. It can be seen as a cloud device or server or any other storage device. For this same reason, the requirements for it were not strict. In the network shown in Figure 9, a simple Windows machine is being used. The machine is running Windows 10, and there were no special reasons to select it, it is just for convenience. Also, Windows gives an advantage that the whole thing can be moved on to Linux, where Linux would be a virtual machine running on a Windows client. During the whole process, this server or machine will be connected using the Ethernet connector.

The machine not only served as an endpoint or storage device, rather it was used to display the video stream coming from the IP camera. The WRM 247+ device was also forced to send the data to the IP of this machine. In this manner, the endpoint for all the data in the system was this machine, so a simple packet sniffing tool was a great way to visualize how different configurations affected the system. The IP camera stream was available through a web interface. Whereas, an HTTP server was created to receive HTTP packets sent from the WRM 247+ device.

3.7 Linux Tools

There were several Linux tools used during the whole implementation of the thesis, but there are a few that need to be discussed considering the research topic. Therefore, a brief introduction to the tools and their usage is necessary. To keep the list small, redundant tools such as installing, deleting, text editors, searching, directory lists, permissions, partitioning, and other such tools are not included.

Ethtool is a complicated tool, which options to configure network to a great extent. However, the use was limited for the research topic. The command was used to check the traffic class settings. Ethtool was used to view statistical values of packet that were transmitted and received on different interfaces of switches on the LS1028ARDB boards on almost all the test cases.

Iperf is a Linux tool that is used to test TCP or UDP connection and speed limits over a wired or wireless interface. There are multiple packages against iperf, the latest being iperf 3. However, the most common commands and utility used for testing is from iperf 2. For the research under discussion, it does not matter which package is being used. Using it is relatively simple, it requires at least two nodes or computers. One is made server whereas the other is the client, and once the server is up and running the client just needs to connect using the terminal command. The most impressive feature of iperf is that it works as best-effort traffic and it tries to send as many packets as it can, which makes it a great tool to test TSN tools Table 1 below shows the commands that are required for this thesis.

Command Options	Usage
Iperf -s	Creates a server on the system, uses default port 5001, and uses TCP communication
iperf -s –p	Creates an iperf server on the specific port, but the same should be used on the client side as well
Iperf -s -u	Creates a server that uses UDP instead of TCP
Iperf -c (server IP)	Creates an iperf client and tries to connect to the server IP mentioned in the command

Table 1. *Iperf 2.0 commands [46]*

SSH is one of the most powerful and useful tools when it comes to communicating over an insecure channel to make sure that the transfer of information is protected using encryption and keys. It can be used with the help of clients such as Putty or OpenSSH, however, for this research, the Windows command prompt was used [47]. SSH was used to configure the WRM247+ device, as it runs an embedded Linux operating system, which implies that there is no display or graphics interface. To counter this issue a Windows device was used to connect to the Linux system using the SSH commands on the Windows command prompt. The whole network is a closed network with no connectivity to the other world, and due to the simplicity of the setup to compare the OSI model layer 2 technologies a simple build was used for WRM247+. This build does not require a secure connection or exchange of keys as a normal SSH connection would require. However, the SSH application is still running on Linux, so an insecure link to the device was still possible. The root@IP (IP address of WRM247+ device) command would suffice in this case.

There is a Linux kernel feature known as traffic control, which has the power to organize the transfer of information in the form of different priorities. The mean feature of TC is to shape data in such a way that the traffic is controlled for example by limiting the data rate, it can also add policies which will mean applying checks on the type of packets or the network features such as ports or IP addresses. TC can also control the scheduling and dropping of packets to use the bandwidth accordingly. TC rules are applied based on ingress or egress, in other words, they are related to incoming or outgoing packets respectively. The TC tool is a complex tool and comes with a number of features, where many of which are outside the scope of this research. For this reason, only the commands that are related will be discussed. The three main objects in TC are filter, qdisc, and class, for this topic only qdisc and filter are useful. Qdisc is basically a queueing scheme, it is set on a network interface. When the Linux kernel sends the packet to the network interface it is the first queue on the qdisc set for the specific interface.

Object and Options	Usage
tc qdisc add/change/replace/link/delete	Type of operation to be performed on the qdisc
dev (interface)	Takes the interface name
Tc filter add/change/delete/replace/get	An operation to be performed on the filter
protocol (protocol)	Protocol number, e.g ICMP is 1
prio	Priority number
Filter type	e.g u32, route, flow
Traffic type	egress or ingress
Action	Skbedit(only one used) to set the priority number for the type of data

Table 2. *TC commands [48]*

As for the filters, they are linked with the qdisc and cannot be implemented separately. Filters object has multiple options where it can further segregate the data for example route, flow, control group, syntax-based check, and other options. TC commands used for the research purpose are as mentioned in Table 2

Tsntool is a tool used to set the TSN features in the LS1028ARDB board. It can be installed separately using the flex-builder tool or a root image can be downloaded from the NXP servers which already contain the tsntool and related packages. There is not much documentation available for it, which is understandable given that the command is simple and easy to use. The TSN-capable switches on the LS1028ARDB can be configured to use IEEE 802.1 Qbv or IEEE 802.1 Qbu standards. The commands for this tool are mentioned in Table 3.

Command	Usage
tsntool	To enter tsntool configuration mode in the Linux terminal
qbvset --device (interface) --entryfile (xyz.txt)	This command sets IEEE 802.1 Qbv values. The first argument takes the interface on which the TSN standard is to be activated. The second argument sets the gate control list, which governs the flow of different priority queues on the set times. Contents of an entryfile can be as shown t1 11111110b 5000 t1 is just the name of the entry, the second parameter defines which traffic classes are allowed, whereas the third value is the duration allowed in nanoseconds
qbuset --device (interface) --preemptable (value)	To set IEEE 802.1 Qbu values this command is used. Preemptable parameter set what class of traffic is preemptable and which class is express. The value of preemptable can be e.g. 0xfd where each value corresponds to a traffic class combination.

Table 3. *Tsntool Commands [43]*

3.8 Network Configuration

After the selection of the components, the next step was to apply the configurations for each of them. As mentioned in 3.3 the default firmware and Linux build were not suitable to run TSN features. So, the following steps were performed to get the LS1028ARDB in working condition

1. Downloading and installing the latest firmware from the NXP website
2. Installing the flex-builder tool provided by NXP on a host Linux machine from NXP website
3. Downloading Linux boot and root files, along with different version images like Ubuntu mainland or Lite
4. Editing Linux kernel using the flex-installer to include TSN feature config options

5. Creating Linux boot and root images using the flex-builder
6. Installing the firmware, boot, and root images on the USB stick using the flex-builder tool
7. Connecting the USB device to the LS1028ARDB board and booting the system from the USB drive

After performing all the above-mentioned steps the LS1028ARDB boards are now using the customized kernel and a Ubuntu mainland 20.04.2 build. This build provides the tsn-tool library which was used to test the different standards, the changes and the reasoning behind them will be discussed in detail in the next chapter.

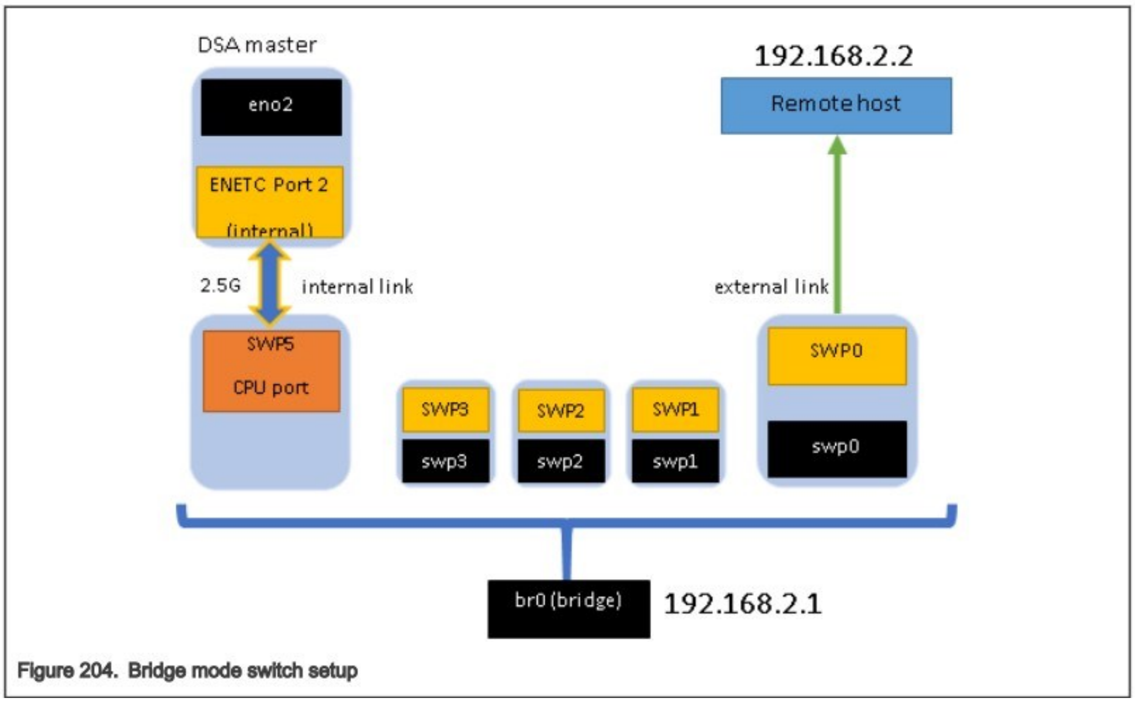


Figure 12. *Visualization of Ethernet switches when in bridge mode[43]*

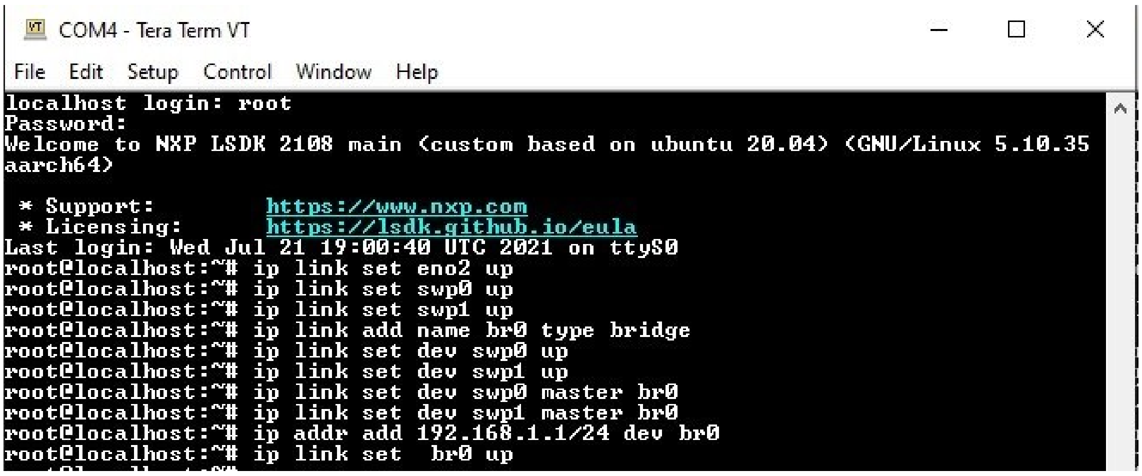
The NXP board provides multiple runtime usage options, the most relevant ones are single port mode, bridge mode, and gateway mode. The single port mode initializes all the ports as independent nodes, but OSI model layer 2 packet forwarding is not enabled, as all the traffic received on the individual nodes is sent to the processor directly. However, the need for this network is to have data link layer packet forwarding as a default so the bridge configuration seemed to be the best match.

Steps required to configure the LS1028ARDB board in bridge mode

1. Enabling eno2 interface
2. Enabling all the required ports or interfaces, for example, swp0,swp1

3. Creating a bridge and moving the ports under the bridge, making the bridge master of the ports
4. Giving bridge IP and enabling bridge interface.

192.168.1.0/24 network address was used for this project. This means that the default routing was done through 192.168.1.1/24. The IP address allocated to the IP camera was 192.168.1.11/24, similarly, the IP address for the server was 192.168.1.10/24. There were no hard and fast rules, the IP addresses were allocated randomly, however, they were kept static for ease of configuration and debugging. Also, note that the server address is the exact address that will be written in the WRM 247+ device.

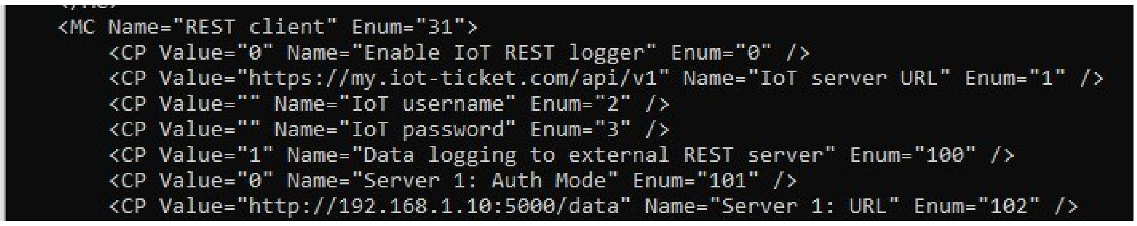


```
localhost login: root
Password:
Welcome to NXP LSDK 2108 main (custom based on ubuntu 20.04) (GNU/Linux 5.10.35
aarch64)

 * Support:        https://www.nxp.com
 * Licensing:      https://lsdk.github.io/eula
Last login: Wed Jul 21 19:00:40 UTC 2021 on ttyS0
root@localhost:~# ip link set eno2 up
root@localhost:~# ip link set swp0 up
root@localhost:~# ip link set swp1 up
root@localhost:~# ip link add name br0 type bridge
root@localhost:~# ip link set dev swp0 up
root@localhost:~# ip link set dev swp1 up
root@localhost:~# ip link set dev swp0 master br0
root@localhost:~# ip link set dev swp1 master br0
root@localhost:~# ip addr add 192.168.1.1/24 dev br0
root@localhost:~# ip link set  br0 up
```

Figure 13. *LS1028ARDB board network configurations*

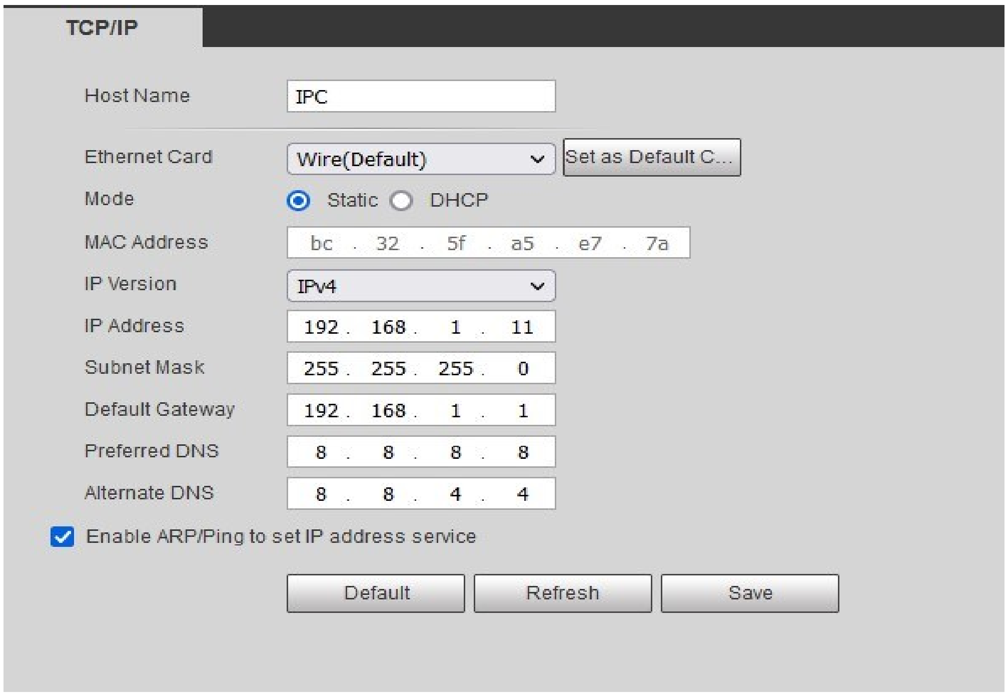
After performing these steps both the boards are now up and running, and all the devices connected to the boards can reach each other on a small internet. The next devices to configure are the IP camera, WRM 247+, and the server. WRM 247+ device continuously produces information that is transferred to the HTTP server address set in its configuration.xml file.



```
    <MC Name="REST client" Enum="31">
        <CP Value="0" Name="Enable IoT REST logger" Enum="0" />
        <CP Value="https://my.iot-ticket.com/api/v1" Name="IoT server URL" Enum="1" />
        <CP Value="" Name="IoT username" Enum="2" />
        <CP Value="" Name="IoT password" Enum="3" />
        <CP Value="1" Name="Data logging to external REST server" Enum="100" />
        <CP Value="0" Name="Server 1: Auth Mode" Enum="101" />
        <CP Value="http://192.168.1.10:5000/data" Name="Server 1: URL" Enum="102" />
```

Figure 14. *WRM247+ REST server change*

This value can vary as the devices come with the default IP address or domain name to send the information. The XML file was edited by remote access as the ssh application is already running on the device. The next part was to give the server an IP address, the process can be different for various operating systems, however, the testing was performed on a Windows machine. The camera device can either take a dynamic IP address or a static IP address that can be configured for it.

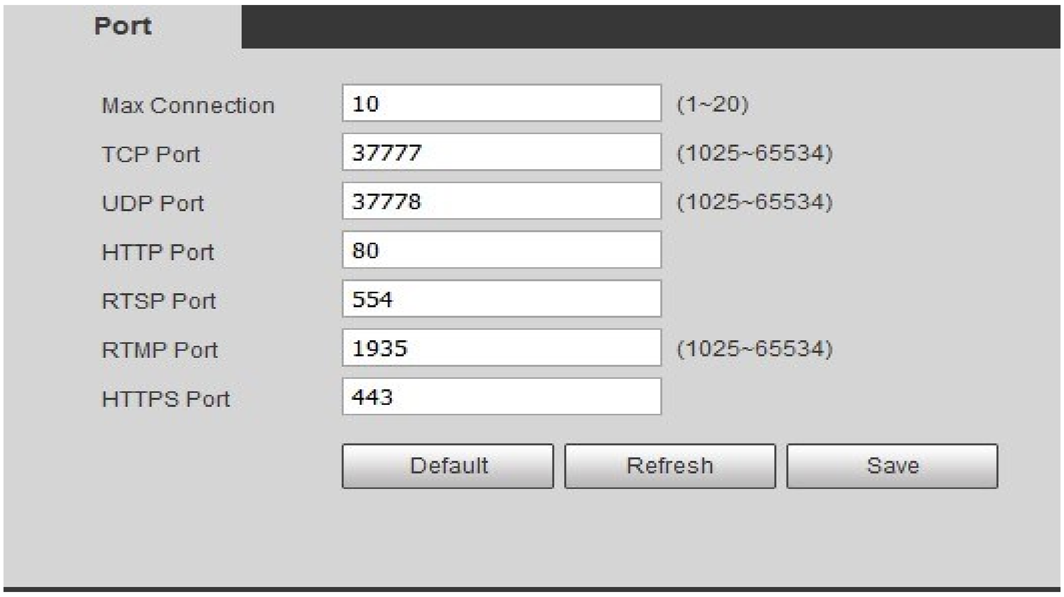


TCP/IP

Host Name	IPC
Ethernet Card	Wire(Default)
Mode	[✓ Static] DHCP
MAC Address	bc . 32 . 5f . a5 . e7 . 7a
IP Version	IPv4
IP Address	192 . 168 . 1 . 11
Subnet Mask	255 . 255 . 255 . 0
Default Gateway	192 . 168 . 1 . 1
Preferred DNS	8 . 8 . 8 . 8
Alternate DNS	8 . 8 . 4 . 4

✓ Enable ARP/Ping to set IP address service

Figure 15. *IP Camera network settings*



Port

Max Connection	10	(1~20)
TCP Port	37777	(1025~65534)
UDP Port	37778	(1025~65534)
HTTP Port	80	
RTSP Port	554	
RTMP Port	1935	(1025~65534)
HTTPS Port	443	

Figure 16. *IP Camera port settings*

For simplicity, a static IP was allocated when setting up the camera device in the initial steps. The port and addresses are set manually, and they serve the purpose that the communication can be held without any issues. Also, note that the arp and ping option is selected to make sure that the device can be visible if the stream is not working. The RTSP port is being set, which was used for debugging purposes. This protocol can be used to stream the data, but a small delay was noticed, and as it is not the main goal of the research the issue was not further investigated. The IP and port settings can be changed from the web interface as well.

Now that all the devices are configured, the devices are now connected to different interfaces of the LS1028ARDB boards. To connect the network as shown in Figure 9, the swp0 and swp1 interface of the first board will be used. Similarly, swp0,swp1 and swp2 interfaces of second boards will be used. The connections can be enabled in any order, there are no specific requirements for it. This will be the final setup, however, there will be different testing scenarios, and they will be discussed in more detail in the next chapter.

4. TESTING AND ANALYSIS

In this chapter, multiple testing scenarios will be discussed. The first two tests will target the first research question mentioned in 1.3. The third and fourth tests pertain to the second research question. Whereas the last test will be used for Ethernet and TSN to answer the third research question. The testing is divided into different phases because Ethernet uses best-effort techniques to send the information whereas, TSN has a whole new way of sending packets. The TSN configurations used for tests 3 and 4 have infinite possibilities, the tests are there to indicate how by adding scheduling and preemptive features the whole dynamics of sending packets is changed by TSN. However, to answer the third research question, the same setup will first use Ethernet and TSN to make a comparative analysis. Note that here again there are unlimited possibilities for TSN configurations, but the settings used in tests 3 and 4 will be used to keep matters simple.

4.1 Best Effort Traffic with Video Stream

This is a small network where a camera device is connected to a LS1028ARDB board, that board is further connected to another LS1028ARDB board which is in return connected to a computer or server. The computer also serves as an iperf client for TCP connections, whereas the second LS1028ARDB board also works as an iperf server. In this manner, the center point or node is overloaded with packets. As iperf acts as a best-effort communication, with no set data rate it tries to send as many packets as it can. Whereas, the camera device or video streaming has a defined data rate. The test was performed for a few seconds considering the length of the log file it generates.

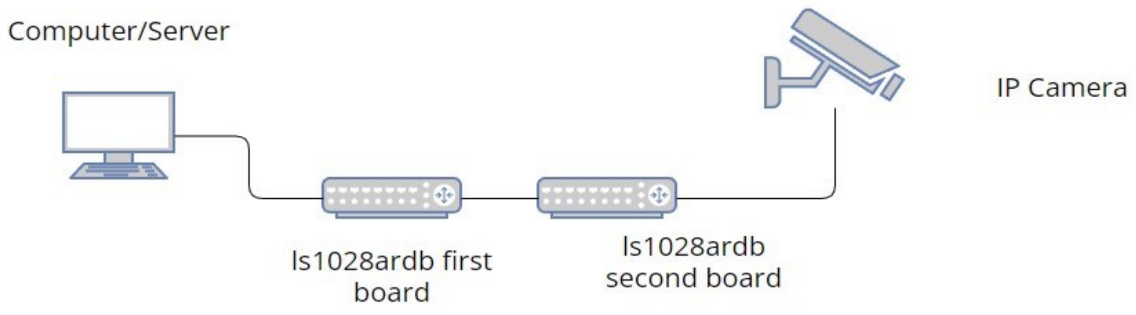


Figure 17. *The network used for tests 1 and 2*

For the network mentioned in Figure 17, there is no special build requirement for the Layerscape Linux image running on the LS1028ARDB boards. Here both the boards are running bridge configurations with 192.168.1.1/24 as the address. The camera and the computer are configured as mentioned in 3.8. The next part is to start Wireshark on the computer and start sniffing the packets that are being generated by both the camera and the iperf server which is running on the second LS1028ARDB board.

The test is run for around 25 seconds, where the IP camera traffic is constantly being pumped from the camera to the computer device. The second traffic type is best-effort traffic which is generated for 10 seconds only, from 5 to 10 seconds as shown in Figure 19.

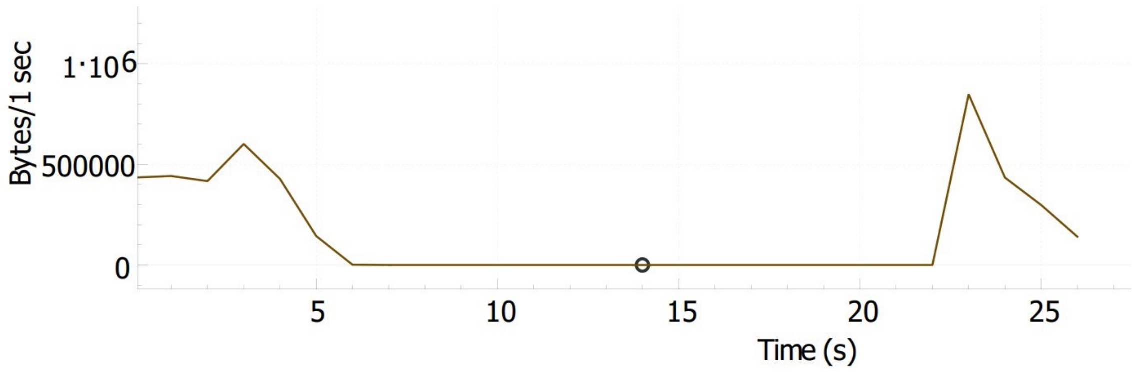


Figure 18. *The data rate of the IP camera stream as seen on the computer throughout test 1*

Figure 18 shows the data rate of the IP camera during the whole duration of the test. There is a strange pattern in the traffic flow. There seems to be no packet transmission from 6 seconds to 22 seconds. This behavior is not normal, as the IP camera was still active, yet no traffic was being received at the computer.

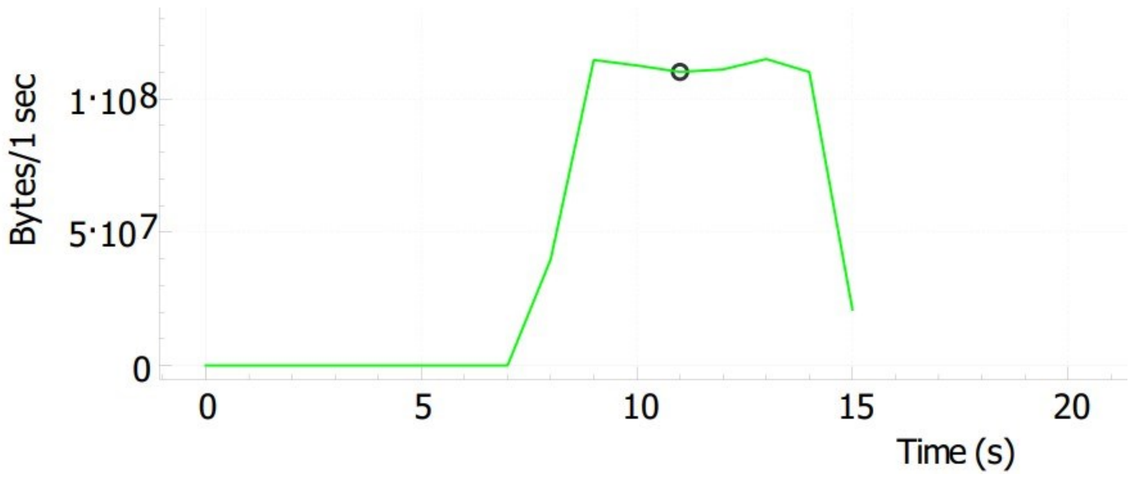


Figure 19. *The data rate of best-effort traffic as seen on the computer throughout test 1*

After analyzing Figure 19, it is clear that the data rate of the IP camera seen in Figure 18 is negligible or close to zero around 6 to 22 seconds due to the reason that the channel is busy, and the best effort traffic has completely occupied the bandwidth during that time. This is not acceptable when considering real-time systems. The way the channel is being misused by iperf traffic it is not allowing any other packet to pass, which will result in catastrophe.

It is clear that Ethernet by default cannot divide the bandwidth efficiently, and there is no way that the link can be trusted to transmit time-critical messages along with best-effort traffic. The next part is to check the latency performance of the Ethernet medium. For the same test, the round trip time ratings were taken for the IP camera.

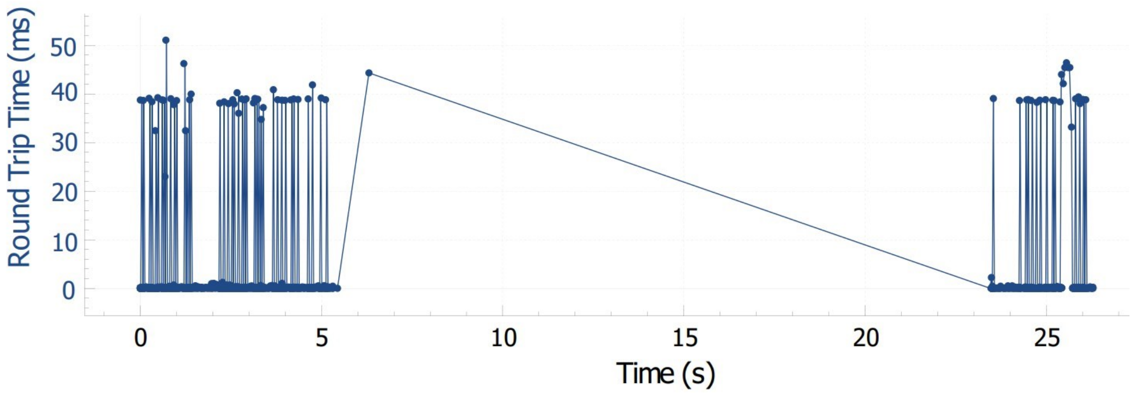


Figure 20. *Round trip time values for IP camera traffic for test 1 from the computer's perspective*

The round trip time graph for IP camera traffic shows that the minimum latency was close to 1 microsecond and the maximum was around 55 milliseconds. In this case, the average latency value does not provide us with any information, rather it will misguide us as the data rate between 6 seconds to 22 seconds is negligible due to the channel being occupied by best-effort traffic.

4.2 Best Effort Traffic with Video Stream Using Traffic Classes

As the traffic on normal Ethernet is unpredictable, the next step is to add traffic classes using the Linux tool tc. By using tc, different streams of data can be allocated priorities as a result the data can then be queued using traffic queues on the Ethernet interface. This configuration is not much different from 4.1, all the devices are set up and connected in the same settings as Figure 17. However, here best effort traffic was allocated priority 5, and the remaining traffic was queued on priority 0.

For this test, there are a few perquisites. The layerscape Linux image provided by the board manufacturer will not suffice here. The default build and boot options do not automatically enable tc options from the Linux kernel. Therefore, the Linux kernel was customized, and the options were enabled. The options that must be enabled are CONFIG

NET ACT POLICE, CONFIG NET SCH INGRESS, CONFIG NET SCH EGRESS, CONFIG NET SCHED, CONFIG NET SCH MQPRIO, CONFIG NET CLS, CONFIG NET CLS ACT, CONFIG NET ACT SKBEDIT, CONFIG NET CLS U32, CONFIG CLS U32 PERF, CONFIG CLS U32 MARK and CONFIG NET EMATCH U32.

After all the options are set, the bootable was prepared using the flex-builder tool provided by NXP. Once the build is ready, we start setting tc rules on the second LS1028ARDB port connecting both boards. This step is necessary as this is the port that is being used for both streams. The camera is using this port to send data to the computer and the same port is being used for iperf traffic, the traffic between the second LS1028ARDB board and the computer.

The set of tc rules used to set up the second board port is tc qdisc add dev swp0 clsact, tc filter add dev swp0 ingress prio 1 u32 match ip sport 5001 0xffff action skbedit priority 5 and tc filter add dev swp0 egress prio1 u32 match ip sport 5001 0xffff action skbedit priority 5

Test 2 execution was done for around 13 seconds, where the IP camera traffic was being sent from the camera to the computer. As there were no commands used for the IP camera, by default it was being sent as priority 0 throughout the duration of the test. The best effort traffic in this case was sent as priority 5, from around 2 seconds to 12 seconds. The traffic received from the IP camera at the computer is similar to Figure 18, which depicts that even if traffic is queued based on different priorities, there is no effect on the Ethernet layer

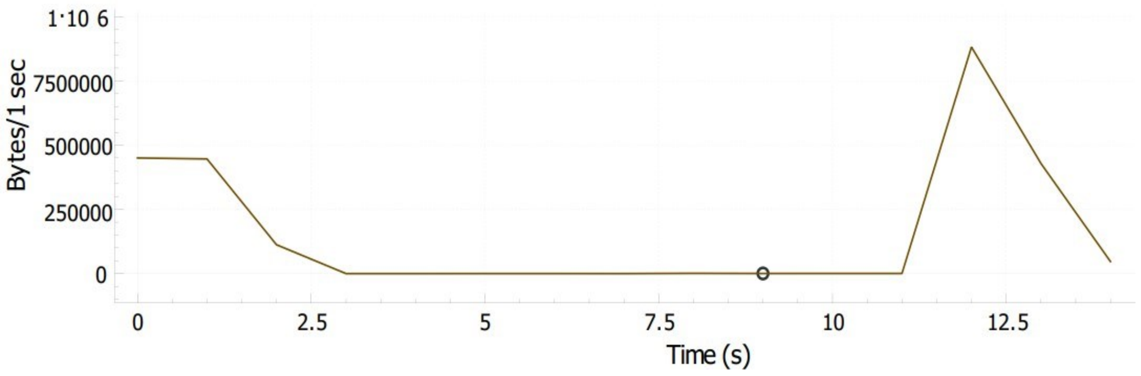


Figure 21. *The data rate of the IP camera as seen on the computer throughout test 2*

Best effort traffic is queued according to priority 5 had almost the same effect on the IP camera traffic as in test 1. When the iperf traffic started to transmit TCP packets the whole Ethernet channel was occupied by it, as a result, negligible or close to zero IP camera packets were being received at the computer during iperf traffic. Therefore, even with queuing traffic with different priorities, Ethernet still cannot make sure that if all the traffic packets will be entertained during the time the data is being sent.

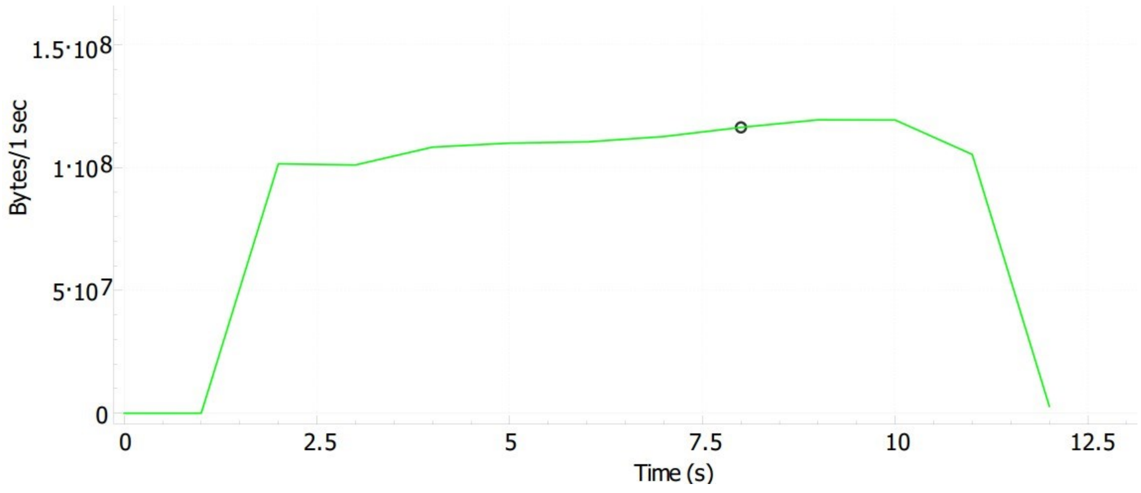


Figure 22. *The data rate of best-effort traffic as seen on the computer throughout test 2*

The latency values for this test were also close to the values seen in test 1. The minimum latency was around 4 microseconds and the maximum was 45 milliseconds which is a slight improvement from test 1. Again the huge 10 seconds gap between 2.5 seconds to 12.5 seconds average latency value is meaningless. However, it can clearly be seen in both the latency graphs that the channel can deliver packets within 50-55 milliseconds given the flow is not disrupted by best-effort traffic.

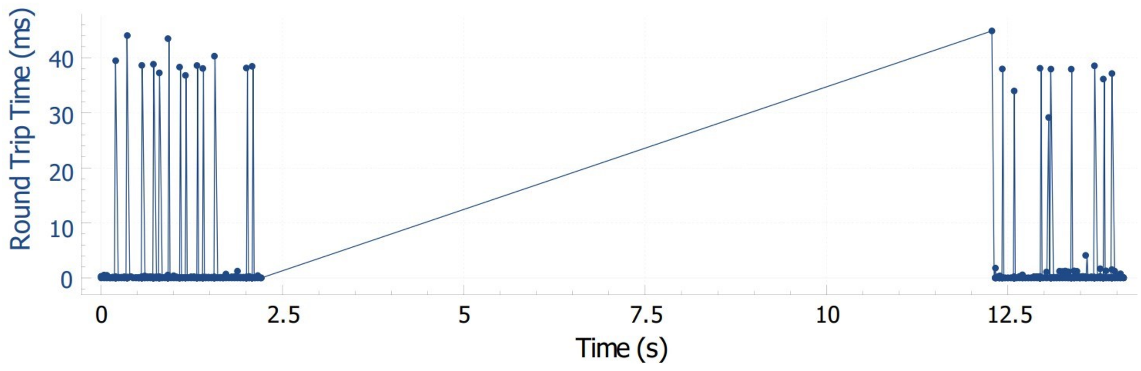


Figure 23. *Round trip time values for IP camera traffic for test 2 from the computer's perspective*

4.3 IEEE 802.1 Qbv Scheduling Best Effort and ICMP

The flow of traffic in Ethernet with or without traffic classes has been rather random, with no determinism, and particularly not safe for time-critical applications. Now, the IEEE 802.1 Qbv standard of TSN will be used to express how it can bring order to the network, with the scheduling of packets as well as controlling different priorities of data using the gate control list. The setup is now changed, and a more simple network configuration is used to clearly explain how TSN capabilities affect the flow of traffic. Instead of an IP camera, another computer is connected to the second LS1028ARDB board. This new device will be sending ping packets to the computer connected to the first LS1028ARDB board. Note that the IP camera stream is not being used, but ICMP traffic is used. This was done to show how traffic with low data rates can be affected when queued with high data rate traffic. Now that TSN features are enabled the port connecting both the boards is now running a gate control list which will provide scheduling of packets according to the values in the lists. The working of the list will be discussed later in this section.

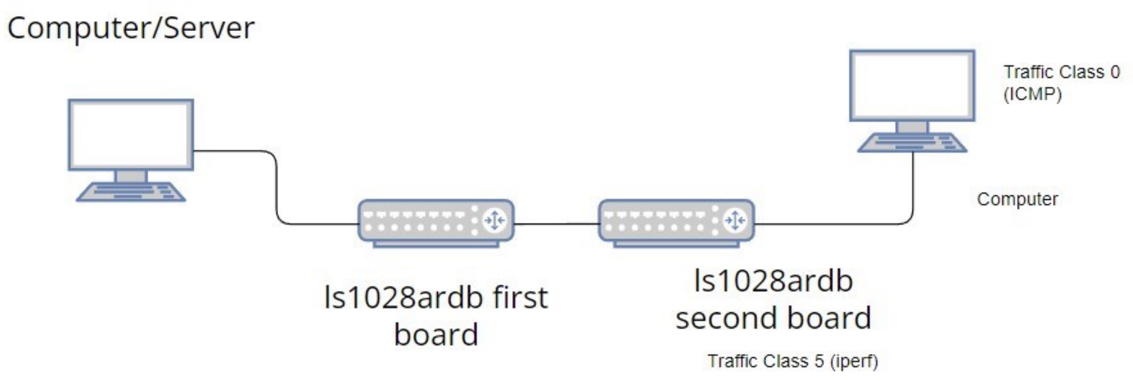


Figure 24. *The network used for tests 3 and 4*

As the tc tools and capabilities were not automatically available on the default Linux image provided by NXP, the same case was observed for the TSN features. The boot options related to TSN were set before the Linux image could be built and used for testing purposes. The kernel options required for TSN were freescale options related to TSN controller ENETC and switches that run TSN also called felix. Other than that, VLAN 8021Q, PTP 1588 CLOCK, MSCC FELIX SWITCH TSN, and MSCC FELIX SWITCH PTP CLOCK options were a must.

Once the board is running with all the options set above, the gate control list can now be set. The IP address of the server computer is still the same 192.168.1.10/24, the new computer device is given an IP address of 192.168.1.4/24 for no specific reason. There were no changes made to the best effort traffic, the iperf server is still running on the second LS1028ARDB board and the client is running on 192.168.1.10/24 computer. The iperf traffic is priority 5 traffic whereas ICMP is priority 0. The gate control list can have multiple entries, but in this case, only two entries are used one where only priority 0 traffic will be transferred and the second where priority 5 traffic will be sent. First, an entry file is needed to create all the entries required against all the queue traffic priorities. Then the file will be set using the qbvset command available in the tsntool package. The order of commands should be as:

```
cat > qbvtest.txt << EOF

t0 00010000b 2000

t1 00000001b 1000000

EOF
```

This will create an entry file, now the following command will set the parameters of an interface's gate control list

```
tsntool  qbvset --device swp0 --entryfile qbvtest.txt
```

Now the interface follows a cycle, where priority 5 is allowed to transfer information for 2000 nanoseconds, and then priority 0 is allowed to send data for 1 millisecond. Note that when the gate is closed for the traffic type, the packets are not discarded rather they are queued, and will start transmitting once the gate opens. This is the reason why this solution alone will not be enough for smooth communication.

The test was run for around 13 seconds. The best-effort traffic starts at around 6 seconds and ends around 16 seconds. The ICMP packets were generated every 1 second, the frequency of the ping packets could be set to a lower value. However, this decision was made to test how the network responds to relaxed conditions. Even with the ping command generating ICMP packets every 1 second there was an abnormality seen in the graph.

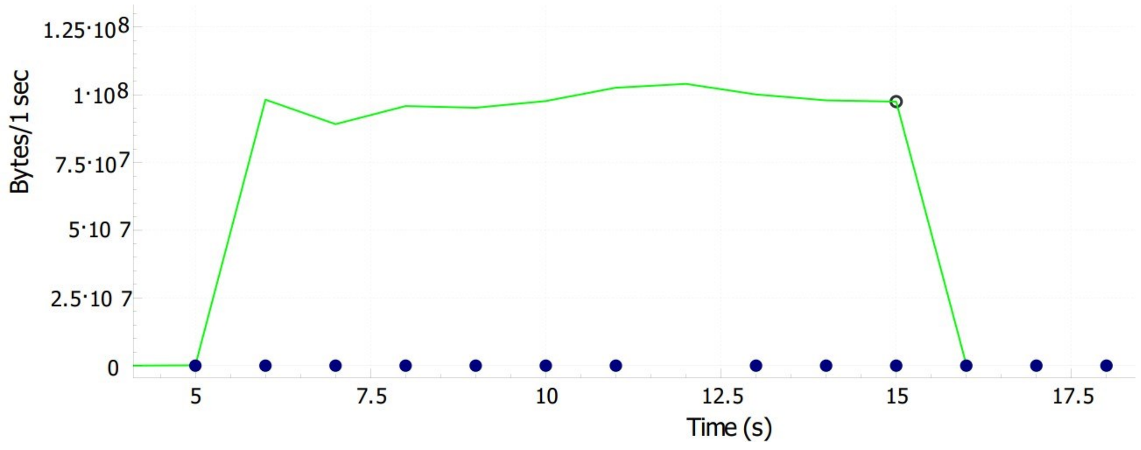
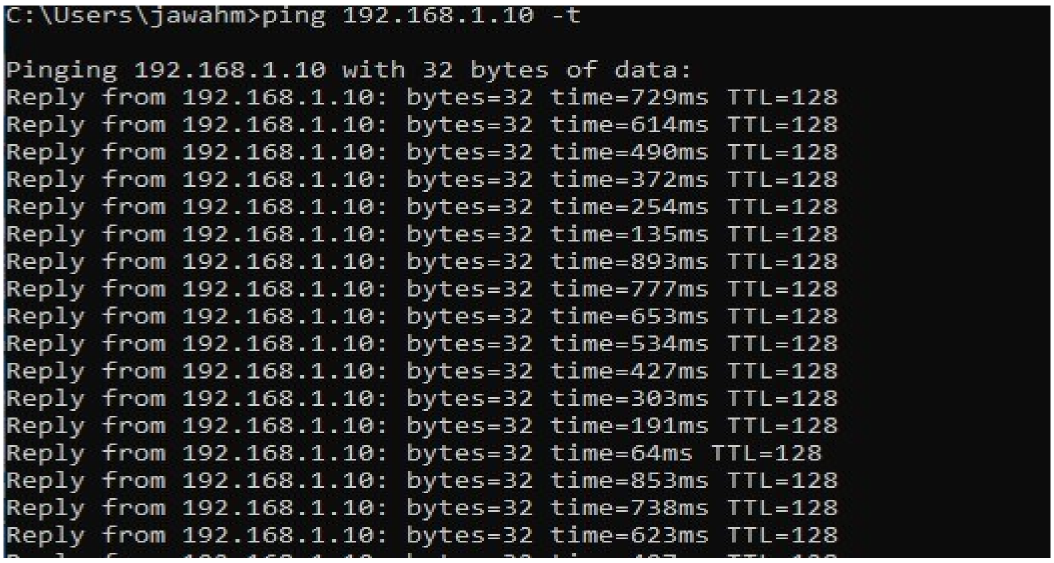


Figure 25. *The data rate of Best effort traffic(green) and critical data packets (blue) as seen on the computer throughout test 3*

Figure 25 depicts how the ICMP packets are regularly sent and received after the best-effort traffic is completed, whereas when the best-effort traffic is being sent the ICMP packet close to 12 and 13 seconds is missed. The question arises, with the ICMP traffic set to communicate for 1 millisecond every 1.00200 milliseconds, how can the packets be missed. The issue here is that when the traffic gate is closed for a specific traffic the traffic then starts to queue on the gate, and as soon as the traffic is allowed again the large packets of best-effort traffic occupy the channel whereas the ICMP packets have to wait. This also results in large jitter and latency inconsistencies, as can be seen in the ping result shown in



```
C:\Users\jawahm>ping 192.168.1.10 -t

Pinging 192.168.1.10 with 32 bytes of data:
Reply from 192.168.1.10: bytes=32 time=729ms TTL=128
Reply from 192.168.1.10: bytes=32 time=614ms TTL=128
Reply from 192.168.1.10: bytes=32 time=490ms TTL=128
Reply from 192.168.1.10: bytes=32 time=372ms TTL=128
Reply from 192.168.1.10: bytes=32 time=254ms TTL=128
Reply from 192.168.1.10: bytes=32 time=135ms TTL=128
Reply from 192.168.1.10: bytes=32 time=893ms TTL=128
Reply from 192.168.1.10: bytes=32 time=777ms TTL=128
Reply from 192.168.1.10: bytes=32 time=653ms TTL=128
Reply from 192.168.1.10: bytes=32 time=534ms TTL=128
Reply from 192.168.1.10: bytes=32 time=427ms TTL=128
Reply from 192.168.1.10: bytes=32 time=303ms TTL=128
Reply from 192.168.1.10: bytes=32 time=191ms TTL=128
Reply from 192.168.1.10: bytes=32 time=64ms TTL=128
Reply from 192.168.1.10: bytes=32 time=853ms TTL=128
Reply from 192.168.1.10: bytes=32 time=738ms TTL=128
Reply from 192.168.1.10: bytes=32 time=623ms TTL=128
```

Figure 26. *Ping result when IEEE 802.1Qbu is not being used.*

The jitter in this case was **205.4** and the average latency, in this case, was **577ms**

The test results showed how much the system was improved just by using one standard of TSN. Other than missing just one packet, the rest of the data made it through the medium. However, there is still an issue of jitter and latency which are unreasonable. This type of system is unreliable when it comes to real-time systems. This surely needed improvement.

4.4 Best Effort and ICMP with IEEE 802.1Qbv and Qbu

The scheduling feature of IEEE 802.1 was able to queue traffic according to the priorities, however, it failed when it came to jitter issues. To counter this issue IEEE 802.1 Qbu standard which deals with frame pre-emption was used alongside IEEE 802.1 Qbv. To test this configuration the setup remained the same as shown in Figure 24, with exactly the same IP addresses and the same gate control list parameters used in 4.3. There are no special Linux kernel options that were needed for the frame-preemption standard, as that was already done in the previous section. The only change in this section and 4.3 was that the priority 0 traffic was marked as express and priority 5 traffic is preemptible. This means that the queued priority packets will only send the required minimum packet while the gate closes during transmission. These packets will resume when the gate reopens for the specific priority traffic. In this manner, the packets in the queue will no longer occupy the channel. This is a result of a smaller guard band which is in effect due to the IEEE 802.1 Qbu standard. The command required to define best-effort traffic as preemptable traffic is tsntool qbuset --device swp0 --preemptable 0x20

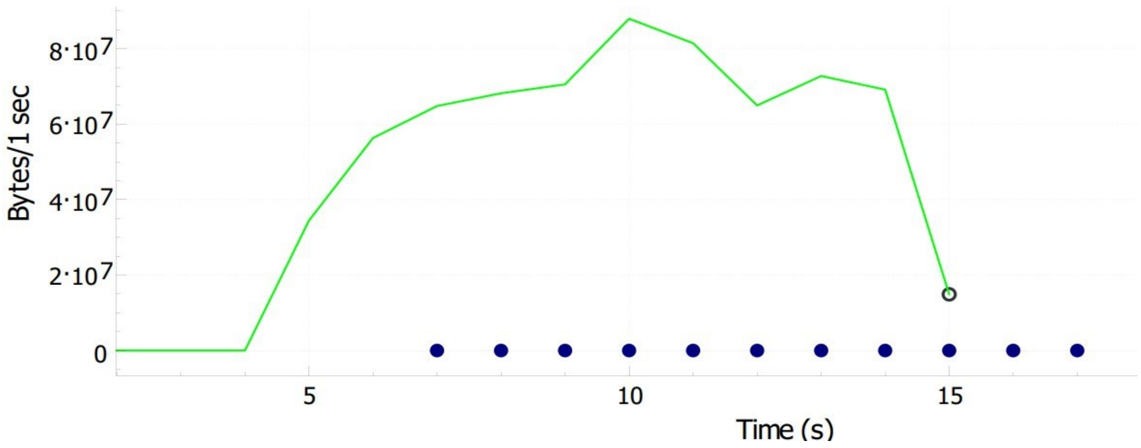
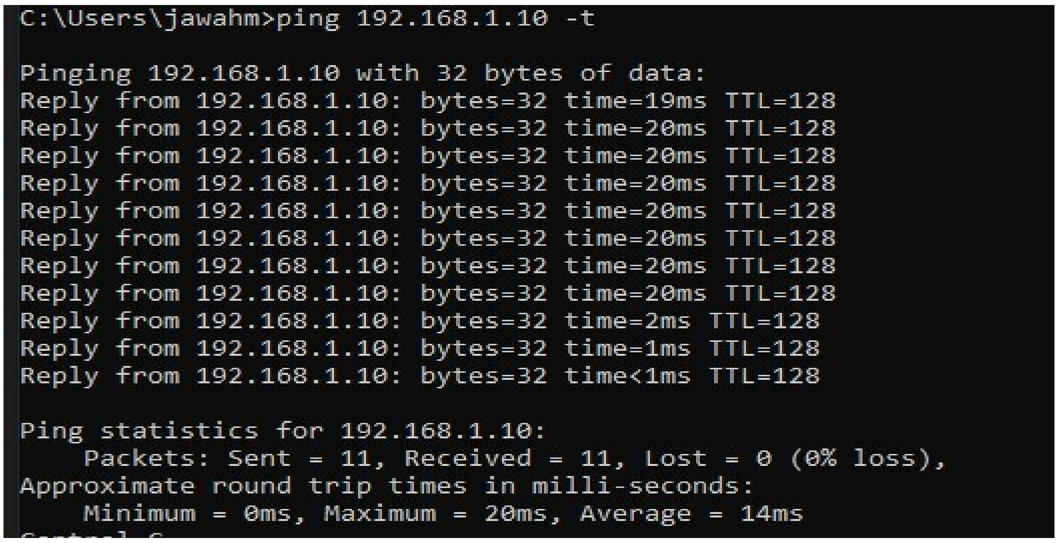


Figure 27. *The data rate of Best effort traffic(green) and critical data packets (blue) as seen on the computer throughout test 3*

The test was run for 15 to 16 seconds, where the best-effort traffic started after 5 seconds and lasted 10 seconds. The second traffic was ping packets, which started around 7 seconds and were kept alive till the end of the test. The most notable difference here is that all the ping packets are successfully sent and received, which was not the case in test 3. There was another notable change that the data rate of the iperf stream was not as smooth as before.



```
C:\Users\jawahm>ping 192.168.1.10 -t

Pinging 192.168.1.10 with 32 bytes of data:
Reply from 192.168.1.10: bytes=32 time=19ms TTL=128
Reply from 192.168.1.10: bytes=32 time=20ms TTL=128
Reply from 192.168.1.10: bytes=32 time=20ms TTL=128
Reply from 192.168.1.10: bytes=32 time=20ms TTL=128
Reply from 192.168.1.10: bytes=32 time=20ms TTL=128
Reply from 192.168.1.10: bytes=32 time=20ms TTL=128
Reply from 192.168.1.10: bytes=32 time=20ms TTL=128
Reply from 192.168.1.10: bytes=32 time=20ms TTL=128
Reply from 192.168.1.10: bytes=32 time=2ms TTL=128
Reply from 192.168.1.10: bytes=32 time=1ms TTL=128
Reply from 192.168.1.10: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.1.10:
    Packets: Sent = 11, Received = 11, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 20ms, Average = 14ms
```

Figure 28. *Ping results with IEEE 802.1 Qbv and IEEE 802.1 Qbv both being .*

The jitter value, in this case, was **1.8**

The average latency was **14.81 ms**

The configurations done for this test turned out to be the most beneficial for critical traffic being queued at priority 0. This was the only test case where the best-effort traffic jumped around. Such flow of traffic for best-effort transmission is never an issue, given the critical packet is being delivered timely, with reduced jitter and latency. The setups used in this test is the best in term of critical packet delivery, jitter, and latency.

4.5 Testing Ethernet and TSN in a Generic Real Life Example

IEEE 802.1 Qbu and Qbv standards have shown how TSN can fix latency and jitter issues in a network. With this being said, the whole system was tested using traffic-generating tools, next a setup close to real life was used to analyze how Ethernet and TSN will perform. It should be clear by now that Ethernet will not be able to compete with TSN. For this test, the network mentioned in section 3.1 was used. For Ethernet, the settings are standard, but the best-effort traffic was queued with priority 5. For TSN, the scheduling cycle was 1 milliseconds for priority 0 traffic and 2 microseconds for priority 5 class, this made a total cycle of 1.00200 milliseconds. Therefore, priority 0 traffic consisted of a video stream from the IP camera and HTTP messages from the WRM247+ device generated every 5 seconds. The TSN test was run for almost 10 seconds, where all the traffic streams were transmitting from 0 seconds.

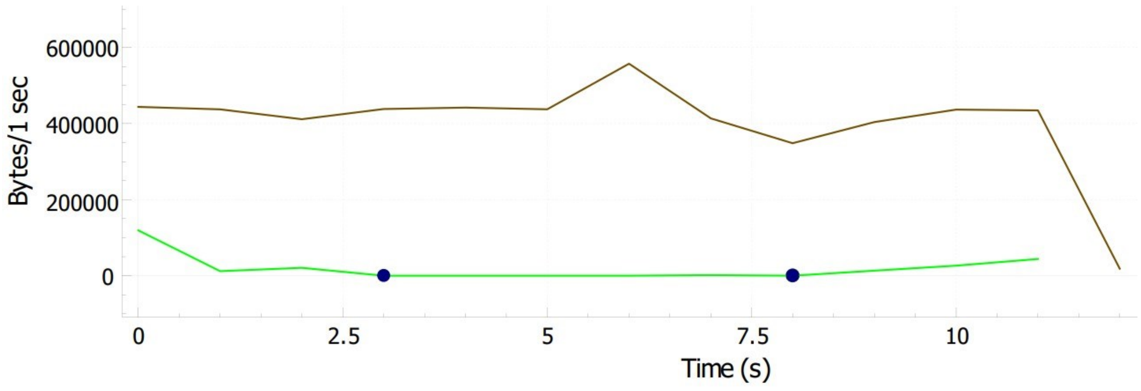


Figure 29. *The data rate of IP Camera stream (brown), best effort traffic(green), and WRM247+ messages(blue) as seen on the computer device with TSN. The network diagram in section 3.1*

The graphs in Figure 29 show the flow of traffic when TSN settings were used. The HTTP messages were received on time, the data coming from the video stream was consistent and the best-effort traffic suffered due to the small transmission window. Note how the best-effort traffic is significantly lower than in test cases 3 and 4. It was due to the frequent transmission of large IP camera stream packets instead of ICMP ping messages sent in section 4.4. With small windows to transmit best-effort data rate decreased and only a few megabytes per second were sent. On the contrary, the data rate before was hundreds of megabytes per second. This configuration shows that TSN made sure that express traffic or in other words critical communication was not being hindered by best effort or less critical packets. Note that the transmission rate of WRM247+ was deliberately kept low, as even this will be affected when discussing Ethernet.

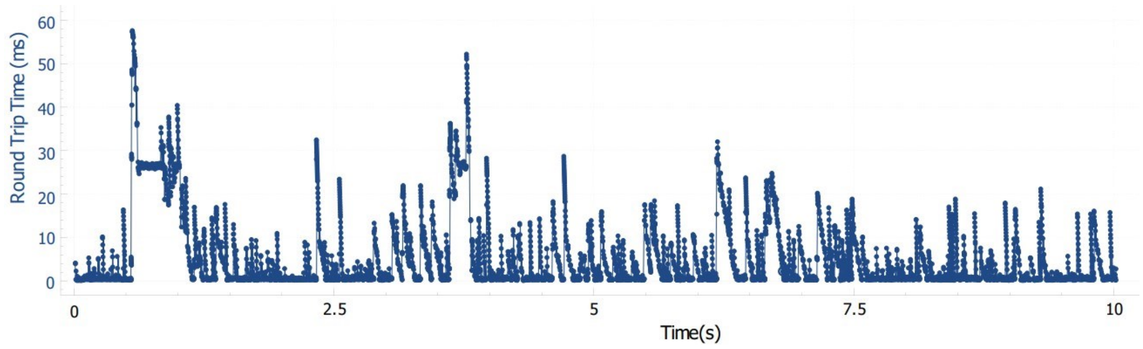


Figure 30. *Latency and Jitter performance of time-critical traffic for test 5 with TSN*

The latency and jitter performance of time packets marked as priority 0 was satisfying. Remember that for TSN the timing window for best-effort traffic was kept small. This was done to give a sense that priority 0 traffic is time-critical and priority 5 traffic is not. Another crucial point to note is that the transmission windows can be of any size, the selection of transmission windows was done to make the best effort traffic struggle. As a result, the jitter and latency values of best-effort traffic were low.

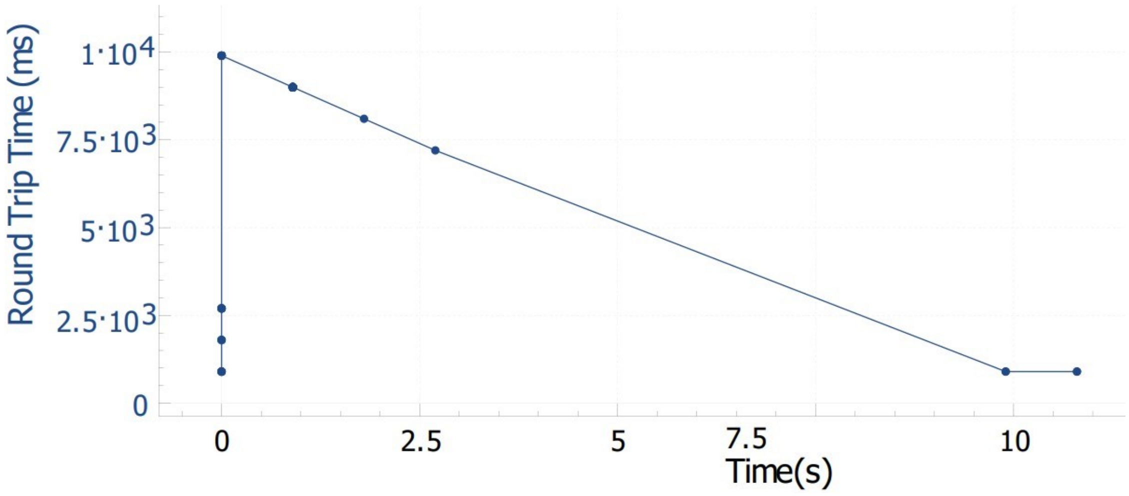


Figure 31. *Latency and Jitter performance of best-effort traffic for test 5 with TSN*

The transmission time of the gate control list can be changed. That is why TSN is the clear winner here, it can divide the whole channel according to the gate control list. This allows it to govern different types of traffic classes. The gate control list has multiple entries, where the same traffic class or prioritized queue can reappear. With the possibility of TSN, unfortunately, we cannot go any further. This will lead to endless configuration of assigning different priorities to different traffic types, then making a gate control list on that basis. Furthermore, the list would then have to be tested with multiple network layouts.

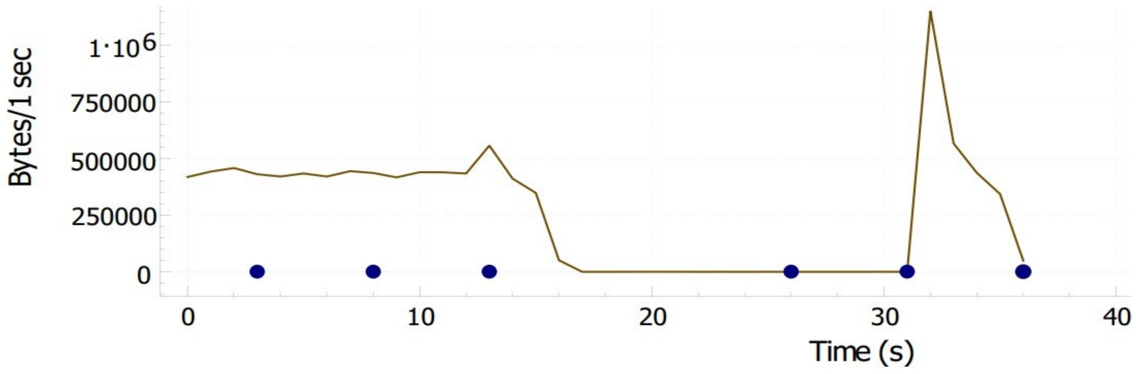


Figure 32. *The data rate of IP Camera (brown) and WRM247+ packets(blue) as seen on the computer during the test execution with Ethernet.* ***Note:*** *The network diagram in section 3.1*

The Ethernet test was run for around 35 seconds. IP camera and WRM247+ devices were generating traffic from the start whereas, best effort traffic was started around 15 seconds. The best-effort traffic was run for 10 seconds. The result in the case of Ethernet was well below par. But it was not unexpected, as the results were similar to test 1 and test 2.

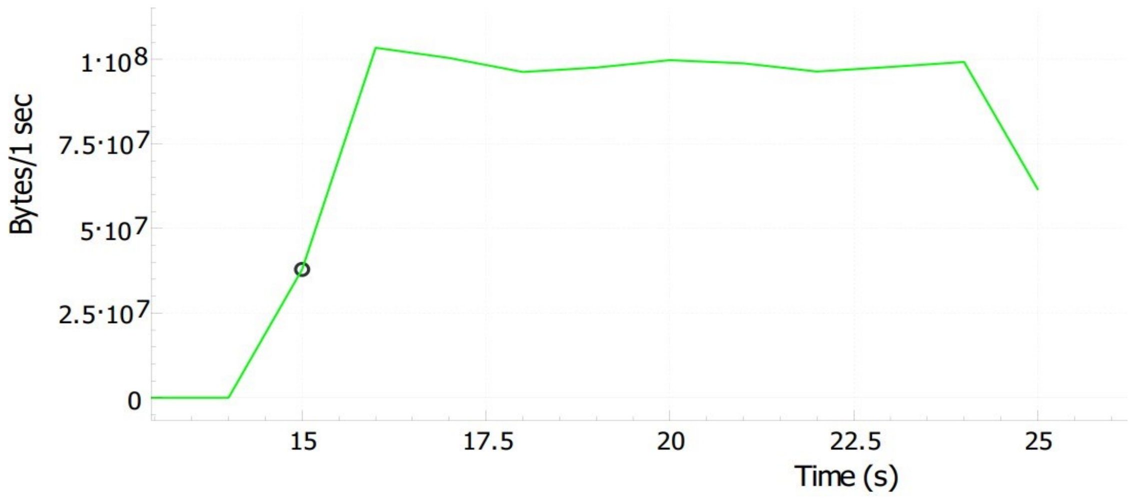


Figure 33. *The data rate of best-effort traffic as seen on the computer during the test execution with Ethernet. Network diagram in section 3.1*

Figure 32 depicts a decrease in the data rate of the IP camera stream as well as missing WRM247+ device packets. The reason is visible in Figure 33. The drop in IP camera data rate and WRM247+ packets around 15 to 26 seconds was due to iperf traffic. The best-effort traffic just occupied that whole bandwidth and kept transmitting until it is stopped around 26 seconds. For 10 seconds straight the channel was busy transmitting iperf traffic, whereas a negligible amount of IP camera packets were sent. The WRM247+ packet is being sent every 5 seconds, but it is surprising to see that even those packets were not allowed between 16-26 seconds when iperf traffic was being transmitted. There is also another uncertainty when it comes to IP camera traffic resumption.

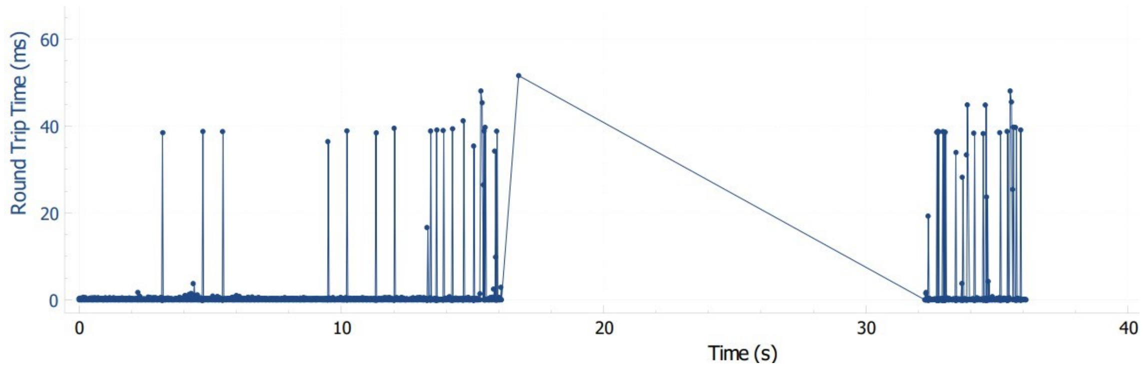


Figure 34. *Latency and Jitter performance of priority 0 traffic in test 5 with Ethernet*

The packets are not started until 32-33 seconds even when there is no other communication taking place other than HTTP messages from WRM247+. This shows that disruptions caused by best-effort traffic put other traffic in an unusual recovery phase, which will result in disasters if the traffic is time-critical systems.

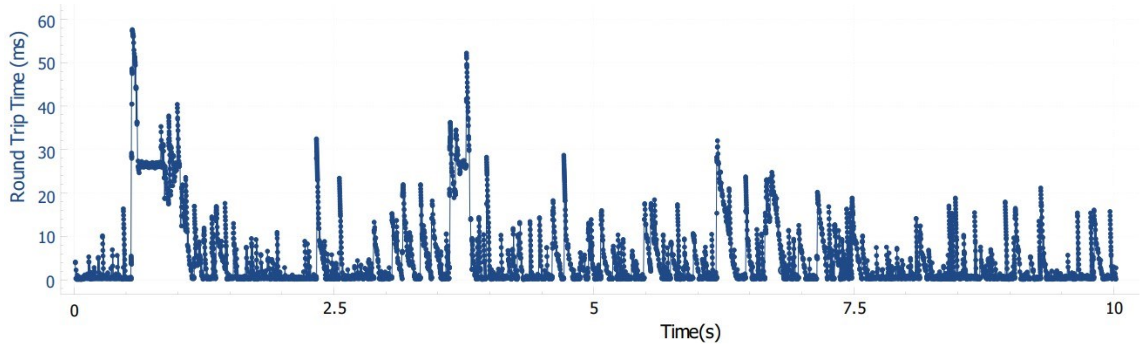


Figure 35. *Latency and Jitter Performance of Priority 5 Traffic in Test 5 with Ethernet*

The latency and jitter performance of priority 0 traffic saw a huge downgrade from TSN whereas the latency and jitter performance of best-effort traffic saw improvements. These were great results for best-effort traffic, however, that was not intended at all. The latency and jitter of only priority 0 traffic mattered. Which is why Ethernet has failed here.

4.6 Comparison Between Ethernet and TSN Based on the Tests Performed

In this chapter, we performed multiple tests with different configurations and network setups on Ethernet and TSN. The first two tests used the same setup to test the limits of Ethernet, with a small addition of prioritizing traffic in the second test case. Tests 3 and 4 were used to evaluate TSN features, where both the test cases used the same network with additional TSN features being enabled for the fourth test. After testing Ethernet and TSN on different networks, it was only logical to test both technologies on the same network configuration. For this purpose, a fifth test was performed. The table provides a summary of all the findings while performing all the tests. The jitter values show the maximum jitter and latency is the average latency. The best-effort traffic values are not discussed here, as that is redundant information, considering the whole point is to send data with priority 0.

Test No.	Test Description	Jitter	Latency	Remarks	Verdict on Time Critical
1	Video Stream with best-effort traffic	52	~20ms	The latency value in this does not carry much significance as there was a period of 15 seconds, where almost no packets were sent	Not recommended
2	Video stream priority 0 and best effort priority 5	44	~3ms	Here again, the latency value is misleading as the was a period of 10 seconds where no packets were sent	Not recommended
3	Ping as priority 0 and best effort priority 5 (IEEE 802.1 Qbv cycle	205.4	577ms	These values are particularly high. For this reason alone the setup is	Not recommended

	1.00020ms with 1ms priority 0 traffic and 0.0020ms best effort traffic)			not recommended, especially since the jitter value being that high is not acceptable.	
4	Ping as priority 0 and best effort priority 5 (Same Qbu settings as before, with the addition of best effort being made preemptable with 802.1 Qbu)	1.8	14.81ms	The jitter value is extremely desirable with that low ping. This is the best setup and is recommended for real-time systems.	Recommended
5	Ethernet performance in real-life scenario as described in section 3.1	55	20ms	Ethernet cannot deal with critical traffic with best-effort traffic, hence such a system should not be used for time-critical applications.	Not Recommended
5	TSN performance on the real-life scenario as mentioned in section 3.1 (same gate control list and preemption settings as used in test 4)	60	15ms	When the configuration recommended in test 4 was put into a real-life scenario, the performance was satisfying	Recommended

Table 4. *Analysis of Tests Performed in Section 4*

5. CONCLUSION

A major part of the thesis was to analyze the main standards of TSN, mainly IEEE 802.1 Qbv scheduling and IEEE 802.1 Qbu frame pre-emption. The main idea was to provide enough discussion points to evaluate how Ethernet and TSN compare. The research was specifically data link layer oriented, and no upper-layer protocols were to be used. The comparison between the two technologies was performed on hardware, which was of prime importance. There are studies related to this topic but most of them deal with simulations and proprietary software, whereas the study was performed on commercially available TSN boards and devices. To evaluate both Ethernet and TSN, a simple yet effective network configuration was required, which should not be complicated to understand yet fulfills the purpose at the same time. For this reason, multiple scenarios were developed, to test both the technologies to the limits. In total, there were five setups, two specifically for Ethernet, two for TSN, and the last network configurations were run on both. The hardware consisted of two LS1028ARDB boards with custom Linux images generated using the tools provided by the manufacturer. They were accompanied by an IP camera and a proprietary Wapice Oy device WRM247+. For testing purposes, the initial four tests were using packets generated by iperf and ping tools.

How well can Ethernet 802.3ab (1000Base-T 1Gbps) transmit critical packets while best-effort traffic is being sent on the same link in terms of jitter and latency?

The first research question pertaining to the use of Ethernet 802.3 ab as a data link layer when transmitting time-critical packets with best-effort messages, has a clear answer. The testing results showed as expected and as discussed during the whole thesis, that Ethernet is not suitable when it comes to time-critical communication. The testing result of Ethernet depicted inconsistencies on every run, regardless of being the same setup. The distribution of traffic was irregular, which in return failed to send packets in time. Ethernet also does not benefit from the queuing traffic on different priorities, there was practically no effect when critical and best-effort packets were sent using 0 and 5 priority respectively. Ethernet also suffered from huge data rate drops during best-effort traffic, rather than data rates of time-critical packets dropping to almost zero. For this reason, the performance of Ethernet 802.3 ab is not satisfactory, therefore it is not recommended.

How well can TSN 802.1 standards transmit critical packets while best-effort is being sent on the same link in terms of jitter and latency?

Regarding the second research question, and evaluation of TSN. The first TSN test only used the scheduling feature of TSN, which uses a gate control list to make sure that the packets are transmitting when the gate is in an open state. In this manner, the critical and best-effort traffic was sent on different queues. This helped with the distribution of the channel according to the gate control list, but without pre-emption, the queued packets would occupy the channel and would transmit even when the gate was closed for a particular priority causing heavy jitter. The solution to this was to enable the frame pre-emption feature of TSN. Once the traffic got controlled using a gate control list, as well as being distributed into express and preemptable traffic the jitter and latency were significantly reduced. With scheduling and preemptive features enabled, TSN was able to deliver timely and with determinism. There were no packet drops or delays in transmission. The performance analysis recommends such a system be used for time-critical applications.

How does an Ethernet 802.3ab connection compare with a connection running 802.1 Qbv and Qbu in terms of jitter and latency?

For the third and final research question, the last test was designed to be a more real-life type to test the same network for Ethernet and TSN. The results of both runs were significantly different in terms of what was transmitted. This came with no surprise as Ethernet had no control over the type of traffic. On the contrary, best effort and express traffic both were being managed by frame pre-emption and scheduling features of TSN. The last test shows how powerful a tool is TSN compared to Ethernet on the data link layer only. There is only one winner when it comes to comparison between Ethernet and TSN, but why is it not being used then. The answer to this question is that TSN is still a relatively new technology, whereas it definitely seems to be the future of the industry. TSN is a data link layer concept which makes it even more desirable in the future. As discussed in earlier chapters, the industry is currently dominated by Industrial Ethernet. Industrial Ethernet makes time-critical messages possible by making the upper layer more stringent to deal with time constraints, whereas the Ethernet layer is changed slightly or no additions to the layer are done at all. TSN is developed in such a way that it can easily replace Ethernet, and the industrial Ethernet tools can be used while TSN is being used as a data link layer. TSN demonstrations are available where famous industrial Ethernet protocols are being used with TSN instead of Ethernet. However, there is still an issue with the availability of TSN.

With respect to the learning outcomes of the study, the topic was of great importance to Wapice Oy given the interest in the latest devices in the market. The study has improved my understanding of a wide range of topics. I have hands-on experience with TSN, which is going to be a big player in future industrial automation and automotive. Other than that, working on embedded Linux images has its advantages as the images are limited in resources. Developing customized bootable and root builds for Linux was also new for me, and it has given me great insight into embedded Linux. Lastly, there was a Wapice Oy based WRM247+ device I was not familiar with. Using it for the thesis proved to be a great opportunity.

REFERENCES

[1] Mall, R., 2009. *Real-time systems: theory and practice*. Pearson Education India.

[2] Li, Q. and Yao, C., 2003. Real-time concepts for embedded systems. CRC press.

[3] Jacob, R., 2022. Challenges and recent advances in the design of real-time wireless Cyber-Physical Systems. *BenchCouncil Transactions on Benchmarks, Standards and Evaluations*, p.100036.

[4] Åkerberg, J., Gidlund, M. and Björkman, M., 2011, July. Future research challenges in wireless sensor and actuator networks targeting industrial automation. In *2011 9th IEEE International Conference on Industrial Informatics* (pp. 410-415). IEEE.

[5] HMS-Networks, "Industrial networks keep growing despite challenging times.",[Online]. Available https://www.hms-networks.com/news-and-insights/news-from-hms/2022/05/02/industrial-networks-keep-growing-despite-challenging-times [Accessed 15 January 2023]

[6] Thomesse, J.P., 2005. Fieldbus technology in industrial automation. *Proceedings of the IEEE*, *93*(6), pp.1073-1101.

[7] Di Natale, M., Zeng, H., Giusto, P. and Ghosal, A., 2012. Understanding and using the controller area network communication protocol: theory and practice. Springer Science & Business Media.

[8] Pazul, K., 1999. Controller area network (can) basics. *Microchip Technology Inc*, *1*.

[9] Santos, M.M.D., Stemmer, M.R. and Vasques, F., 2003, June. Evaluation of the timing properties of two control networks: CAN and Profibus. In *2003 IEEE International Symposium on Industrial Electronics (Cat. No. 03TH8692)* (Vol. 2, pp. 874-879). IEEE.

[10] Drahoš, P. and Bélai, I., 2012. The PROFIBUS protocol observation. *IFAC Proceedings Volumes*, *45*(11), pp.258-263.

[11] Powell, J. and Eng, P., 2013. Profibus and Modbus: a comparison. *Automation. com*, *2013*, pp.1-5.

[12] Deveci, B.U., Bas, H., Ummak, E., Albayrak, O. and Unal, P., 2022, August. A Thorough Analysis and Comparison of Data Communication Protocols Used in Industry 4.0: the Case of Smart-CNC. In *2022 9th International Conference on Future Internet of Things and Cloud (FiCloud)* (pp. 199-206). IEEE.

[13] Watteyne, T., Tuset-Peiro, P., Vilajosana, X., Pollin, S. and Krishnamachari, B., 2017. Teaching communication technologies and standards for the industrial IoT? Use 6TiSCH!. *IEEE Communications Magazine*, *55*(5), pp.132-137.

[14] Law, D., Dove, D., D'Ambrosia, J., Hajduczenia, M., Laubach, M. and Carlson, S., 2013. Evolution of Ethernet standards in the IEEE 802.3 working group. *IEEE Communications Magazine*, *51*(8), pp.88-96.

[15] Latha, V.S. and Rao, D.S.B., 2015, November. The evolution of the Ethernet: Various fields of applications. In *2015 Online International Conference on Green Engineering and Technologies (IC-GET)* (pp. 1-7). IEEE.

[16] Li, Y., Li, D., Cui, W. and Zhang, R., 2011, May. Research based on OSI model. In *2011 IEEE 3rd International Conference on Communication Software and Networks* (pp. 554-557). IEEE.

[17] CloudFlare, "What is the OSI Model?", [Online]. Available https://www.cloudflare.com/learning/ddos/glossary/open-systems-interconnection-model-osi/ [Accessed 2 February 2023]

[18] Profinet,"Topology Options: Fieldbuses and Industrial Ethernets", [Online] Available https://us.profinet.com/topology-options-fieldbus-industrial-Ethernet/ [Accessed 5 February 2023]

[19] Bisht, N. and Singh, S., 2015. Analytical study of different network topologies. *International Research Journal of Engineering and Technology*, *2*(1), pp.88-90.

[20] Danielis, P., Skodzik, J., Altmann, V., Schweissguth, E.B., Golatowski, F., Timmermann, D. and Schacht, J., 2014, September. Survey on real-time communication via Ethernet in industrial automation environments. In *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)* (pp. 1-8). IEEE.

[21] IEEE 802.1D Edition 2004 IEEE Standard for Local and metropolitan area networks-Media Access Control (MAC) Bridges Institute of Electrical and Electronics Engineers Std, 2004, IEEE

[22] IEEE 802.1Q Edition 2005 IEEE Standard for Local and metropolitan area networks-Virtual Bridged Local Area Networks Institute of Electrical and Electronics Engineers Std, 2005, IEEE

[23] Moraes, R., Carreiro, F.B., Bartolomeu, P., Silva, V., Fonseca, J.A. and Vasques, F., 2011. Enforcing the timing behavior of real-time stations in legacy bus-based industrial Ethernet networks. *Computer Standards & Interfaces*, *33*(3), pp.249-261.

[24] Lee, K.C. and Lee, S., 2002, November. Performance evaluation of switched Ethernet for networked control systems. In IEEE 2002 28th Annual Conference of the Industrial Electronics Society. IECON 02 (Vol. 4, pp. 3170-3175). IEEE.

[25] Nof, S.Y. ed., 2009. Springer handbook of automation (pp. 1379-1396). Berlin, Heidelberg: Springer Berlin Heidelberg.

[26] Neumann, P., 2007. Communication in industrial automation—What is going on?. Control Engineering Practice, 15(11), pp.1332-1347.

[27] Klasen, F., Oestreich, V. and Volz, M. eds., 2011. Industrial Communication with Fieldbus and Ethernet. VDE-Verlag.

[28] Rostan, M., 2008, November. Industrial Ethernet technologies: overview. In ETG Industrial Ethernet Seminar Series, Nuremberg.

[29] Modbus Organization, 2006. MODBUS Messaging on TCP/IP Implementation Guide: V1. 0b. Modbus Organization.

[30] EtherCAT,"EtherCAT Technology Overview" [Online], Available https://www.ethercat.org/en/technology.html [Accessed 5 February 2023]

[31] CC-Link, "CC-Link IE Field Network" [Online], Available https://www.cc-link.org/en/cclink/cclinkie/cclinkie_f.html

[32] Schiffer, V., 2001, October. The CIP family of fieldbus protocols and its newest member-Ethernet/IP. In ETFA 2001. 8th International Conference on Emerging Technologies and Factory Automation. Proceedings (Cat. No. 01TH8597) (pp. 377-384). IEEE.

[33] Ficzere, D., Patel, D., Sachs, J., Ansari, J., Soós, G. and Varga, P., 2022, April. 5G public network integration for a real-life PROFINET application. In NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium (pp. 1-5). IEEE.

[34] Robert, J., Georges, J.P., Rondeau, É. and Divoux, T., 2012. Minimum cycle time analysis of Ethernet-based real-time protocols. International Journal of Computers, Communications and Control, 7(4), pp.743-757.

[35] Bello, L.L. and Steiner, W., 2019. A perspective on IEEE time-sensitive networking for industrial communication and automation systems. Proceedings of the IEEE, 107(6), pp.1094-1120.

[36] Finn, N., 2018. Introduction to time-sensitive networking. IEEE Communications Standards Magazine, 2(2), pp.22-28.

[37] Samii, S. and Zinner, H., 2018. Level 5 by layer 2: Time-sensitive networking for autonomous vehicles. IEEE Communications Standards Magazine, 2(2), pp.62-68.

[38] Gutiérrez, M., Steiner, W., Dobrin, R. and Punnekkat, S., 2017, April. Synchronization quality of IEEE 802.1 AS in large-scale industrial automation networks. In 2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS) (pp. 273-282). IEEE.

[39] Steiner, W. and Dutertre, B., 2013. The TTEthernet synchronisation protocols and their formal verification. International Journal of Critical Computer-Based Systems 17, 4(3), pp.280-300.

[40] NXP, "LS1028A Reference Design Board Fact Sheet" [Online]. Available: https://www.NXP.com/docs/en/fact-sheet/LS1028ARDBFS.pdf [Accessed 26 April 2023]

[41] Dahua, "Entry Series | IPC-HFW1235S-W-S2 datasheet" [Online]. Available : https://www.dahuasecurity.com/asset/upload/uploads/cpq/prm-os-srv-res/smart/datasheetzipfiles/IPC-HFW1235S-W-S2_datasheet_20210127.pdf [Accessed 26 April 2023]

[42] Wapice Oy, “IoT-Ticket_brochure” [Online]. Available : https://www.wapice.com/media/brochures/IoT-Ticket_brochure.pdf [Accessed 27 April 2023]

[43] NXP, “Layerscape Software Development Kit User Guide” [Online]. Available: https://www.NXP.com/docs/en/user-guide/LSDKUG_Rev21.08.pdf [Accessed 27 April 2023]

[44] Máté, M., Simon, C. and Maliosz, M., 2022. Asynchronous time-aware shaper for time-sensitive networking. Journal of Network and Systems Management, 30(4), p.76.

[45] Zhang, P., Liu, Y., Shi, J., Huang, Y. and Zhao, Y., 2019. A feasibility analysis framework of time-sensitive networking using real-time calculus. *IEEE Access*, *7*, pp.90069-90081.

[46] Iperf, “iPerf - The ultimate speed test tool for TCP, UDP and SCTP” [Online]. Available : https://iperf.fr/iperf-doc.php [Accessed 08 May 2023]

[47] SSH, “The SSH Protocol” [Online]. Available: https://www.ssh.com/academy/ssh#the-ssh-protocol [Accessed 08 May 2023]

[48] TC, “tc(8) — Linux manual page” [Online]. Available: https://man7.org/linux/man-pages/man8/tc.8.html [Accessed 08 May 2023]

[49] Ethtool, “ethtool(8) – Linux man page” [Online] Available : https://linux.die.net/man/8/ethtool [Accessed 12 May 2023]

[50] Wireshark, “8.8. The “I/O Graphs” Window” [Online] Available : https://www.wireshark.org/docs/wsug_html_chunked/ChStatIOGraphs.html [Accessed 12 May 2023]

[51] NXP, “Layerscape LS1028A Reference Design Board” [Online] Available: https://www.NXP.com/design/qoriq-developer-resources/layerscape-ls1028a-reference-design-board:LS1028ARDB [Accessed 12 May 2023]