Analysis of automotive traffic shapers in ethernet in-vehicular networks

Thangamuthu, S.

Award date:
2014

Department of Mathematics and Computer Science
System Architecture and Networking Group

ANALYSIS OF AUTOMOTIVE TRAFFIC SHAPERS IN ETHERNET IN-VEHICULAR NETWORKS

Master Thesis

SIVAKUMAR THANGAMUTHU

Supervisors:
dr.ir. Pieter J. L. Cuijpers
prof.dr. Johan J. Lukkien
dr. Nicola Concer

Version 1.0

Eindhoven, August 29, 2015

Abstract

Ethernet is expected to be adopted as the backbone for in-vehicle networks to unify different automotive domains under a single infrastructure. However, the QoS guarantees offered by Ethernet are not sufficient for time-critical control data traffic present in the automotive networks. IEEE 802.1 Time-Sensitive Networking Task Group is working on standardizing different mechanisms that are required to improve the QoS guarantees of Ethernet for automotive and industrial applications. Traffic shaping is one of the mechanisms to improve the end-to-end latency guarantees of Ethernet for the time-critical control data traffic. There are different proposals for a new Traffic Shaper in the IEEE802.1TSN standardization process. The problem is to verify whether these Traffic shapers actually meet the end-to-end latency requirement specified by IEEE802.1TSN and also to identify the best possible Traffic Shaper for automotive network requirements in terms of other performance metrics like configuration complexity and hardware impact. In this thesis work a simulation model is created for three different Traffic Shapers that are proposed in standardization process for shaping the control data traffic. Their performances in terms of end-to-end latency and jitter are compared and verified against the IEEE802.1TSN requirements. Simulation results show that it is possible to achieve the IEEE802.1TSN latency expectation of 100 µs over 5 hops for the control data traffic in all the three traffic shapers under different conditions. Based on the analysis for overall performance, the Burst Limiting shaper is recommended.

Acknowledgement

I would like to express my sincere gratitude to my supervisor at NXP, Dr. Nicola Concer for his exemplary guidance and support throughout the thesis work. Without his insights and constant motivation this work would not be possible. Special thanks to my supervisor at the university, Dr. Pieter J. L. Cuijpers for his guidance and critical comments to bring out the best in me. I would like to extend my profound gratitude to Prof. Dr. Johan J. Lukkien for his valuable suggestions and his confidence in me to handle this thesis work.

Also, I would like to thank Dr. Bas Luttik and EIT ICT labs for introducing me to this project. Futhermore, I would like to thank NXP Semiconductors for giving me the opportunity to work on this project. Thanks to all the employees and fellow interns at NXP for providing me a friendly environment at work.

I am forever grateful to my parents and my uncle for their love and support, without which this endeavor would not be possible. Finally, my hearty thanks to all my friends for their encouragement, moral support and good times.

“தாமின் புறுவது உலகின் புறக்கண்டு
காமுறுவர் கற்றறிந் தார்”

- திருவள்ளுவர்

Translation

“Their joy is joy of the entire world, they see; thus more
the learners learn to love their cherished lore”

- Thiruvalluvar

Meaning

Good scholars will crave to learn more, when they see that while the learning gives joy to them, the entire world also derives joy from it.

Table of Contents

Abbreviations

ECU	Electronic Control Unit
LIN	Local Interconnect Network
CAN	Controller Area Network
MOST	Media Oriented Systems Transport
ADAS	Advanced Driver Assistance Systems
IVN	In-Vehicle Network
TDMA	Time Division Multiple Access
CSMA	Carrier Sense Multiple Access
MAC	Media Access Control
LLC	Logical Link Control
QoS	Quality of Service
AVB	Audio Video Bridging
TSN	Time Sensitive Networking
CDT	Control Data Traffic
Mbps	Mega bits per second
Gbps	Giga bits per second

1. Introduction

1.1 Background

Automobiles for human transport have undergone a tremendous evolution since the birth of the first internal combustion engine based car in 1886 [1]. The cars of the present day are becoming more intelligent with a multitude of complex electronic systems inside them. And the number of electronic control units (ECUs) inside a car is steadily increasing due to the performance, safety and comfort requirements. A top of the line modern car can have up to 70 ECUs [2]. This growth in the number of ECUs inside a car has given rise to different in-car networking technologies like LIN, CAN, FlexRay and MOST [3]. All of these networking technologies are optimized for specific use cases. For instance, CAN is a cost-effective relatively low speed bus network used to connect different ECUs. FlexRay with a maximum bandwidth of 10Mbits/s was designed for dependable real-time communication to connect time critical ECUs like the engine control unit. MOST was developed with a focus of transporting high quality audio data between entertainment ECUs inside the car.

The recent developments in the automotive safety systems like ADAS integrate ECUs, camera and radars providing features like Pre-Crash collision warning, Lane departure warning, Pedestrian warning etc.,. IMS Research predicts that by 2017, camera-aided driver assistance systems alone will number 34 million [4]. Additionally, there is a growing demand for in-vehicle multimedia systems with features like engine noise cancellation, engine noise mimicking, external device connectivity for rear-seat entertainment etc.,. A research study from ABI Research predicts that the global connected infotainment shipments will surpass 50 million units by 2017. Moreover, by 2025, every car will have Internet connectivity [4]. These trends and developments are pushing the bandwidth capabilities of the present day in-vehicle network (IVN) technologies to their limits. Besides this, the networking technology used has to be future proof, as it has to withstand, comply and facilitate the addition of other technological developments throughout the lifetime of a vehicle, for e.g. vehicle to vehicle communication, self-driving vehicles etc.,. This demands flexibility in network configuration and network management with minimal cabling complexity. Above all, these additions should not significantly increase the weight and the cost of the vehicle. So a new approach in IVN is necessary to transform the decentralized network islands with different networking protocols to a hierarchical structure. These constraints drive the automotive industry towards the need for a cheap, easy to manage, convergent backbone for the in-vehicle network.

On the other hand, Ethernet has evolved to become the dominant technology for computer networking and it has already entered the automotive domain via the On-Board Diagnostics (OBD) interface. Ethernet could be a possible technology for the future IVN because of its following advantages

1. It has grown into an extremely mature technology since its beginning in 1970s [5], with billions of interfaces sold in both consumer and industrial automation domains
2. It constitutes a cheap and highly supportive software stack spanning from super computers to small embedded systems with constant evolution from large industries worldwide to adapt to the evolving requirements

3. It is flexible as it is available in different data transmission rates such as 10/100/1000 Mbits/s and can be configured in different types of topologies, redundancy schemes and for different Quality of Service (QoS)

But the recent trigger point for considering Ethernet as a potential technology for the IVN came with the introduction of Broad-R-Reach technology by BroadCom [6] along with the increasing number of works to extend Ethernet's capabilities for real-time guarantees. Broad-R-Reach offers full-duplex switched Ethernet communication over an unshielded twisted pair cable. This provides Ethernet as a cost effective alternative for the backbone of next generation IVN. So, switched Ethernet-based communication is gaining significant importance in IVN. Table 1.0 introduces the different in-vehicle technologies including the possibilities of Ethernet and their typical applications inside the vehicle.

	LIN	CAN	CAN FD	FlexRay	MOST	Ethernet
Standard Release	2002	1986	2012	2000	2000	2011
Raw Data Rate	1-20Kbps	Up to 1Mbps	Up to 8/10Mbps	Up to 10Mbps	50 Mbps 150 Mbps	100Mbps
Payload	Up to 8 B	Up to 8 B	Up to 64 B	Up to 254 B	60, 117 or 370 B	Up to 1500 B
Physical Media	Single Wire	Unshielded Twisted Pair	Unshielded Twisted Pair	Unshielded Twisted Pair	UTP/Plastic Optical Fiber	Unshielded Twisted Pair
Topology	Bus	Bus	Bus/Passive Star	Bus/Star/Mixed	Ring	Star/Tree/Ring
Automotive Application	Sensors & actuators	Body & Powertrain	Body, Powertrain, Distributed Control, Chassis	Distributed control, Backbone, Chassis	Multimedia	Diagnostics, Backbone, ECU programming, Multimedia
Media Access	Master-Slave	CSMA-CR	CSMA-CR	TDMA	TDMA	Buffered network
Error detection	8-bit CRC	15-bit CRC	17/21-bit CRC	24-bit CRC	CRC	32-bit CRC
Redundancy	NO	NO	NO	Built in: two separate channels	Double ring, redundancy possible	No, possible via link replication
Deterministic Behavior	NO	NO	NO	Built in	Built in	Via AVB, TSN

Table 1.1 Different IVN Networking topologies

1.2 Context

The automotive network consists of different types of data traffic which include time-sensitive control data traffic from sensors and between ECUs, real-time audio/video streams and internet traffic. Each traffic type requires a specific QoS. In the traditional IVN, these traffic types are organized in different network domains and handled by separate networks. The cross communication between different network domains happens through network gateways. In IVN with switched Ethernet, ECUs are connected via one or more Ethernet switches that create the backbone of the network and provide connectivity between any pair of ECUs. In addition to offering an increased data rate and bandwidth, the Ethernet IVN backbone

should integrate different automotive network domains. This includes the transportation of time-sensitive data traffic with certain guaranteed QoS in terms of communication latency, reliability and redundancy. Figure 1.1 shows the possibility of converged backbone in IVN.

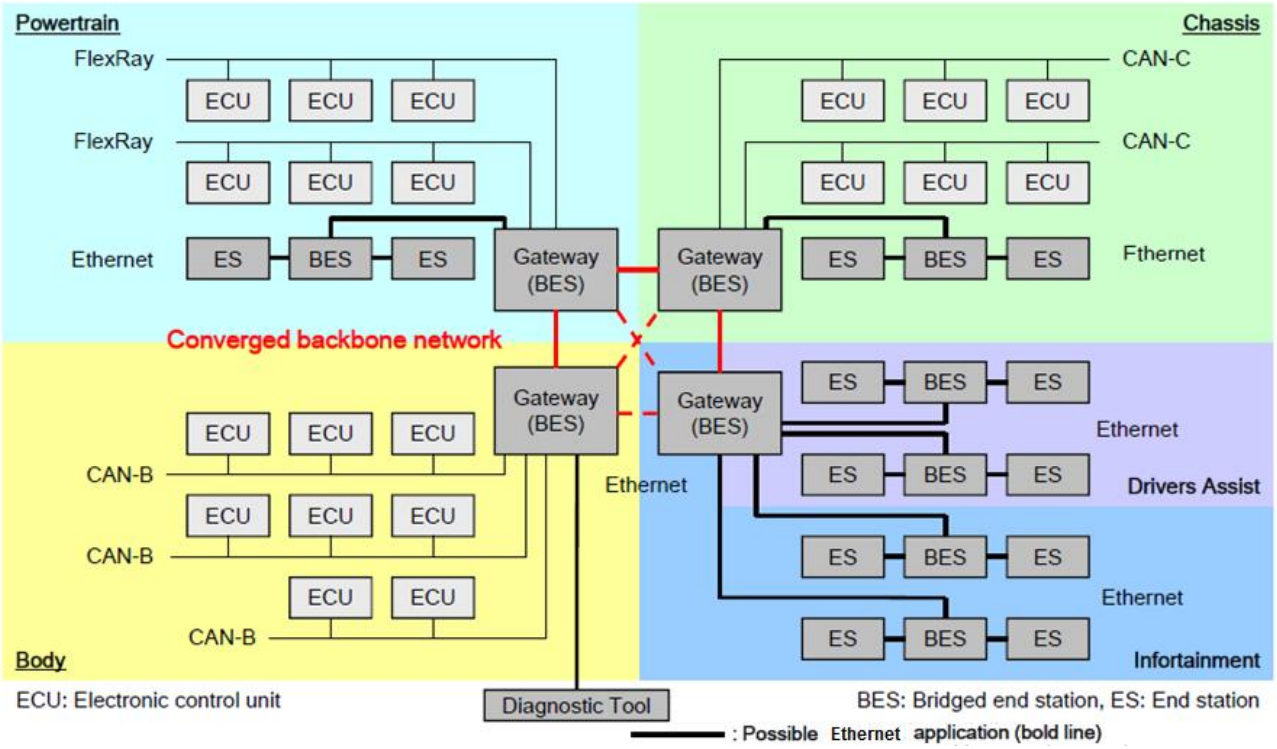


Figure 1.1 Possible Converged backbone IVN

The traditional Ethernet protocol for computer networks is optimized only for "Best Effort Delivery" of a message [7]. This means that it does not provide any guarantee that data is delivered or that a guaranteed quality of service level is assured. "Best Effort" in many cases, works best for lightly loaded networks where the average delay is the primary metric or for cases where it is difficult to classify the data traffic based on their time sensitiveness. "Best Effort" is not sufficient in cases where the worst case delay is the important metric. In the context of IVN, "Best Effort Delivery" is not sufficient due to the following reasons

1. Automotive entertainment and safety systems deal with live audio and video data, which require a fixed end-to-end delay for quality reception
2. Control messages from sensors to ECUs and between different ECUs in the automotive control domain have varied but even more stringent end-to-end delay requirements to respond in a time-critical manner

These requirements emphasize that the core aspect of IVN backbone is to deliver data within a certain time window under a specified maximum delay. So, to establish Ethernet as an IVN backbone and to accommodate the time-sensitive traffic from protocols like CAN, FlexRay and MOST in the in-vehicle networking domain, it becomes essential to extend the Ethernet to offer guaranteed QoS in terms of end-

to-end latency for time-sensitive data traffic. The specific requirements for automotive network are detailed in the Chapter 5.

1.3 Basis for the Work

The capabilities of the switched Ethernet to accommodate real-time data traffic have been investigated extensively. Chapter 2 explains more on the approaches and different mechanisms to improve switched Ethernet to offer QoS guarantees for time-sensitive data traffic. In parallel to these works, the switched Ethernet standard itself is being extended to offer QoS guarantees. Figure 1.2 shows the evolution and the current state of Ethernet standards to achieve time triggered and reliable communications.

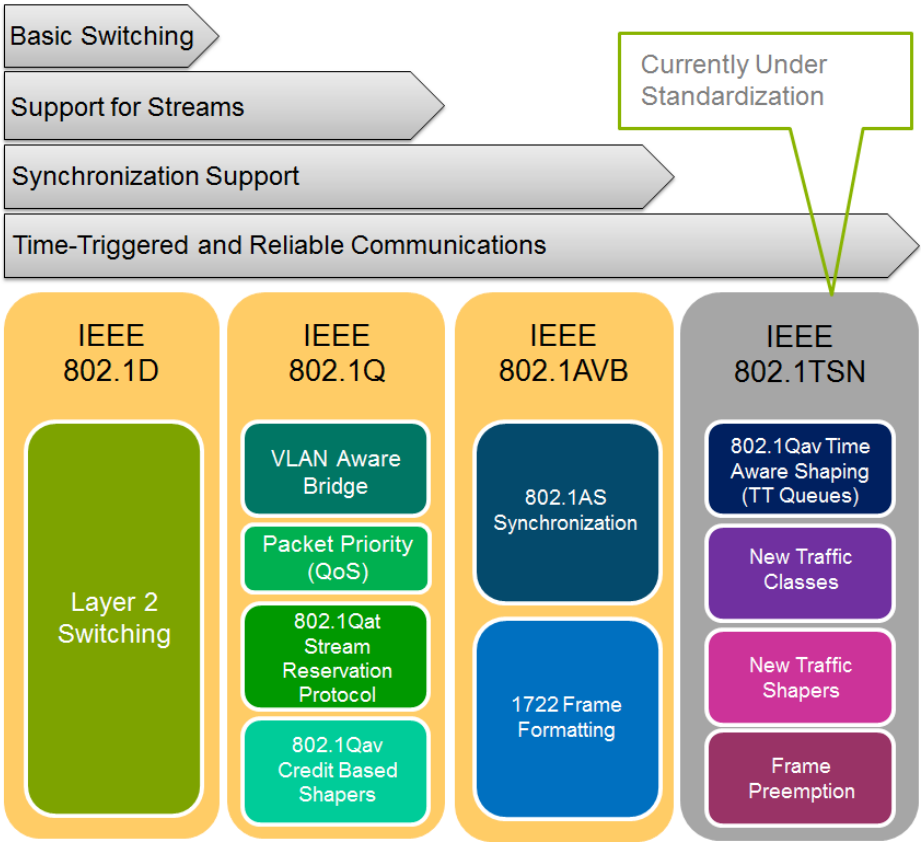


Figure 1.2 Evolution of switched Ethernet Standards

The basic switching features offered by IEEE802.1D standard were not sufficient for traffic patterns with QoS requirements for streaming or for industrial and automotive domains. IEEE802.1Q standard extended the available features by adding support for Virtual LANs (VLANs) and providing traffic priorities in order to facilitate traffic classification and queuing of different data streams. A brief description of traffic classification is given in Chapter 3. The initial standard IEEE802.1Q-2005 only included a strict priority based transmission selection mechanism for prioritized delivery of streams. This mechanism allows only the highest priority traffic to go first and was inadequate to fulfill hard latency guarantees.

Applications concerning live audio and video require low latency in the order of few milliseconds (maximum 2 ms from the source to destination) in order to achieve media synchronization (lip sync) [7]. Automotive applications involving time-critical control data require even lower latencies in the order of tens of microseconds (e.g. Engine control messages). So, identifying the worst case latency of the network is crucial to guarantee these hard latency requirements.

One of the solutions to achieve low latency in switched Ethernet is to segregate and minimize the interference of non-time-sensitive data traffic with the time-sensitive data traffic through prioritized traffic classification and to improve the traffic shaping mechanism for transmission selection. Traffic shapers, usually present at the egress port of a network element (switch or an end node) are mechanisms to select a specific data stream for transmission, based on different criteria of performance and latency guarantees of that stream. In other words, they prioritize and shape the data traffic from different domains to achieve latency and other QoS guarantees.

Each data traffic class has a specified bandwidth reservation limit and is associated with certain priority. The traffic shaping mechanism being an integral part of the Layer 2 (MAC Layer) of the network element is responsible for selecting the next data traffic class to be transmitted out of the network element based on rules imposed by the priority and bandwidth reservation of that traffic class. By defining and adjusting the rules for transmission selection, it is possible to avoid the interference of low priority traffic with the time-sensitive data traffic thereby guaranteeing the QoS for time-sensitive traffic.

As the standardization moved towards IEEE802.1AVB with a goal to achieve isochronous delivery of messages for audio and video streaming, a substandard IEEE802.1Qav was developed to include a transmission selection mechanism called Credit Based traffic shaper. The Credit Based traffic shaper was a critical part of the IEEE802.1AVB standard for Audio and Video bridging. The traffic shaping mechanism along with the IEEE802.1AS time synchronization standard guarantees QoS for real-time audio and video streams. Thereby, the IEEE802.1AVB enabled switched Ethernet to be used for automotive infotainment systems. A more detailed study of the IEEE802.1AVB standard, Credit Based Shaper and the QoS guarantees delivered by the standard are presented in Chapter 4.

To achieve further lower latencies over Ethernet and to enable a standardized switched Ethernet for industrial and automotive control networks, an enhancement of IEEE802.1AVB standard called IEEE802.1TSN is currently under standardization process. One of the features of IEEE802.1TSN is to introduce a new traffic shaping mechanisms to accommodate and guarantee deterministic end-to-end latency for time-sensitive data traffic. There are different suggestions and proposals from the members of the standardization committee for the new traffic shaping mechanism to be included in addition to the existing Credit Based shaper for audio and video data streams. The IEEE802.1TSN standard and the new traffic shaping mechanisms has a direct impact on the use of switched Ethernet for IVN backbone, as it includes specifications for the time-sensitive control data traffic from protocols like CAN and FlexRay. The IEEE802.1TSN standardization process and the new traffic shapers to improve the QoS of Ethernet for time-sensitive data traffic forms the basis for this thesis work.

1.4 Problem Statement

There are currently three different proposals for the traffic shaping mechanism to be included in the IEEE802.1TSN standard, namely the Time Aware shaper, Burst Limiting shaper and Peristaltic shaper.

Each of these three traffic shaper hardware mechanisms proposed has its own advantages and disadvantages in achieving the goal of very low latency for the time-sensitive control data traffic (100 µs over 5 hops)[8]. Other than the latency requirements for the control data traffic, the automotive industry imposes certain constraints for Ethernet IVN network elements in terms of flexibility to add new data streams, configuration complexity and hardware cost [9]. In order to fulfill these requirements, all the three traffic shaping mechanisms proposed for IEEE802.1TSN standardization process need to be studied, evaluated and compared for feasibility, performance and cost effectiveness.

1.5 Proposed Solution

A hardware implementation for evaluation and performance comparison of all the three traffic shapers is a costly experiment. So the approach is to build to a simulation model of the entire switched Ethernet network including models for switches and end-nodes where each of the traffic shapers can be plugged in as a separate Layer-2 component and evaluated for performance. This solution addresses the following problems

1. To evaluate whether the proposed traffic shapers really guarantee the end-to-end latency requirement of 100 µs over 5 hops for the time-critical control data traffic in a switched Ethernet network
2. To compare the compliance with the standard, configuration flexibility and hardware complexity in terms of memory of all the traffic shapers proposed
3. To benchmark the best traffic shaper for the automotive Ethernet IVN requirements
4. To provide a software simulation platform for designing and evaluating a new traffic shaper meeting the automotive Ethernet IVN requirements

1.6 Goals

The primary goal of this thesis work is to compare the three different traffic shaping mechanisms for the possibility of achieving the end-to-end latency requirements specified by IEEE802.1TSN standard [8] for time-sensitive control data traffic . Though there is no requirement for message delivery variations (jitter), the control data traffic is ideally intended to have a jitter close to zero. This thesis work aims to identify a traffic shaper with minimal possible jitter for the control data traffic.

In addition to this the thesis work aims to achieve the following goals

1. Flexibility in configuration of the transmission period of the Control Data Traffic ranging from 500µs to 1000ms as this is a requirement from the automotive industry for switched Ethernet based IVN [9]
2. Fair bandwidth utilization for all traffic classes as per their bandwidth reservation. In other words, the traffic shaping mechanism should adhere to the bandwidth reservations for all traffic classes as specified by the Stream Reservation Protocol of IEEE802.1TSN standard.
3. Minimal hardware impact in terms of memory and computational units on the network elements. It is essential to minimize hardware complexity and costs since the traffic shaping mechanism is a hardware part of the network elements (Switches and End nodes).
4. Implementation in compliance with or with minimal impact to the other evolving standards in the IEEE802.1TSN standardization process

Secondary Goal

1. Implementation of different traffic shapers as a plug and play module to the existing SystemC software model at NXP for ease of integration and future analysis

1.7 Methodology

The input for the literature study on the different traffic shaping mechanisms proposed for the IEEE802.1TSN standard is obtained from the IEEE standardization committee presentations, proposals and standard drafts. These proposals are collected, organized and a comparative study is made for benchmarking. Considering the implementation, a simulation model of the automotive Ethernet network according to the IEEE802.1AVB standards is already available at NXP in SystemC. This thesis work uses the existing model as a base framework to implement the benchmarked traffic shapers and to adapt it to the evolving IEEE802.1TSN standards. It is then used for simulative analyses to compare the performances of the three traffic shapers implemented.

The evaluation mechanism involves a validation phase and a testing phase. The validation phase includes the test scenario similar to the one presented in [10] where the authors analyze the impact of one of the traffic shapers (Time Aware Shaper) on the AVB data traffic class. A similar test setup is created and tested for the performance of all implemented traffic shapers. The theoretical maximum end-to-end latency is computed for the worst case behavior in all the three shapers. The simulation results are then compared with the theoretical values for validation. For the testing phase, based on the inputs from NXP, a realistic automotive network scenario is created and tested for latency, bandwidth utilization, memory and other QoS requirements. The results of all traffic shaping mechanisms are compared and the best feasible shaper is recommended based on the overall performance.

1.8 Contributions

This thesis work makes the following contributions

1. A compilation of descriptions of three different traffic shaping mechanisms proposed for the IEEE802.1TSN standard
2. A software simulation model of three different traffic shaping mechanisms to extend automotive Ethernet IVN to achieve QoS requirements for time-sensitive data traffic
3. A simulation environment compatible to both the existing IEEE802.1AVB standard and the evolving IEEE802.1TSN standard which facilitates the configuration of different network topologies and generation of different data traffic classes to analyze the behavior of the automotive IVN
4. A modular software implementation of the traffic shaping mechanisms in SystemC to allow the plug and play of different traffic shaping algorithms to the existing Ethernet network element model (Switch and End node).
5. Mathematical representation of the worst case end-to-end latency behavior of all the three traffic shapers under considered assumptions
6. An analysis and a comparative study of performances of three different traffic shaping mechanisms proposed by the IEEE802.1TSN standardization committee
7. A conclusion on the feasible traffic shaper(s) for switched Ethernet IVN

1.9 Thesis Structure

The remaining part of this thesis is organized as follows

Chapter 2 presents the State of the Art and the related works to improve the QoS in Ethernet to accommodate time-sensitive traffic

Chapter 3 gives an overview of the traffic classification and prioritizing features in the IEEE802.1Q standard to include the time-sensitive traffic

Chapter 4 introduces the IEEE802.1AVB standard and gives an overview of the sub-standards included

Chapter 5 explains in detail about the traffic shaping algorithms proposed in the IEEE802.1TSN standard

Chapter 6 explains the simulation environment, the tools used and the implementation of the three different traffic shaping algorithms

Chapter 7 explains the test scenarios for validation phase

Chapter 8 presents the improvements implemented to reduce the end-to-end latency for control data traffic

Chapter 9 includes the test case scenario and results obtained

Chapter 10 includes a detailed discussion on the performance of the three traffic shapers and suitable recommendations

Chapter 11 concludes the thesis work with Chapter 12 briefing the possible future work

2. State of the Art and Related Work

This chapter gives an overview of different approaches that facilitate the inclusion of real-time data traffic in switched Ethernet. The correlation between these approaches and the three traffic shaping mechanisms considered is highlighted. Finally, a list of related works in analyzing the individual traffic shaping mechanisms is presented.

2.1 State of the Art

The capabilities of switched Ethernet to accommodate real-time data traffic have been investigated extensively. Two main approaches include the synchronized time-triggered TDMA based medium access approach and the asynchronous event-triggered approach by over-provisioning and prioritizing. The example for the former technique is the TTEthernet (AS6802) [11] and for the later technique it is the IEEE802.1AVB standard with Credit Based Traffic Shaping [12].

TTEthernet offers guaranteed low latencies and finds application in aerospace and aviation industry. It requires static schedule configuration for sending time-triggered traffic in the link. The concept of static scheduling based on the arrival time of all the time-triggered traffic is adapted for the Time Aware traffic shaper [39]. But the problem with static scheduling is that it needs highly accurate synchronization (1µs accuracy) among the nodes and also it is not flexible to add new traffic since the schedule has to be modified at all nodes.

The concept of Flexible Time-Triggered-Switched Ethernet presented in [13] offers flexibility to configure the schedules of the Control Data Traffic dynamically by using a trigger message from a Master Switch in the network. But this concept has not been evaluated for ultra-low latency requirements specified by 802.1TSN. Also, adding this type of Master/Slave configuration in the network for configuring new streams has an impact on the existing sub-standards for Stream Reservation, namely 802.1Qat and 802.1Qcc and this has to be studied separately. For IEEE 802.1TSN compatibility, it is required that the work on traffic shaping should not induce major changes to these sub-standards.

On the other hand, the event-triggered approach from IEEE 802.1 AVB, i.e. the Credit Based shaping technique provisions the prioritized traffic based on the bandwidth reserved for them. This method offers flexibility to add new data traffic dynamically. But the intention of IEEE802.1AVB is to guarantee QoS only for isochronous audio/video streams. It does not include highly time-critical Control Data Traffic in in-vehicle network. The authors in [14] [15] include the time-critical control data traffic in AVB network by considering it as the highest priority traffic and re-arranging the priorities for existing AVB traffic. This method exploits the Credit Based traffic shaper to shape the control data traffic. The authors of [14] achieve a latency of 175 µs over 3 hops for control data traffic with a payload size of 92 bytes. These results show that the Credit Based shaper is not sufficient to guarantee the QoS requirements specified by IEEE802.1TSN [8]. The Burst Limiting shaper introduced in [16] uses the basic principle as that of the Credit Based shaping technique but with certain modifications to include the control data traffic.

Apart from these two main approaches, there are many other techniques available in the literature which tries to include the real-time data traffic in switched Ethernet networks. A concept called synchronous pacing used in Residential Ethernet [17] and IEEE1394 [18] where the common timeline is divided into

cycles of equal width and the frames are labeled with cycle numbers before transmission. The frame received in one cycle is transmitted in the successive cycle as quickly as possible. This Peristaltic shaper enhances this technique to deliver guaranteed end-to-end latencies for time-critical control data traffic.

The Virtual Clock approach proposed in [19] re-arranges the frames for transmission based on their arrival rate. This mechanism offers flexibility as the incoming frames are dynamically re-arranged. But the work focuses to deliver only average throughput and average queuing delay which is not sufficient for the control data traffic in automotive Ethernet IVN. Other possibilities include the use of real-time scheduling concepts like Earliest Deadline First for the re-ordering of frames in terms of their urgency to be transmitted. This approach called the Urgency based scheduler [20] is another potential traffic shaping technique considered for the IEEE802.1TSN standard. Since there is only very little information available regarding this shaper in the standardization process, it is not considered for this thesis work.

2.2 Related Work

Since all the three traffic shaping mechanisms are still under IEEE standardization, much of the related work deals with proving the performance of each of the shapers theoretically and using simulations. The presentations available from different members of the IEEE standardization committee mostly include theoretical arguments for the comparison of these three traffic shapers.

Boiger in [21] provides a comparative study of the Burst Limiting shaper and the Peristaltic shaper. The work illustrates the working of these two shapers under network overload situations. But it does not really give insights to the end-to-end latencies achieved for the control data traffic. It concludes that the Burst Limiting shaper should block the overload of control data traffic in order to protect the downstream bridges. Also it concludes that the Peristaltic shaper is not backward compatible to the Credit Based shaper in terms of implementation. This theoretical analysis provides sufficient insights for modeling the Burst Limiting shaper and the Peristaltic shaper.

Goetz in [22] compares the simulation results of Burst Limiting shaper and Time Aware shaper using an industrial network use case with eight bridges. The size of the control data traffic is varied and simulations are performed. The results show that for control data traffic with constant frame sizes of 64 bytes, the end-to-end latency in a Burst Limiting shaper varies between 80 µs to 90 µs. On the other hand, the latency remains constant at 80 µs for Time Aware shaper. With random size control data traffic (size varied from 64 bytes to 512 bytes), there was only a minimal difference in latencies between the Burst Limiting shaper (372 µs) and the Time Aware shaper (368 µs). However, this scenario assumed a transmission period of 250 µs for control data with constant frame size and 875 µs for the control data traffic with random frame size. This does not comply with the IEEE802.1TSN standard. Though this work includes only the best effort traffic in addition to the control data traffic for the simulations, it provides sufficient details for the configuration of the Burst Limiting shaper.

The presentations [23][24][25][26][27] provide illustrations and theoretical methods to compute the worst case end-to-end latency for a control data frame. Though these presentations concentrate on different scenarios and variable frame sizes, they offer valuable details for identifying the worst case behavior of each of the traffic shapers. These methods are adapted in computing the theoretical maximum values for end-to-end latencies in the validation phase of this project.

Meyer *et al.* in [10] present a network scenario to analyze the impact of the Time Aware shaper on AVB Class A traffic. The authors present a theoretical analysis for the maximum end-to-end latency of AVB Class A data traffic and compare their simulation results with the theoretical values. The results show that the size of the time slot for control data traffic and the spacing of time slots play a vital role in affecting the end-to-end latency of the AVB Class A data traffic. This work provides the needed the network scenario to simulate the worst case behavior of all the three traffic shapers.

Rahmani *et al.* in [29] propose a novel traffic shaping mechanism called Simple Traffic smoother for video traffic in in-vehicle communication. The goal is to space out the video traffic based on a predefined smoothing interval to avoid traffic bursts and to analyze the resource usage in terms of queue sizes. This work succeeds in achieving efficient resource usage for variable bit rate video traffic in a double star topology. The results prove that smoothing out bursts equally can improve the resource utilization. This work along with the presentation in [28] about performance tradeoffs to achieve end-to-end latency is considered as an input for creating the static schedule for control data traffic in Time Aware shaper.

3. Traffic Classification

The selection of particular data traffic for transmission depends on the priority associated with that traffic. So, it is essential to understand how data traffic from different domains is classified and prioritized before describing the traffic shapers for transmission selection. This chapter gives an overview of switched Ethernet specifications for prioritization and classification of different data traffic.

3.1 Prioritization

Figure 3.1 illustrates the structure of an Ethernet frame with VLAN tag field. IEEE802.1Q standard specifies VLAN tag as an optional field used to specify the Virtual LAN identification [30]. A VLAN is a logical sub-network defined via the switches. VLAN has properties which allows to

1. Limit the number of links used for a broadcast
2. Define different priorities to the traffic of different VLANs
3. Restrict access to portions of the network
4. Define different active topologies

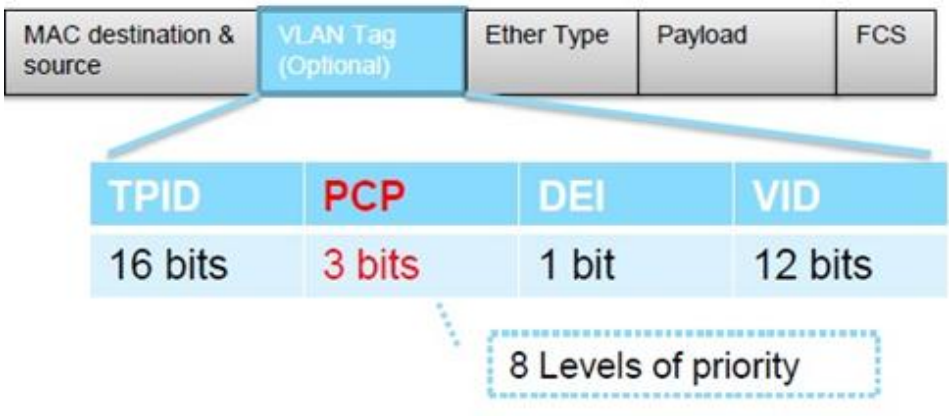


Figure 3.1 Ethernet Frame structure with VLAN Tag

A VLAN Tag is composed of a fixed set of fields as follows (extracted from the IEEE802.1Q standard [30])

TPID: Fixed by the standard, indicates to the switch that this frame has a VLAN tag and the switch has to read further to find the EtherType field

PCP: Priority-Code-Point which identifies the priority of the frame (0 to 7 with 0 as the lowest priority)

DEI: Drop eligible bit used by input mechanisms at the switches to mark frames that violate the configuration restrictions (e.g. maximum bandwidth allocated for a specific VLAN)

VID: Identifier for a VLAN (upto 4096 values but not all the switches necessarily support the whole range)

The PCP field values can be used to classify different traffic classes based on their QoS needs and map them to different priority levels.

3.2 Traffic Types and Priority mapping

Though it is complex to represent a full description of the QoS needs of the applications and network services with just 8 priority levels using the PCP, IEEE802.1Q gives a pragmatic way of traffic classification. It simplifies the classification requirements and specifies eight different traffic types as shown in the Table 3.1, which can be useful for traffic segregation based on their QoS requirements [30].

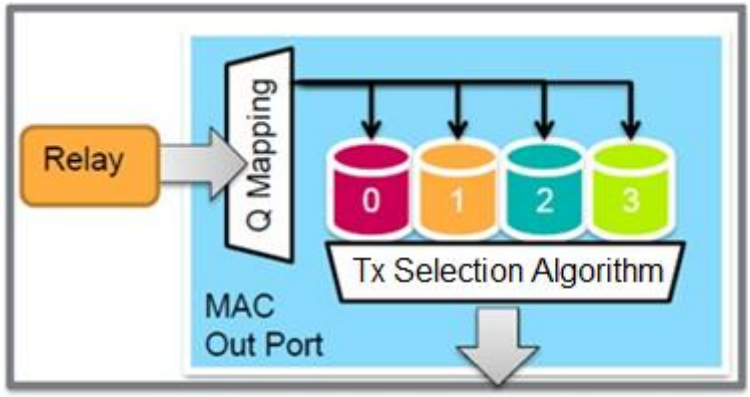
PCP	Priority	Acronym	Traffic Types
1	0 (lowest)	BK	Background
0 (default)	1	BE	Best Effort
2	2	EE	Excellent Effort
3	3	CA	Critical Applications
4	4	VI	Video, < 100ms latency and jitter
5	5	VO	Voice, < 10ms latency and jitter
6	6	IC	Internetwork Control
7	7 (highest)	NC	Network Control

Table 3.1 Priority mapping of different IEEE802.1Q recommended traffic types

If the network element supports only one output queue, then all the traffic is considered as the Best Effort traffic. So the default traffic is the Best Effort traffic. At the same time, the default PCP used by the nodes to communicate is 0. This is the reason for associating PCP value 0 to Best Effort traffic. But if Background traffic is available, it takes the lowest priority. So, the Best Effort traffic is always transmitted with a PCP value of 0 and it takes a higher priority value than Background traffic if the latter is available in the network.

Each output port of a network element (Switch and End nodes) supports a limited number of queues. Different traffic types can be grouped together to match the number of traffic class queues available in a network element. IEEE802.1Q standard suggests a list of group combinations for mapping traffic classes to the queues available [31].

For a better illustration, we assume a scenario where a network element has only four queues and there are 8 different traffic classes available in the network. These eight different traffic classes can be segregated by associating priorities based on the application requirements. Table 3.1 lists the priority and PCP suggestions provided by the IEEE802.1Q standard [31]. These traffic classes can be mapped to the available four queues as illustrated in the Figure 3.2



Frame PCP	0	1	2	3	4	5	6	7
Queue Number	0	0	1	1	2	2	3	3

Figure 3.2 Queue mapping for Traffic Types

This shows that the frames with priority 0 and 1, here the Background traffic type and the Best Effort traffic type are grouped together and mapped to the same queue (Queue 0).

3.3 Traffic Specifications in IEEE802.1AVB and IEEE802.1TSN

Traffic specification is a way to characterize the bandwidth that a data stream with a constant bit-rate can consume. The traffic specification of a data stream is defined by two elements

1. Maximum Frame Size: Defines the amount of data that is generated periodically (excluding overhead) by the stream
2. Minimum Frame Interval: Defines how often the data is being sent

The IEEE802.1AVB standard provides three different traffic class specifications based on their priorities and interval times namely Class A, Class B and Class BE. Class A has the highest priority, with Class B as the next priority and then followed by Class BE. Class BE has no interval of arrival as it is classified to include the legacy Ethernet traffic which can have random arrival intervals. Also Class BE has no latency constraints. There is no restriction on the maximum frame size of any traffic class. So the frame sizes can be up to the Ethernet standard restriction (MTU of 1500 bytes) for both 100 Mbits/s and 1Gbits/s links. Table 3.2 lists the traffic class specifications in IEEE802.1AVB standard and their QoS constraints in terms of end-to-end latency.

Traffic Class	Max. Frame Size (MTU)	Min. Frame Interval	Expected end-to-end Latency (7 hops)
Class A	1500 bytes	125 µs	2 ms
Class B	1500 bytes	250 µs	50 ms
Class BE	1500 bytes	-	-

Table 3.2 Traffic Specifications in IEEE802.1AVB. Class A covers the audio domain and Class B covers the video traffic

Class A traffic specification is intended to classify audio streams in the AVB network. For example, let us assume an audio source generating a 32 bit stereo audio stream at 48 kHz. Accounting for the left and right channel of the stereo, a sample of 64 bits is generated approximately every 20µs. This audio stream can be captured in an AVB Class A stream that generates a frame every 125µs. In this case, a single Class A frame can store 6 left and right audio samples. In a similar fashion, Class B is intended to classify video streams.

Moreover, in the professional live audio scenario, the maximum delay between a live audio source and its destination is limited to 10ms. The processing delays may use up to 8ms and there is only 2ms available for the network to transfer the data from the source to the destination [7] [32]. This explains the latency requirements for the AVB Class A stream.

As the IEEE802.1TSN was intended to service even more time-critical control applications in automotive network, an additional traffic class specification for the control data traffic namely Class CDT was introduced. Also frame size restrictions were imposed for other traffic classes to minimize the effect of their interference with the control data traffic. Class CDT has the highest priority followed by Class A, Class B and Class BE respectively. Table 3.3 lists the modified traffic specifications with their corresponding QoS constraints in terms of end-to-end latency. The frame sizes listed are for 100 Mbits/s links. These specifications are not yet part of the standard as the standard itself is in the process of development and they are only based on the general agreement of the standardization committee during the plenary meetings [8]. However, since they are more likely to get standardized; it serves as an input for this thesis work. The frame sizes are larger for 1Gbits/s links and the complete specification can be found in [8].

Traffic Class	Max. Frame Size (MTU)	Min. Frame Interval	Expected end-to-end Latency
Class CDT	128 bytes	500 µs	100 µs (for 5 hops)
Class A	256 bytes	125 µs	2 ms (for 7 hops)
Class B	256 bytes	250 µs	50 ms (for 7 hops)
Class BE	256 bytes	-	-

Table 3.3 Traffic Specifications in IEEE802.1TSN. A new data traffic class (CDT) for the time critical control data is added

4. IEEE 802.1AVB standard

The IEEE802.1AVB is the standard that includes many new protocols to enable switched Ethernet for time sensitive audio and video applications. It forms a base framework for the evolving IEEE802.1TSN standard to target more real-time applications. This chapter gives a brief introduction to the IEEE802.1AVB standard and some of the protocols involved in it.

The IEEE Audio/Video Bridging task group (AVB TG) [33] was created with responsibilities for developing standards to deliver low latency streaming services over IEEE 802 networks. The main goals for AVB TG were as follows [7]

1. To provide a network wide precision reference clock
2. To restrict the per hop network delays to a known small value
3. To limit the interference of the non-time-sensitive traffic for time-sensitive traffic

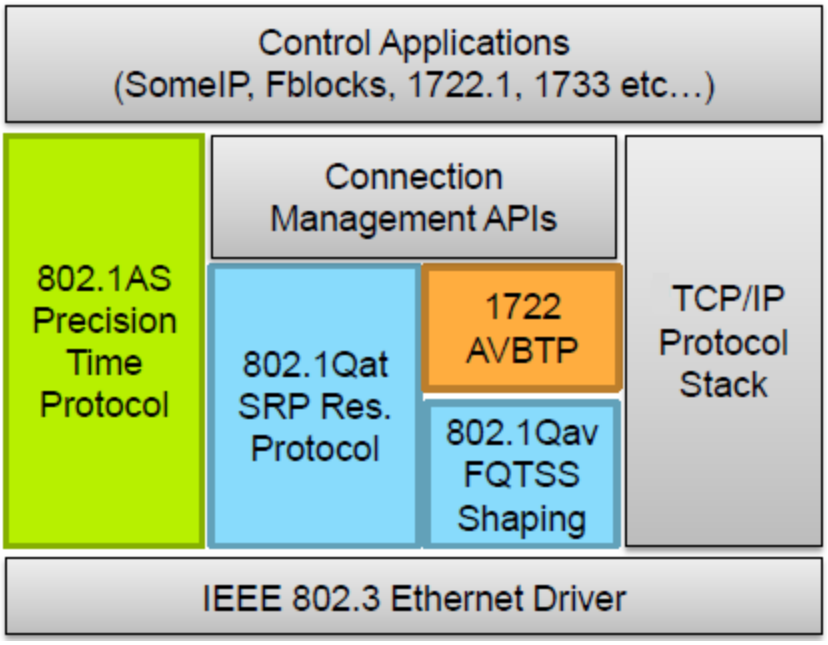


Figure 4.1 Overview of IEEE802.1AVB Standard

Based on the above goals, the task group specified different mechanisms which were based on the Medium Access Control (MAC) Layer to support guaranteed QoS in a switched Ethernet network [7]. The overview of the set of protocols specified in the IEEE802.1AVB standard compared to the legacy protocols is given in the Figure 4.1. These protocols are essential to guarantee a worst case end-to-end latency for audio and video streams. This chapter gives an overview of the different sub-standards which form the IEEE802.1AVB standard and the Chapter 4 explains the traffic classification and the QoS guarantees expected out of the AVB standard.

4.1 IEEE 802.1AS

The IEEE 802.1AS protocol [34] aims to achieve one of the crucial goals for time-sensitive communication which is to deliver precise timing and synchronization over all the nodes in the network. The standard aims to achieve accurate distributed timing with a worst case synchronization error of ±500ns in a seven hop AVB network.

The synchronization process is executed in two steps.

1. Selection of a Grand Master (GM) node. The Grand Master is a node with the best available source of time (e.g. atomic clock, GPS, etc.) among all the nodes of the network and is selected using a best master clock algorithm (BMCA). The synchronization of all other nodes is with respect to the Grand Master. The standard also provides plug and play capability with automatic GM selection and automatic reconfiguration in case of GM change or error.
2. Synchronization of the distributed nodes. The synchronization information is distributed to the network by using two message types namely the *Sync* and *Follow_Up* messages. The *Sync* message is synchronization information and the *Follow_Up* message is the correction value for the previous information to improve accuracy. The correction is performed at all intermediate and the end systems in the network.

IEEE802.1AS requires HW support of the MAC or PHY layers for time-stamping the incoming/outgoing frames but does not require the ability to change the local clock PLL configuration. It specifies the minimum requirements for the local clock of each node in the network. These minimum requirements are listed in the Table 4.1.

Fractional Frequency offset relative to Timing Application Interface (TAI)	+/- 100PPM
Max Frequency Drift	1PPM/sec
Time Measurement Granularity	<= 40ns (25MHz)
Jitter Generation	< 2ns peak to peak

Table 4.1 Minimum requirements for local clock

4.2 IEEE 802.1Qat

The IEEE802.1Qat sub-standard of IEEE802.1AVB specifies a signaling protocol named Stream Reservation Protocol (SRP) for end-to-end management of resource reservation in the AVB network [35]. This protocol allows dynamic registration or de-registration of data streams in the whole AVB network to allocate or release the bandwidth associated with that particular stream.

The registration process has two steps. In the first step, the sources of AVB content (Talkers) generate a *talker advertisement frame* with traffic specifications such as *StreamID, MaxFrameSize* and *MaxIntervalFrames*. The *talker advertisement* is a multicast communication independent of the AVB destination (Listener) and each of the intermediate switches in the network is able calculate the required bandwidth for the stream with this information. If a switch has enough resources, the frame is forwarded

without any modifications. If there are no sufficient resources available at a switch, the advertisement frame is modified to *talker failed frame* with failure information and is forwarded to all the listeners.

When the *talker advertisement frame* is received at the listener, the next step in the registration process begins. If the *talker advertisement* does not contain any failure information, the listener can decide to register and subscribe this particular stream. In this case, the listener generates a *listener ready frame* and transmits it to the talker in order to register to the specific data stream and to inform the intermediate switches along the path to allocate the required bandwidth for the same. If there is no sufficient resource at an intermediate switch, the frame is modified to *listener asking failed* and forwarded to the previous switches to inform about the insufficient bandwidth capacity.

The stream reservation is successful only if the bandwidth requirement of the stream is available on all the switches along the path from the talker to its listeners. The standard also specifies the procedure to handle failed *talker advertisements* and *listener registrations* due to the lack of bandwidth capacity in one or more intermediate switches in the network.

The de-registration of a data stream can be initiated by both talkers and listeners. A talker can de-register a stream at any time by transmitting the *de-register stream frame* and a listener can do this by transmitting a *de-register attach frame* in a similar procedure as that of registration. In this case, the switches along the path releases the bandwidth reserved at the ports for that particular stream.

4.3 IEEE 802.1Qav

The IEEE802.1Qav standard named "Forwarding and Queuing of Time Sensitive Streams" specifies mechanisms to deliver guarantees for real-time audio/video traffic (with bounded latency and jitter) [36]. The standard aims to achieve latency requirements as mentioned in Table 3.2 for both Class A and Class B AVB streams. The two main mechanisms provided by IEEE802.1Qav to achieve guaranteed QoS in a switched Ethernet network are explained in the following sub sections

		Number of Queues or Traffic Classes						
		2	**3**	**4**	**5**	**6**	**7**	**8**
Priority	**0 (Default)**	0	0	0	0	0	0	1
	1	0	0	0	0	0	0	0
	2 (Class B)	1	1	2	3	4	5	6
	3 (Class A)	1	2	3	4	5	6	7
	4	0	0	1	1	1	1	2
	5	0	0	1	1	1	2	3
	6	0	0	1	2	2	3	4
	7	0	0	1	2	3	4	5

Table 4.2 Recommended Priority mapping for AVB Classes

4.3.1 Priority re-mapping

The IEEE802.1Qav standard extends the priority mapping rules specified by the IEEE802.1Q standard (described in Chapter 3) to accommodate the AVB Class A and Class B traffic classes. Table 4.2 shows the IEEE802.1AVB recommended VLAN priority re-mapping for Class A and Class B with priority values 3 and 2 respectively depending on the total number of queues available. Each entry in the table indicates the queue index allocated to a particular traffic. The modified priority mapping for this thesis work is given in Table 4.3. Since this thesis work uses 8 queues per output port, the corresponding queue indices of Class CDT, Class A and Class B traffic are highlighted. It is to be noted that Class CDT has the highest priority and hence it is assigned the highest queue index. Class A is assigned a queue index of 6 and queue index 5 is assigned to Class B. Class BE with default priority uses the queue with index 1. Class CDT traffic is tagged with PCP value 3, Class A with 5 and Class B with 4.

		Number of Queues or Traffic Classes						
		2	**3**	**4**	**5**	**6**	**7**	**8**
Priority	**0 (Default)**	0	0	0	0	1	1	1
	1	0	0	0	0	0	0	0
	2	1	0	0	1	2	3	4
	3 (Class CDT)	1	2	3	4	5	6	7
	4 (Class B)	1	1	1	2	3	4	5
	5 (Class A)	0	1	2	3	4	5	6
	6	0	0	0	1	2	2	2
	7	0	0	0	1	2	2	3

Table 4.3 Priority mapping used this thesis work. Each output port have 8 queues. Table 4.2 is modified to include Class CDT traffic with highest priority. Class CDT, Class A and Class B are allocated to queues 7, 6 and 5 respectively

4.3.2 Transmission Selection Algorithms

IEEE802.1Qav specifies traffic shaping and scheduling mechanisms to enable the transmission of AVB and non-AVB legacy Ethernet frames. These mechanisms play a vital part in delivering guaranteed QoS for the AVB streams.

4.3.2.1 Strict Priority Selection

This is the default selection algorithm to select legacy Ethernet frames for transmission. This mechanism is supported by all network elements (Switches and End Nodes) and the frames are selected for transmission based on the priority values of the queues they are in. Figure 4.3 shows the organization of the transmission selection mechanism in an output port of an AVB network element.

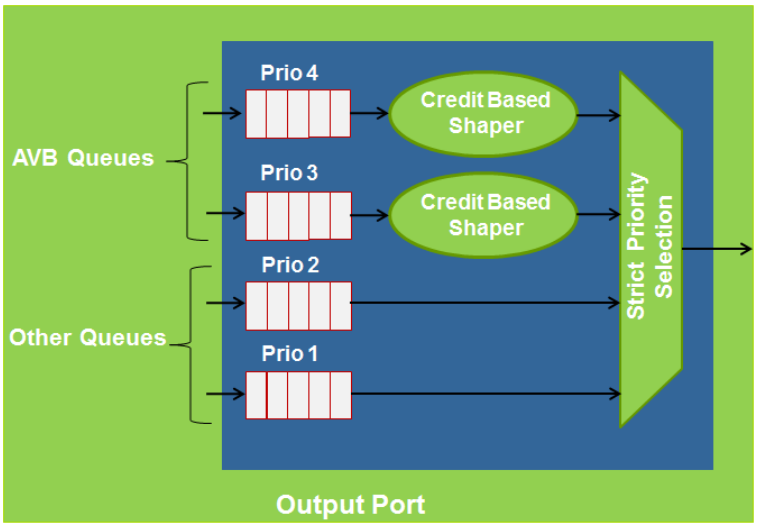


Figure 4.3 Transmission Selection at Output Port of a network element

4.3.2.2 Credit Based Shaper Selection

The credit based shaper algorithm is similar to the leaky bucket algorithm. The two AVB traffic classes namely Class A and Class B are organized in two separate queues which are associated with their corresponding VLAN priorities. In this case, Class A queue has higher priority than Class B queue. Each queue is associated with a credit based shaper. The algorithm defines credits in bits to be associated with each of the AVB traffic class queue. The transmission of a frame from an AVB class queue is only allowed when the credits associated with that particular traffic class is greater than or equal to zero, considering there are no conflicting frames. When an AVB frame is removed from the queue and transmitted, the credits associated with this AVB class queue are decreased at a rate called *sendSlope*. If the AVB frame is available in the queue but is waiting for frames from other queues to be transmitted, then the credit associated with this AVB Class queue is increased at a rate *idleSlope*. The algorithm limits the maximum and minimum number of credits that can be accumulated by using parameters *hiCredit* and *loCredit*.

The *idleSlope* for an AVB traffic class at which the credit is accumulated for that traffic class is defined as

$$idleSlope = reservedBytes/FrameIntervalTime$$

The *sendSlope* for an AVB traffic class at which the credit is decreased for that particular traffic class is defined as

$$sendSlope = idleSlope - portTransmitRate$$

The rules for calculating the credits are as follows

1. If there is positive credit but no AVB stream frame to transmit, the credit is set to zero.
2. During the transmission of an AVB stream frame the credit is reduced with the *sendSlope*
3. If the credit is negative and no AVB stream frame is in transmission, credit is accumulated with the *idleSlope* until zero credit is reached.

4. If there is an AVB stream frame in the queue but cannot be transmitted as a non AVB stream frame is in transmission, credit is accumulated with the *idleSlope*. In this case the credit accumulation is not limited to zero, also positive credit can be accumulated.

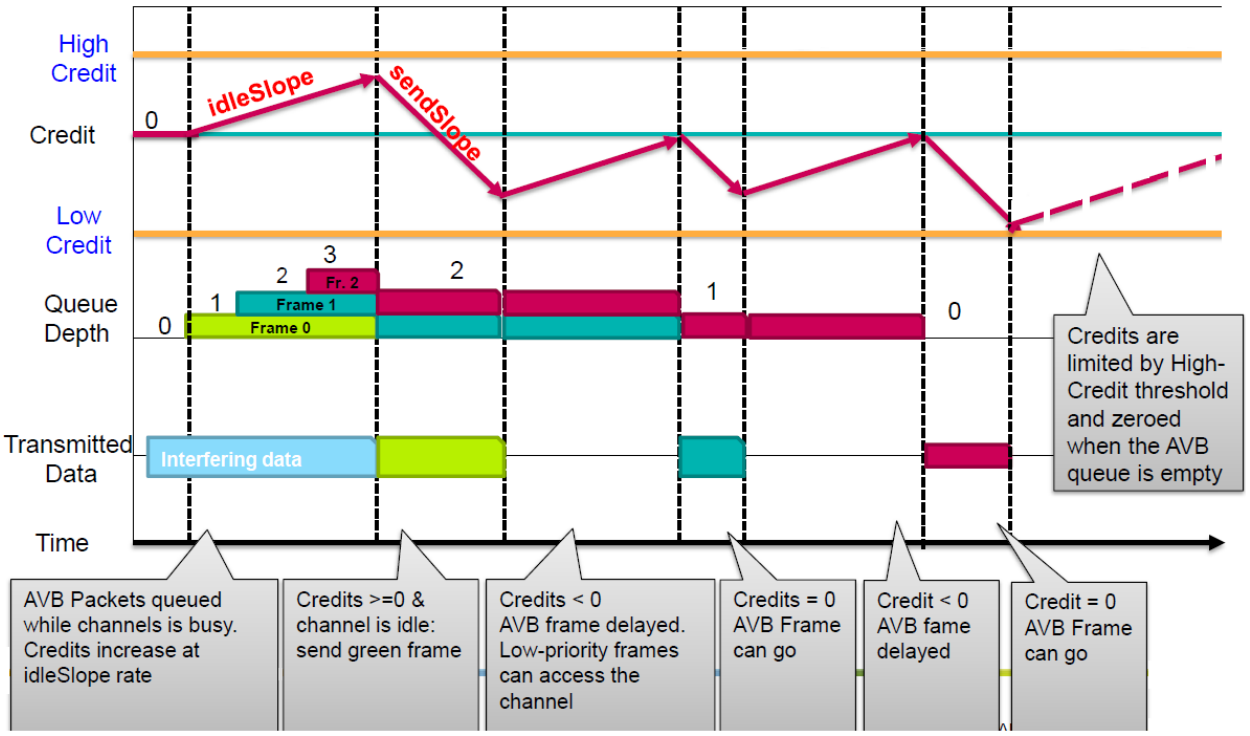


Figure 4.4 Credit Based Shaper operation for a particular AVB traffic class

The *hiCredit* is the maximum limit which the credit of an AVB traffic class can reach. This is limited by the size of the maximum interference for this traffic class. So *hiCredit* is calculated as follows

$$hiCredit = maxInterferenceSize \times (idleSlope/portTransmitRate)$$

The *loCredit* is the maximum burst limit that an AVB traffic class can reach once the frame is selected for transmission after it has accumulated credits during the idle time. So loCredit is calculated as

$$loCredit = maxClassFrameSize \times (sendSlope/portTransmitRate)$$

Figure 4.4 illustrates a scenario for the operation of credit based shaper algorithm. All the frames in the figure correspond to the same AVB traffic class in this example.

4.4 IEEE1722

IEEE1722 Audio/Video Transport Protocol (AVTP) [37] is a Layer-2 transport protocol for the end nodes in the AVB network with no impact on the switches. This software protocol specifies particular encapsulation of audio and video frames in order to meet the QoS requirements. Also IEEE1722 defines the following

1. The media types supported by the protocol (e.g. SD-DVCR, MPEG2 TS, Uncompressed Audio etc.)
2. The formats and encapsulations for the supported raw and compressed audio/video formats
3. The synchronization mechanisms at the end nodes including clock reconstruction features and latency normalization
4. A multicast address assignment protocol (MAAP) to define the ID for new streams

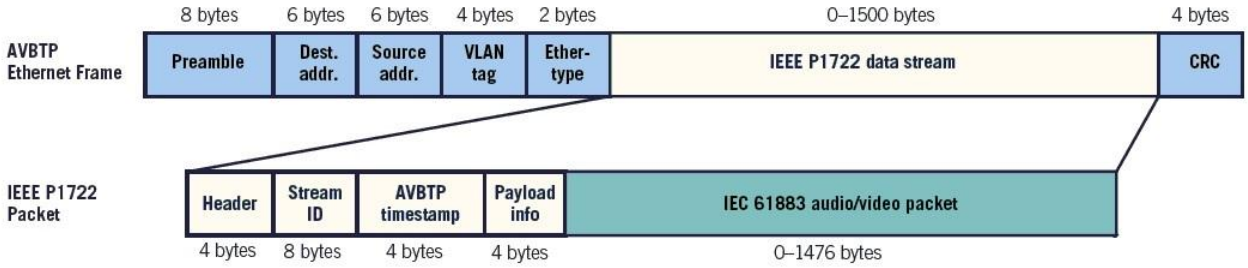


Figure 4.5 1722 AVBTP Ethernet Frame

The standard IEEE1722 frame specified as the AVBTP payload shown in Figure 4.5 is limited to a maximum size of 1476 bytes [37]. In addition to the payload, information like *Header*, *StreamID*, *AVTPTimestamp*, *GatewayInfo* and *PacketInfo* are encapsulated by an Ethernet frame. *StreamID* is used to identify and distinguish different streams. *AVTPTimestamp* contains the presentation time of a frame i.e. the sum of the time when the sample was presented to the AVTP at the talker side and a constant *MaxTransitTime* for the worst case network latency. The *MaxTransitTime* depends on the AVB traffic class of the given stream. The MaxTransitTime of Class A and Class BAVB streams are 2ms and 50ms respectively as derived from their latency constraints. The *PacketInfo* gives information about the payload and the *GatewayInfo* is reserved for the use of AVTP gateways.

5. Traffic Shaping Algorithms for Control Data Traffic

The Credit Based traffic shaping mechanism in IEEE802.1AVB needs to be improved to achieve better latencies for time critical Control Data Traffic (refer section 2.1). From Figure 4.4, it can be seen that there are two main methods by which the Credit Based Shaper can be improved.

1. Avoiding the spreading out of high priority traffic
2. Minimizing or avoiding interference from other traffic

This chapter details three different traffic shaping algorithms namely

1. Burst Limiting Shaper
2. Time Aware Shaper
3. Peristaltic Shaper

These three shapers are under consideration by the IEEE802.1TSN standardization committee to achieve very low latencies (100 µs over 5 hops) for Control Data Traffic. Two of these three algorithms namely Burst Limiting shaper and Time Aware shaper use the above mentioned techniques for minimizing end-to-end latency. The Peristaltic shaper aims to restrict the residence time of the CDT frames in an Ethernet switch to a fixed maximum limit.

5.1 Burst Limiting Shaper

The concept of the Burst Limiting shaper is to allow the high priority Control Data Traffic (CDT) to burst within a certain threshold, thereby avoiding the spreading out of high priority traffic. This algorithm uses similar concepts of credit accumulation and credit reduction techniques as that of the Credit Based shaper but with certain modifications to the rules for selecting a CDT frame for transmission [22]. As shown in Figure 5.1, the traffic classes are organized in separate queues which correspond to their VLAN priorities. Class CDT queue has the highest priority by default and it is associated with the Burst Limiting shaper. However, the priority associated with the CDT queue is changed dynamically by the Burst Limiting shaper. The AVB Class A has the next priority level followed by Class B and other traffic classes. The AVB queues are associated with the Credit Based shapers. The transmission selection mechanism here is a strict priority based transmission selection.

The Burst Limiting shaper algorithm specifies rules for changing the priority of CDT dynamically and for incrementing/decrementing credits to select the next frame to be transmitted.

A CDT frame is selected for transmission only

1. If the credit associated with the CDT queue are less than or equal to a maximum value (*max_level*)
2. If the *dynamic priority* of the CDT queue is higher than the priorities of all other queues

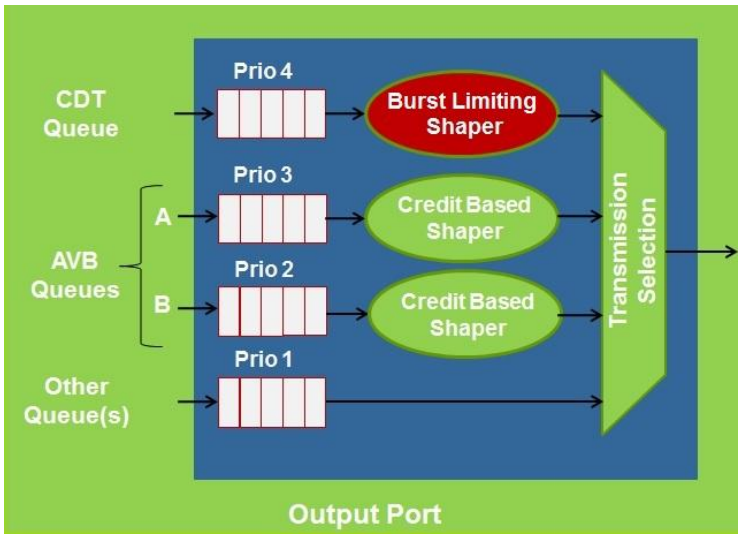


Figure 5.1 Output port with queue for CDT and Burst Limiting traffic shaper

The rules for incrementing or decrementing the credits are as follows [38]

1. The credits associated with the CDT queue are decremented at the rate of *idleslope* only when a frame from a non-CDT queue is selected for transmission or when there is no frame in any of the queues
2. The credit associated with the CDT queue are incremented at the rate of *sendslope* only when a CDT frame is being transmitted
3. The credit cannot increase more than *max_level* and cannot decrease below zero

These conditions introduce two new parameters namely *max_level* and *dynamic priority* for CDT which are to be defined. *max_level* is the maximum allowed credit limit for the CDT within a transmission period. This implies that *max_level* is determined by the fraction of total bandwidth allocated for CDT. The rate *idleslope* is a product of the bandwidth fraction for CDT and the transmission rate of the port (*portTxRate*).

$$idleslope = BandwidthFraction \text{ x } portTxRate \tag{5.1}$$

So, the max_level of the credits can be calculated as

$$max\_level = idleslope \text{ x } CTP + SafetyMargin \tag{5.2}$$

CTP in the above equation 5.2 denotes the Common Transmission Period of the CDT. The Common Transmission Period (*CTP*) is the minimum interval between successive CDT frames. The term "common" allows the flexibility to have multiple CDT traffic with different transmission periods to be grouped to have the same priority. In this case, the *CTP* is the "least common multiple" of all the transmission periods. The *CTP* for CDT according to the IEEE 802.1TSN assumptions is 500 µs [8]. The *SafetyMargin* is a tolerance limit can be used to increase the *max_level* threshold to tolerate overload situations within a transmission period.

The value of the credit determines the priority of the CDT queue. The CDT queue has the highest priority until the credit reaches *max_level*. The credit is incremented at the rate of *sendslope* as the CDT frames are being sent. The *sendslope* depends on the sum of bandwidth allocated to all traffic other than CDT. So the *sendslope* is given by,

$$sendslope = portTxRate - idleslope \quad (5.3)$$

When the credit value reaches *max_level*, the priority of the CDT queue is changed to the lowest of all the queues. This allows the traffic from other queues to be selected for transmission. The credit is decreased when the frames from non-CDT queues are transmitted and as it reaches the *resume_level*, the priority of the CDT queue is restored to the highest level. The *resume_level* is a fixed percentage of *max_level* for e.g. 10%.

$$resume\_level = 0.01 \times max\_level \quad (5.4)$$

By this mechanism, the CDT traffic is given the highest priority when the amount of CDT traffic is within the limits. At the same time, the bandwidth fraction allocated to a specific data traffic class is also respected.

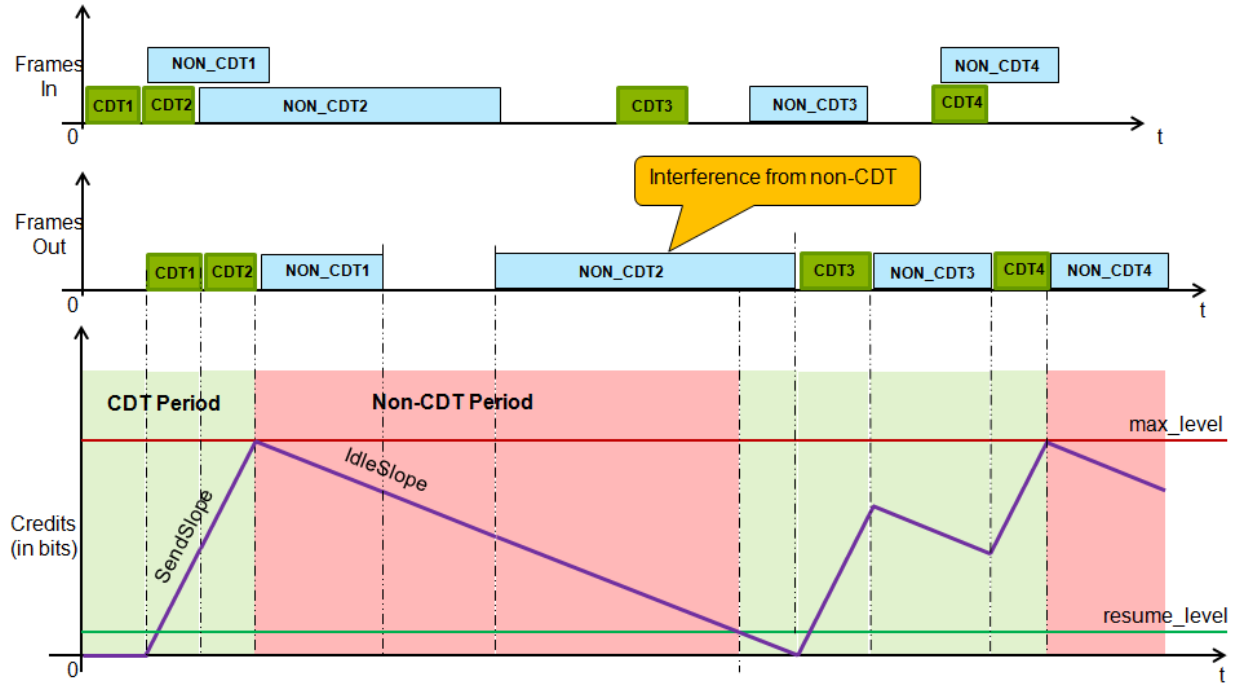


Figure 5.2 Operation of Burst Limiting Traffic Shaping Algorithm in an output port

The scenario shown in Figure 5.2 illustrates the operation of Burst Limiting shaper in an Ethernet switch. The Frames-In represents the arrival of frames in the input port of the switch targeting the same output port and the Frames-Out represents the frames that are being transmitted in that output port. Frames are numbered in the order they arrive and also named according to their traffic class. It is to be noted that the frames are selected for transmission only if they are completely received. The *CDTPeriod* and *Non-CDTPeriod* shaded regions represent the regions in the timeline where the priority of the CDT queue is

the highest and the lowest respectively. Also it is to be noted that the non-CDT frames can be transmitted in the *CDT-Period* and vice versa if there is no higher priority frame available to be transmitted.

5.2 Time Aware Shaper

The Time Aware shaper avoids interference from other data traffic to achieve low latency for CDT. The principle is to precisely know the time of arrival of every control data stream and to block the traffic from all other queues during this time to allow the CDT to be selected for transmission [39]. This gives rise to a schedule of time intervals which determines when the CDT queue can be opened or closed for transmission. So the Time Aware shaper is a mechanism which blocks certain queues based on the schedule to allow data traffic from other queues. An example schedule is shown in the bottom of Figure 5.3 with CDT and non-CDT time slots.

The blocking mechanism is implemented by associating each traffic queue of the output port with a time aware transmission gate [39]. These gates are triggered by a gate driver. The gate driver takes the schedule as its input and generates an open or close trigger for all the gates at corresponding time intervals in the schedule. By this, the entire schedule transforms to a *GateEventList*. Each entry in the *GateEventList* is called a *GateListEntry* and has two fields namely

1. *timeInterval* : Specifies the width of the time slot
2. *gateEvent* : Specifies a set of bits with each bit representing an open or close event for a particular queue (1 for open, 0 for close)

A frame from the CDT queue is selected for transmission only if the transmission gate of the CDT queue is open. On the other hand, a frame in a non-CDT traffic class queue is available for transmission only

1. If the transmission gate for that queue is in the open state and
2. If there is sufficient time available to transmit the entirety of that frame before the next gate-close event

The second condition above completely avoids the interference of the non-CDT frame in the CDT time slot. Figure 5.3 shows the Time Aware traffic shaper at an output port with different queues, GateEventList, gate driver and individual gates. The protocol specifies a maximum limit of 127 to the number of entries in the *GateEventList* [39]. The last entry in the *GateEventList* is always a *repeat* event. The *repeat* signifies the end of the present cycle and starts the schedule from the beginning for the next cycle. A cycle represents a complete CDT transmission period. So, it becomes necessary to know the schedule of arrival of all CDT streams for at least one transmission period. This schedule of *GateEventList* is created offline for all the network elements (switches and end nodes) in the network. These distributed schedule configurations among all the network nodes require highly accurate time synchronization. This is provided by the IEEE802.1AS protocol [34] which is explained in section 4.4.

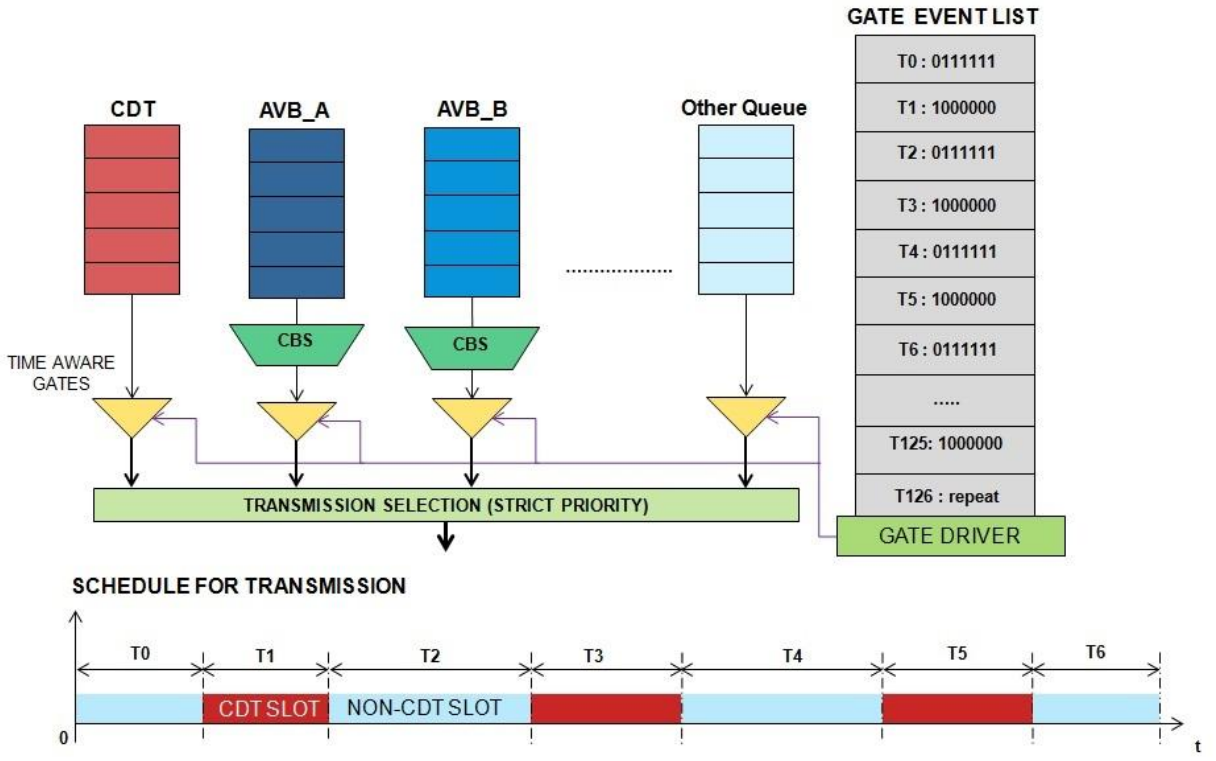


Figure 5.3 Time Aware Shaper and the schedule for transmission at an output port

It is to be noted that the AVB class traffic (both Class A and Class B) are already associated to a Credit Based shaper and the gates of the Time Aware shaper governs the selection of the traffic from the AVB queue only after it is being shaped by Credit Based shaping technique.

5.3 Peristaltic Shaper

The main factor in reducing the end-to-end latency for CDT frames is to minimize their residence time in the Ethernet switches. The Peristaltic shaper aims to achieve this by specifying rules to restrict the residence time of a CDT frame in a switch to a fixed maximum limit. The Peristaltic shaper works by dividing the time line into odd and even peristaltic phases of equal widths [40]. A CDT frame received in an odd phase is selected for transmission in the successive even phase and vice versa. This requires the frames to be marked with a *phase label* when they are received. Further, this implies that the residence time is dependent on the width of the peristaltic phase. The width of the peristaltic phase (*peristaltic cycle time*) is determined based on the required maximum end-to-end latency of a CDT frame and the maximum size of the CDT frame. The IEEE802.1TSN requirement specifies 100µs end-to-end latency for CDT frames over 5 hops. This gives a maximum residence time for a CDT frame in a switch to be 20µs neglecting the propagation delay. So each peristaltic phase is fixed to be 20µs wide.

The traffic classes are organized in separate queues which correspond to their VLAN priorities. A frame in a CDT queue is selected for transmission only if the following conditions are true [41]

1. The *peristaltic transmission phase* label of the frame is different from the *peristaltic reception phase* label of that frame

The following internal parameters are associated with each queue that supports the peristaltic shaper algorithm:

1. *Peristaltic reception phase label (pRphase)*: The phase value to be assigned to frames on reception. This parameter takes the value *even* or *odd*, and is determined as described in equation 5.5.
2. *Peristaltic transmission phase label (pTphase)*: The current peristaltic transmission phase. This parameter takes the value *even* or *odd*, and is computed as in equation 5.5 when the frame is available in the CDT queue for transmission.
3. *Peristaltic cycle time (pCycle)*: The period of time, in nanoseconds, representing the width of a peristaltic phase.

The value of *pRphase* and *pTphase* is computed in the same manner and is derived from the current time known to the system.

$$pRphase\ or\ pTphase = \left\lfloor \frac{current\_time}{pCycle} \right\rfloor modulo\ 2 \tag{5.5}$$

Figure 5.4 shows the working of Peristaltic shaper with odd and even phases. The time instances t1, t3 and t5 represent the time of transmission of the CDT frame at the Talker and switches respectively. The instances t2, t4 and t6 represent the time of reception of the CDT frame at the switches and the Listener respectively. The *pRphase* for this frame is computed based on this time at the respective switches. The time interval t2 to t3 and t4 to t5 represent the residence time of the CDT frame at Switch1 and Switch2 respectively.

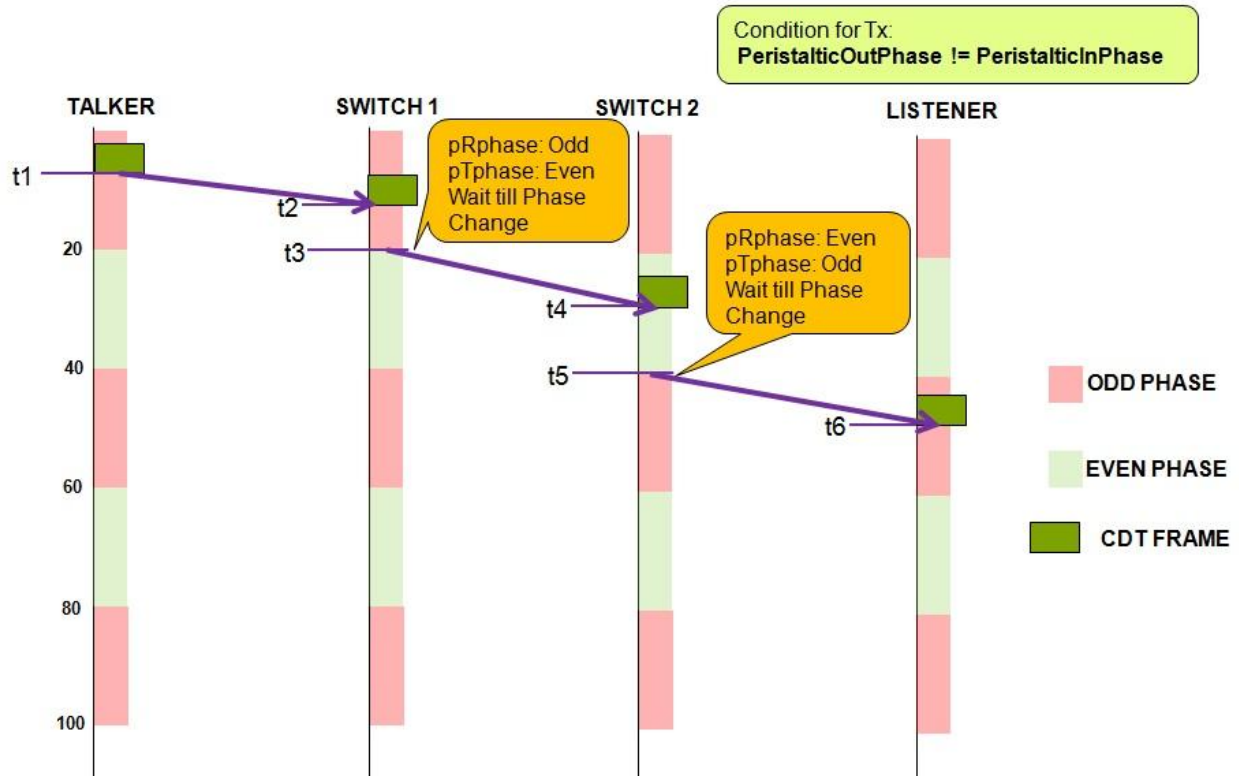


Figure 5.4 A CDT frame being transmitted from a talker to listener over 3 hops according to Peristaltic Shaper Algorithm

It is to be noted that, for simplicity only the CDT frame is considered in this example and the interferences from the non-CDT frame are not portrayed. The worst case scenario for Peristaltic shaper with interference from non-CDT traffic is detailed in section 7.2.3

6. Simulation Environment

A simulator developed using the SystemC language for modeling Ethernet IVN according to the IEEE802.1AVB standard is already in use at NXP. This thesis work extends the existing simulator according to the IEEE802.1TSN standard, to include and analyze the traffic shapers for Control Data Traffic. This chapter provides an overview of the SystemC language, the software architecture of the Ethernet IVN network elements (switches and End nodes) along with the assumptions and restrictions of the simulated model. It also explains the implementation details and the modularity achieved for the plug-and-play of all the three traffic shapers considered for the Control Data Traffic.

6.1 Overview of SystemC

As the complexity of the Systems-on-chip (SoCs) increases, using the gate-level or RTL simulation to characterize and explore an SoC for all the use case scenarios is not feasible. So there is a need to model systems at a higher abstraction level. SystemC is a system design and modeling language that includes both hardware and software concepts. It is created as a C++ library to facilitate functional modeling of systems. It offers a top-down approach for modeling a system starting from a simulator-level with flexibility to inject the concepts of hardware (e.g. clock) wherever required. The major features implemented in SystemC include [42]

1. Time model
2. Concurrency model
3. Module hierarchy to manage structure and connectivity
4. Communications management between concurrent units of execution
5. Data types for modeling digital logic and fixed-point arithmetic

Also SystemC provides flexibility to model the hardware using different time abstractions as follows

1. Un-timed model (UT): Time is not modeled at all
2. Loosely timed model (LT): Time is updated depending on events
3. Approximated time model (AT): Each event is characterized by a specific delay
4. Register Transfer Logic model (RTL): Clock is modeled
5. Pin and Cycle Accurate model : Exact clock period is considered

This implies that a SystemC design can be potentially translated to a physical implementation through its RTL model. As SystemC is an extension of the C++ library, it is compatible with the existing C++ software tools and it is completely free of charge. The above features enable SystemC to be best suited for hardware-software co-design.

Figure 6.1 shows the architecture of the SystemC language. It can be seen that the SystemC includes a simulation kernel. The simulation kernel in SystemC is implemented as a cooperative multitasking model. The kernel merely co-ordinates and orchestrates the swapping of various simulation processes and advances the simulation time accordingly to achieve concurrency [42].

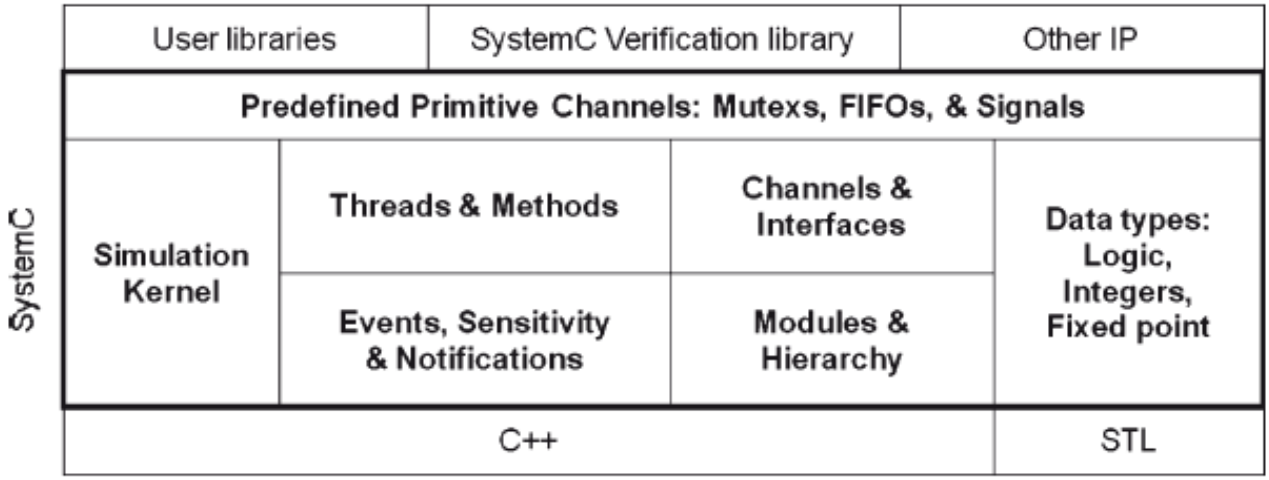


Figure 6.1 Architecture of the SystemC language [42]

6.2 SystemC Components

This section explains some of the crucial components of SystemC language that are essential for modeling the Ethernet IVN.

6.2.1 SystemC Module

Hardware designs typically contain hierarchical blocks to reduce complexity. SystemC encapsulates each hierarchical block as a module and separates it from the interfaces to other modules. Modules are classes that inherit from the SystemC SC_MODULE base class. Modules are registered with the simulation kernel (during the execution of the constructor for the SC_MODULE class) and may contain other sub-modules, simulation processes, and channels and ports for connectivity with other modules.

6.2.2 Simulation Processes

Simulation processes in SystemC are simply member functions of SC_MODULE classes that are registered with the simulation kernel. Because the simulation kernel is the only caller of these member functions, they need no arguments and they return no value. An SC_MODULE class can also have processes that are not executed by the simulation kernel. These are normal C++ member functions which are invoked within the simulation processes inside the module class. There are two types of simulation processes in SystemC which are invoked by the simulation kernel.

SC_METHOD

An SC_METHOD is a member function of an SC_MODULE class where time does not pass between the invocation and return of the function. SC_METHOD is called repeatedly by the simulation kernel.

SC_THREAD

An SC_THREAD is also a member function of an SC_MODULE class and is invoked by the simulation kernel. SC_THREAD simulation process differs from the SC_METHOD process in two

aspects. First, an SC_METHOD is invoked multiple times by the kernel but the SC_THREAD is invoked only once by the kernel. Second, an SC_THREAD has the option to suspend itself and potentially allow time to pass before continuing the execution. In this aspect, an SC_THREAD is similar to a traditional software thread execution.

Both SC_METHOD and SC_THREAD are the basic units of concurrent execution in SystemC and they are never invoked by the user. In the Ethernet IVN model implementation, only the SC_THREAD processes are used with *wait* delays for suspending them. SC_METHODs are not used.

6.2.3 Port

Processes need to communicate with other processes in the same module and in other modules. Ports are classes that allow processes to access channels that are out of the module boundaries. In a simplistic way, ports are like pins of a hardware component. An *sc_port* is a class template inheriting from a SystemC interface. This implies that ports connect one module to the other through a standard SystemC interface which could be, for example, a FIFO interface. SystemC ports are always defined within the module class definition.

6.2.4 Export

Export serves the similar purpose to that of a SystemC port. But the idea of *sc_export* is to move the communication channel inside the defining module for hiding some of the connectivity details and using export externally just like a channel. The usage of *sc_export* gives the designer to exert control over the communication interface by sharing only selected communication channels publicly with other modules. That is, *sc_export* is used to hide the interfaces within modules. Usually an *sc_port* of one module is connected to an *sc_export* of other module for communication.

6.2.5 TLM Interfaces

SystemC is continuously evolving with addition of new libraries such as

1. TLM: Transaction Level Modeling Library
2. SCV: SystemC Verification Library
3. SystemC-AMS: SystemC for Analog and Mixed-Signal design

The TLM library includes definition for standard interfaces and provides a set of APIs to use them. It also defines a set of channels that implement above these interfaces. The TLM standard library has two revisions namely TLM1.0 and TLM2.0. Only TLM1.0 is used in the Ethernet IVN implementation.

One of the important TLM1.0 interface used in the Ethernet IVN model is the TLM *bidirectional blocking interface* called the *tlm_transport* interface. This is a request-response interface used for communication between processes (here SC_THREADs). The invoking thread is blocked until the response is received. A *tlm_transport* is a function implemented within a module and a pointer to this function is passed to the invoking modules as a template argument via the *sc_port-sc_export* connection. Thus the *tlm_transport* is implemented as a hidden interface within a module and connected to other modules via *sc_port-sc_export* connections. Figure 6.2 shows communication between two SystemC

modules where an SC_THREAD in SC_MODULE1 invokes a *tlm_transport* interface inside SC_MODULE2 through a *sc_port-sc_export* connection.

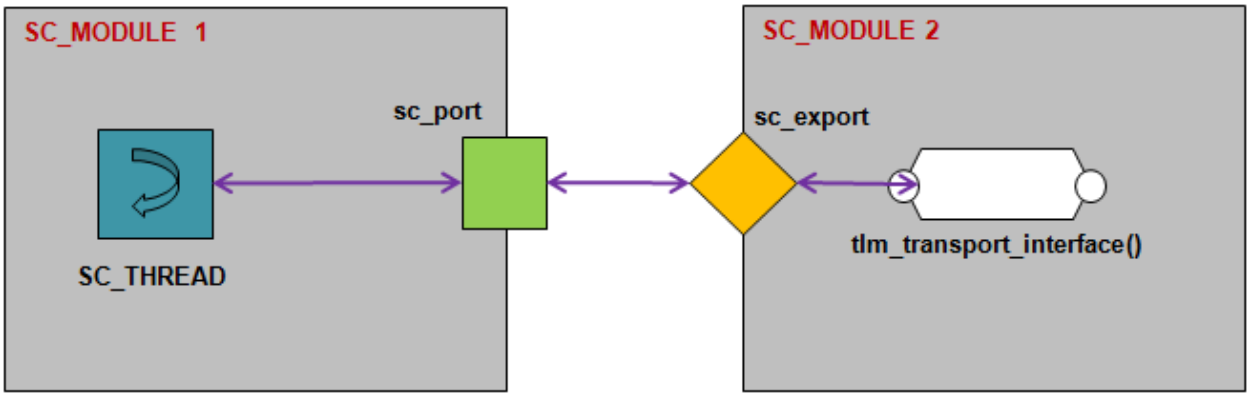


Figure 6.2 An example of communication between two SC_MODULEs through sc_port-sc_export connection

6.3 Software Architecture

This section explains the software architecture and organization of the Ethernet IVN network elements. Figure 6.3 explains the layered software architecture inside an end-node (ECU) and a switch. Each block is implemented as a SystemC module class with the module specifications in a *.h* file and module definitions in a *.cpp* file. The modules are connected using *sc_port-sc_export* connection through *tlm_fifo* interfaces or *tlm_transport* interfaces as explained in the section 6.2.

6.3.1 End Node (ECU)

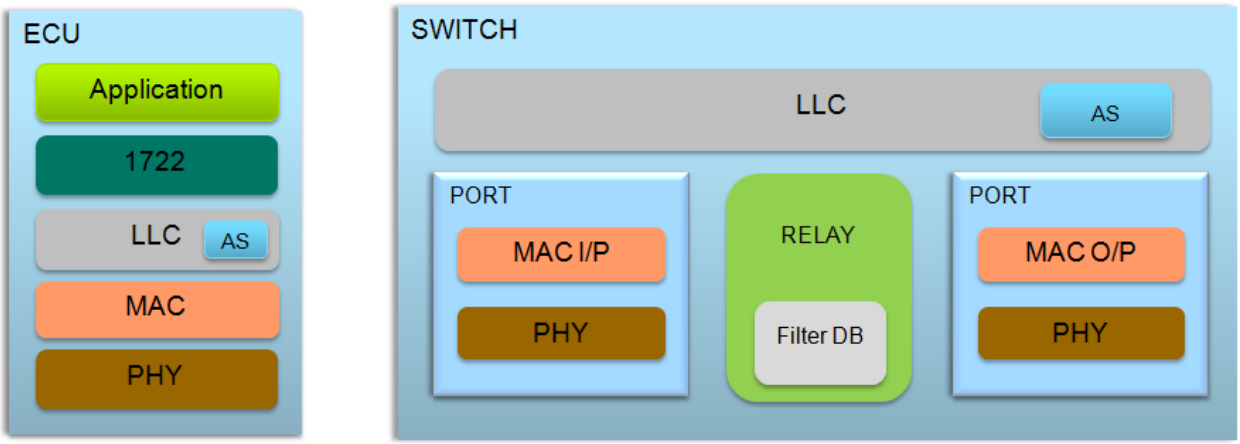
The end-node can serve as a Talker (source) or a Listener (destination) or both. Figure 6.3(a) explains the layered software architecture in an end-node (ECU). The Application layer consists of a source and a sink process threads to generate or receive the configured traffic. More details about Application layer and traffic generation is provided in section 6.5. The 1722 block represents the IEEE1722 AVTP Layer-2 transport layer. The transport layer packs the sending frames and unpacks the receiving frames according to the IEEE1722 transport protocol as explained in section 4.4. The LLC block represents the Logical Link Control layer which implements the control operations for the switched Ethernet network. Also the LLC layer includes the IEEE802.1AS time synchrnoization protocol implementation as a sub-module. The ECU has only one port and thereby consists of only one instance of Medium Access Control (MAC) layer and Physical (PHY) layer.

6.3.2 Switch

The Ethernet switch on the other hand, does not contain the application layer and transport layer. As the function of the switch is to receive the frames and relay it on to the other network elements, it does not require an application layer. But to receive the network control and synchronization frames and to act upon it, the switch has a Logical Link Control (LLC) layer with the IEEE802.1AS sychronization support. Further more to deliver the incoming frames to the corresponding output ports in order to route them correctly towards their destination, the Ethernet switch includes a Relay module. The Relay module is composed of two processes namely the *learn process* and *forward process*. The *learn process* reads the

source address of the incoming frame and adds it to the address table in the FilterDB sub-module if the traffic is already registered for the switch in this path (This traffic registration is to be done dynamically by the Stream Reservation protocol according to IEEE802.1AVB. But since Stream Reservation is not implemented in the model, the traffic/stream registration is done statically before the simulation begins). The *forward process* polls the input ports for incoming frames, checks the FilterDB for its registration and forwards it to the corresponding output port.

Figure 6.3(b) shows an Ethernet switch with only two ports, one input port and one output port. But a switch can have many input ports and output ports. The input ports are constructed with one instance of the MAC I/P module and a PHY module. The output ports are constructed with one instance of the MAC O/P module and a PHY module. But there is only one instance of the Relay module which is connected to multiple input ports and output ports.



(a) (b)

Figure 6.3 Layered Software Architecture in an Ethernet (a) End-Node (ECU) and (b) Switch

The software architecture explained in this section is completely implemented in SystemC and it exists as a fully functional simulator model at NXP to simulate and evaluate the IEEE802.1AVB standard. This thesis work re-organizes the MAC layer in both ECU and switch to include the new traffic shapers for Control Data Traffic.

6.4 Traffic Shaper Modules

Traffic shapers are integral components of each output port in an Ethernet network element (both switch and end-node). They are associated to the prioritized output queues and aid in selecting a queue from which the next frame is to be transmitted. This queue selection function is a part of the MAC layer and so the traffic shapers are organized as sub-modules of the MAC layer. Figure 6.4 shows the visualization of the traffic shaper module in a network element. The figure also portrays the modular organization of the Credit Based Shaper (CBS) for the AVB Class queues and the option to select one of the three shapers for the Control Data Traffic (CDT) queues. The acronyms within the Traffic Shaper module represents Burst Limiting Shaper (BLS), Time Aware Shaper (TAS) and the Peristaltic Shaper (PS).

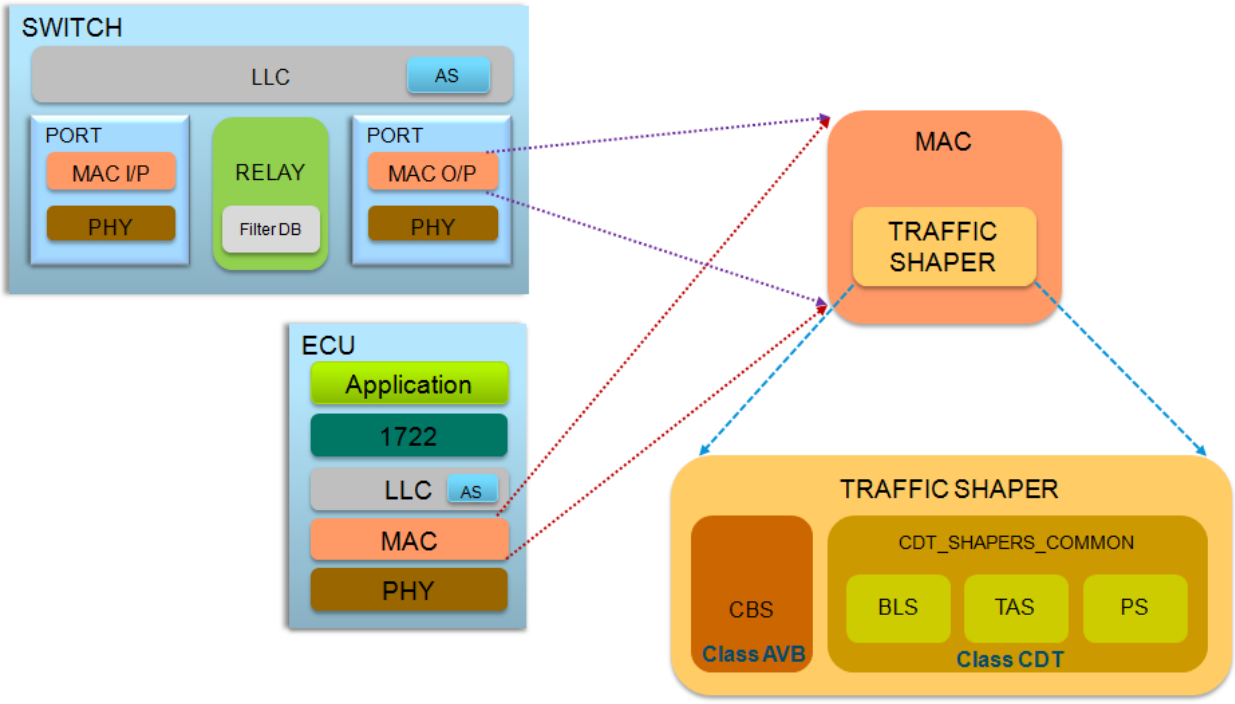


Figure 6.4 Modular organization of the Traffic Shaper module and its sub-modules

The Traffic Shaper module is a part of the MAC layer of the ECU and the MAC O/P layer of the switch. The Traffic Shaper module is composed of two sub-modules namely CBS and CDT_SHAPERS_COMMON. The CBS module includes the Credit Based shaping algorithm for the AVB class output queues (for both Class A and Class B). The sub-module CDT_SHAPERS_COMMON acts a composite module containing all the three traffic shapers for CDT. But, based on the configuration, only one of the three CDT shaper modules is selected for compilation and other two shapers are ignored. The selection is implemented as a compiler switch and the value of the compiler switch determines which shaper to be selected for simulation. This provides a plug-and-play option for selecting a shaper for CDT queue. The CDT_SHAPERS_COMMON module can further be extended to include more shapers for CDT. Therefore CDT_SHAPERS_COMMON module provides a platform for the design, development and evaluation of new traffic shapers for CDT.

Figure 6.5 shows the abstract detail of the SystemC level implementation of the Traffic Shaper module within the MAC layer. The *q_selection_thread* in the MAC layer is a simulation process which performs the functions of selecting a queue based on the traffic shaping algorithms, retrieving a frame from that queue and forwarding it to the PHY layer for transmitting. The *q_selection_thread* communicates to the Traffic Shaper module via an *sc_port-sc_export* connection for selecting a queue. The bidirectional blocking interface, *tlm_transport_if* implements the queue selection algorithm inside the Traffic Shaper module. The *tlm_transport_if* interface receives the present status of all the queues as input from the *q_selection_thread* and responds with a queue index. The *tlm_transport_if* interface switches between the CBS or CDT traffic shaping algorithms for queue selection based on the present status of the output queues. It is to be noted that the CDT_SHAPERS_COMMON module contains interface to only one of the three CDT shapers based on the configuration (all the three shapers are shown here). Once the queue index is returned to the *q_selection_thread*, a frame from the corresponding queue is retrieved and

forwarded to the PHY layer for transmitting. The traffic shaping sub-modules CBS, BLS, TAS and PS include the functions for corresponding traffic shaping algorithms to aid the queue selection process.

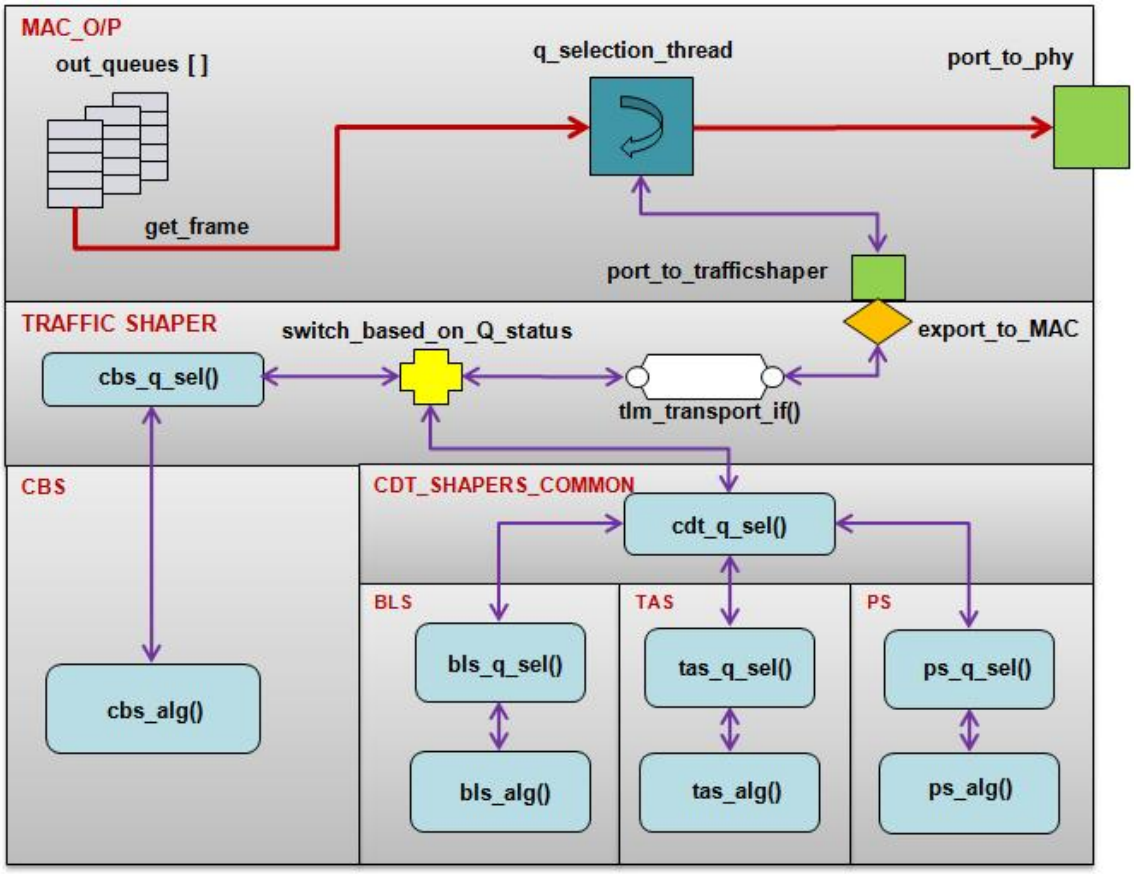


Figure 6.5 Abstract detail of SystemC level implementation of Traffic Shapers

6.5 Application Module and Traffic generation

The Application module in the ECU includes two simulation processes namely the *source_thread* and a *sink_thread*. Based on the configuration of the ECU as a Talker (source) or a Listener (sink), the corresponding simulation process (*source_thread* or *sink_thread* or both) is enabled. The *source_thread* is completely independent from that of the *sink_thread*, i.e the application cannot act on the receiving frames and reply with a response. The *source_thread* merely sends the frames and *sink_thread* merely receives the frames. The application can be configured to send only one type of data traffic (either Class CDT, Class A, Class B or Class BE traffic) but can receive all types of data traffic. The following sub-sections explains the details of generating different data traffic types

CLASS AVB

The *source_thread* generates Class AVB traffic by using a <*class, frame size*> configuration pair. The AVB traffic class is specified by the frame generation interval that is fixed by the IEEE802.1TSN standard (125us for Class A, 250us for Class B) while the frame size is a simulation input parameter that remains constant during the whole simulation.

CLASS BE

The *source_thread* has two options to generate Class BE traffic. The first option is to generate the Class BE traffic randomly at an interval of 2ms to 3ms. The second option is to use the <*load, frame size*> parameter. The network load is measured in Mbps and is processed by the Application module to derive the *inter arrival time* between the frames with the following equations

$$\lambda = \frac{frame\ size}{load} \tag{6.1}$$

$$inter\ arrival\ time = exp\ (\lambda) \tag{6.2}$$

Because Class BE traffic is usually does not have a standard *inter arrival time,* the model adds a degree of randomness to the traffic, by using λ as mean-value for an exponential distribution that is called after the generation of each frame to derive the interval until the next frame. The exponential distribution with mean value λ is used to generate the *Poisson Process which is* typically used to model random traffic.

CLASS CDT

The Class CDT traffic is generated based on a fixed schedule. The schedule includes the time of transmission of a frame for each CDT Talker in the network. It repeats over a period of 500 µs which is the inter arrival time of CDT frames. This schedule table for generating the CDT traffic at an interval of 500 µs is provided as an input to the *source_thread* of each of the Talker in the network and the *source_thread* generates the CDT frames based on the scheduled time. The schedule coordinates the transmission of CDT frames among all the Talkers, such that only one of them can transmit at a time. The advantage of having a coordinated scheduled transmission of CDT over uncoordinated transmission is explained in [24]. Also, as the Time Aware shaper expects a scheduled transmission of CDT, the same scheduled CDT traffic generation method is used for all the CDT shapers for a fair comparison.

6.6 Time Abstraction Model and Delays

As briefed in the section 6.1, SystemC offers different time abstractions to model the system behaviour. The Ethernet IVN model uses the Approximated-Time abstraction provided by SystemC for the simulation. By approximated-time abstraction, each event in the system is characterized by a specific delay. So each SystemC module in the Ethernet IVN model is characterized by a specific delay. These delays are based on the values found in the literature[43]. Figure 6.6 shows all the delay components

involved in transmitting a frame from the Talker application to the Listener application over a switch in the Ethernet IVN model.

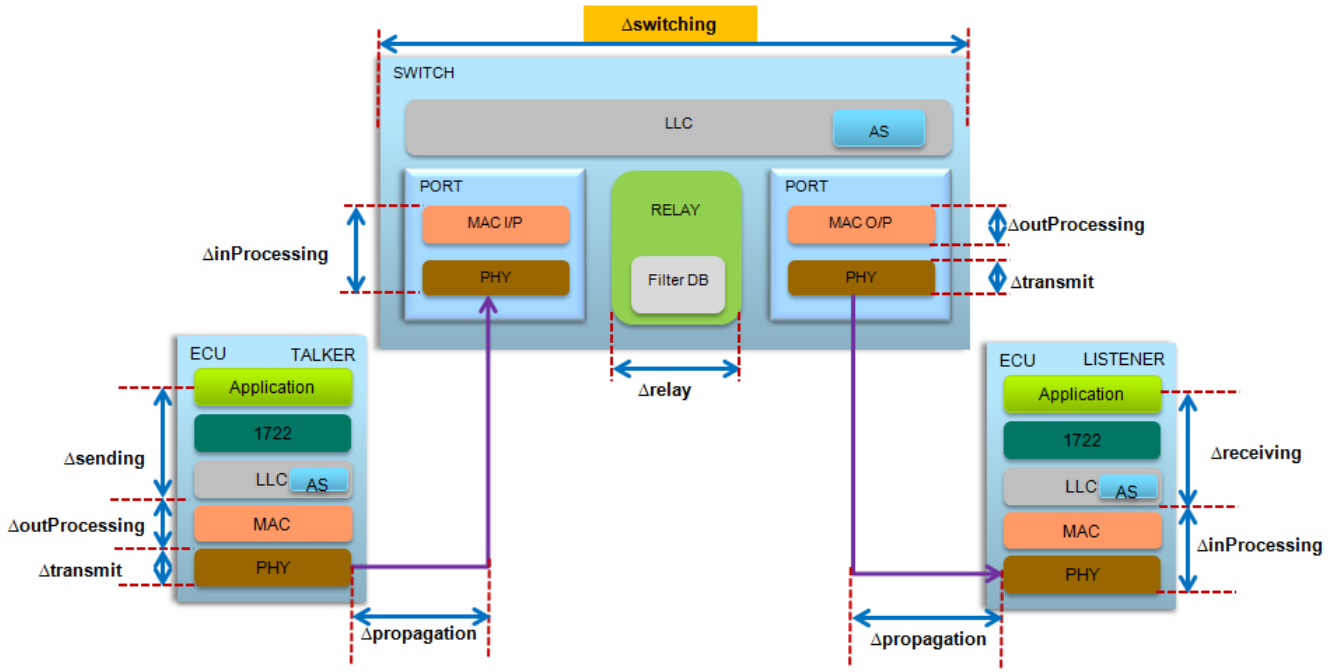


Figure 6.6 Delays involved in the Ethernet IVN simulation model

By this model, the end-to-end delay of a frame over two hops is the sum of all these delays. The delay components can be grouped into talker side delay, switching delay, listener side delay, transmission delay and propagation delay.

$$\Delta talker = \Delta sending + \Delta outProcessing \tag{6.3}$$

Equation 6.3 expresses the delay involved in the Talker modules to deliver the frame from the Application to the PHY to be transmitted over the link. The Δ*sending* delay comprises of the time required to pack a frame according to the IEEE1722 AVTP and the delay at LLC to process and transfer it to the MAC. This delay is fixed as 40 ns in total in the Ethernet IVN model. The Δ*outProcessing* is the delay at the MAC layer for selecting the frame from the queue and transmitting. Since the Application module is capable of sending one type of data traffic, this delay is a constant value of 1µs. So the Talker side delay is the sum of these two delays which is 1.04 µs in total.

The Δ*transmit* represents the time required to push the frame to the link. This delay depends on the link speed (port data rate) and the frame size. The model assumes a link speed of 100Mbps for all the links in the network. At 100Mbps, the time required for the PHY to forward 1 byte of data to the link is 0.08 µs. So the formula to compute Δ*transmit* for a given frame size in bytes is given by the equation 6.4

$$\Delta transmit = framesize\ in\ bytes \text{ x } 0.08\ \mu s \tag{6.4}$$

The switch receives the frame, checks the filter database for the registration of the frame to forward it to the correct output port and the output port modules pushes the frame out to the corresponding link. The reception side Δ*inProcessing* delay is at the MAC input module of the switch. This delay characterizes the time involved in collecting the frames from the PHY buffer and also for input filtering. This delay is fixed as 1 µs in the Ethernet IVN model. The Relay module in the switch includes two delay components; a fixed delay (1 µs) for the relay process to bridge the frame from input port to the output port and a variable delay to poll the frame entry in the filter database and check for the frame registration. The polling delay is variable as it depends on the number of entries in the filtering database. The time for polling the filter database once is fixed as 40 ns. The delay at the output port of the switch is same as that in the port of a Talker. The entire switching delay Δ*switching* is given by the equation 6.5 below

$$\Delta switching = \Delta inProcessing + \Delta relay + \Delta outProcessing \tag{6.5}$$

The minimum Δ*switching* delay for a frame at a switch is measured as 3.062 µs with only one data traffic passing through the switch. The Listener side delay includes the time to get the frame from the PHY and pass it on to the listener application. Equation 6.6 expresses the Listener side delay and it is fixed as 1.02 µs in total for the Ethernet IVN model.

$$\Delta listener = \Delta inProcessing + \Delta receiving \tag{6.6}$$

The propagation delay, Δ*propagation* is the time taken for a frame to travel in the physical link. This time varies on the length of the link. But for simplicity, it is assumed to be 0.538 µs for each of the link in the network [23] [24]. With the equations 6.3 to 6.6, the end-to-end latency for a frame over a two hop network can be expressed as

$$\begin{aligned}\Delta endtoend_{2hop} = {} & \Delta talker + \Delta transmit + \Delta propagation + \Delta switching \\ & + \Delta transmit + \Delta propagation + \Delta listener\end{aligned} \tag{6.7}$$

This equation can be re-arranged as follows

$$\begin{aligned}\Delta endtoend_{2hop} = {} & \Delta talker + 2 \text{ x } (\Delta transmit + \Delta propagation) \\ & + \Delta switching + \Delta listener\end{aligned} \tag{6.8}$$

From equation 6.8, a generalized equation to express the end-to-end delay over any *n* number of hops can be derived as

$$\begin{aligned}\Delta endtoend_{nhop} = {} & \Delta talker + n \text{ x } (\Delta transmit + \Delta propagation) \\ & + (n-1)\, \Delta switching + \Delta listener\end{aligned} \tag{6.9}$$

The fixed values used in the model for these delays can be used to calculate the theoretical end-to-end latency for a frame. The sum of Δ*transmit* and Δ*propagation* is termed as Δ*traversal* for the use of calculations in the following chapters. The variable parameters in this equation are the Δ*transmit* which depends on the frame size and the Δ*switching* or residence time of a frame in a switch. The residence time of a frame in a switch depends on the interfering traffic at the switch and the traffic shaping algorithm.

6.7 Assumptions and Restrictions of the simulation model

The Ethernet IVN model does not implement all the protocols involved in the IEEE802.1TSN standard. For those protocols that are not implemented, valid assumptions or static configurations are used in the model. Also, the Ethernet switch model and the ECU model have certain limitations as compared to the real hardware implementations. This section describes some of the important assumptions and restrictions of the Ethernet IVN simulator.

6.7.1 Assumptions

1. The IEEE802.1AS module is not completely implemented and the synchronization messages are not exchanged between the network elements. All network elements see a common simulation time and are assumed to be perfectly synchronized
2. The transmission of Control Data traffic is coordinated and scheduled such that no two CDT streams interfere with each other
3. All the Talkers are assumed to send data within their allocated bandwidth limits and no Talker misbehaves
4. The data transmission rate at all the links is 100Mbps
5. The messages transmitted are always delivered

6.7.2 Restrictions

1. Stream Reservation Protocol is not implemented. Therefore stream reservation is static and configured before simulation. There is no possibility to dynamically add or remove new data traffic (streams) during the execution of the simulation
2. The transmission of network management frames or the network control frames are not implemented
3. The Talker application can be configured for only one type of data traffic (either Class A, Class B, Class CDT or Class BE)
4. The switches are implemented as store-and-forward switches and the frame can be selected for transmission only if the complete frame is available in the output queue
5. Frame pre-emption is not implemented as the IEEE802.1TSN standard for frame pre-emption is still under discussion and not complete
6. The delays used are chosen by averaging the numbers found in the literature and may change in the real product
7. The SystemC code is not intended to be a perfect reference model for RTL development

7. Validation

The traffic shaping algorithms for CDT have a direct impact on the end-to-end latency of a CDT frame (refer equation 6.9). In other words, the end-to-end latency of the CDT frame is a key performance indicator for a traffic shaping algorithm. To verify and compare the end-to-end latency performance of all the three traffic shaping algorithms for CDT, it is essential that these algorithms are validated for their functionality. This chapter explains the validation methodology, experimental setup and argumentations, with supporting results for verifying the functionality of all the three traffic shaping algorithms for CDT.

7.1 Methodology

One of the approaches to validate the functionality of the traffic shapers for the Ethernet IVN model is to replicate a network scenario from an already validated simulator and compare the results. Since all the three traffic shaping algorithms are relatively new for Ethernet IVN, there exists only very few works against which the implementation can be compared and validated. Moreover, most of the works focus only on the Time Aware Shaping mechanism [10] [14] [15]. This makes the validation of the other two shaping algorithms, viz. Burst Limiting algorithm and Peristaltic algorithm difficult. So a different approach for validation of all the three CDT traffic shapers is required.

The step-by-step process of the methodology followed in this thesis work is given below.

1. Identification of the worst case behavior of all the three traffic shapers for end-to-end latency. This step benchmarks the worst case functional behavior of each of the traffic shapers.
2. Theoretical calculation of end-to-end latency of the CDT frame for these worst cases using the delays involved in equation 6.9.
3. Selection of a common network simulation scenario (topology and data traffic) for all the three shapers.
4. Simulation of this network scenario with all the three traffic shapers.
5. Comparison of simulation results for end-to-end latency with that of theoretical calculations.

The goal is to prove that simulation results for end-to-end latency stay within the theoretical limits obtained through calculations. To arrive at the theoretical calculations for end-to-end latency, it is necessary to understand the worst case behavior of all the three traffic shapers. The following section illustrates the worst case behavior of all the three traffic shapers with examples.

7.2 Worst Case behavior for end-to-end latency

The main goal of all the three traffic shaping algorithms for CDT is to reduce the end-to-end latency. The maximum end-to-end latency for a CDT frame depends on the residence time of that frame inside each switch on the path towards its destination. The residence time inside a switch or the switching delay, $\Delta switching$ is determined by the equation 6.5 (repeated below)

$$\Delta switching = \Delta inProcessing + \Delta relay + \Delta outProcessing \quad (6.5)$$

In this equation, Δ*inProcessing* is constant as the input ports receive the frames one after the other and there is no interference with other incoming frames. The parameter Δ*relay* depends on the amount of traffic associated with this switch since the polling time of the filtering process in the relay module varies, i.e. more the traffic, more the time to check the filter database. The switched Ethernet standard imposes a limitation for the overall switching delay for the switch hardware [43]. The Δ*relay* can be fixed to a theoretical maximum value based on this constraint for the overall switching delay. In the following sections, the sum of Δ*inProcessing* and Δ*relay* is considered as bridging delay Δ*bridging*.

$$\Delta bridging = \Delta inProcessing + \Delta relay \tag{7.1}$$

The delay Δ*outProcessing* of a CDT frame at output ports is influenced by the interfering traffic and time to select the CDT frame for transmission. This indirectly depends on the capability of the traffic shaping algorithm at the output port to avoid the interference from other data traffic and to select the CDT frame for transmission. So, the worst case behavior of the traffic shaper at the output port of each switch affects the residence time of the CDT frame. This worst case behavior at each switch contributes to the maximum end-to-end latency of the CDT frame.

7.2.1 Worst case end-to-end latency in a Time Aware Shaper

The Time Aware shaping algorithm described in section 6.2 allows the CDT frames to be transmitted in dedicated time slots (CDT slots). The algorithm avoids the interference from other data traffic completely by imposing a condition to check that the size of the frame to be transmitted in the non-CDT slot is not big enough to interfere with the start of the next CDT time slot. So, the CDT time slot is completely free from interference and therefore the worst case for Time Aware shaper is not caused by the interfering traffic. This implies that the Time Aware shaper delivers a fixed end-to-end latency for the CDT frames regardless of the presence of other data traffic.

The worst case for the Time Aware shaper is caused by scheduling. The simplistic schedule configuration for Time Aware shaper is to use the same schedule in all the nodes of the network. Assuming that all the nodes are perfectly synchronized, the starts and ends of CDT slots are also perfectly synchronized and happen at the same time at all nodes. Figure 7.1 shows such a synchronized schedule for a two hop network with a Talker sending a CDT frame to a Listener over a switch. The time line is shown for all the three nodes in the network. The red bands in the time line represent the CDT slots and the blue bands represent the non-CDT slot. Let us consider a scenario where the Talker is transmitting a CDT frame at time t1 in the CDT slot. This frame traverses the link and reaches the switch at time t2. It happens that the time t2 just misses the CDT slot and is at the beginning of a non-CDT slot. The CDT frame received cannot be transmitted in the non-CDT slot. So apart from the bridging delay, Δ*bridging*, there is an additional wait (Δ*schedule*) for the CDT frame inside the switch until time t4 where the next CDT slot begins. The frame is transmitted out of the switch at t4 and reaches the listener at time t5. The worst case can occur at all the switches in case of multiple hops.

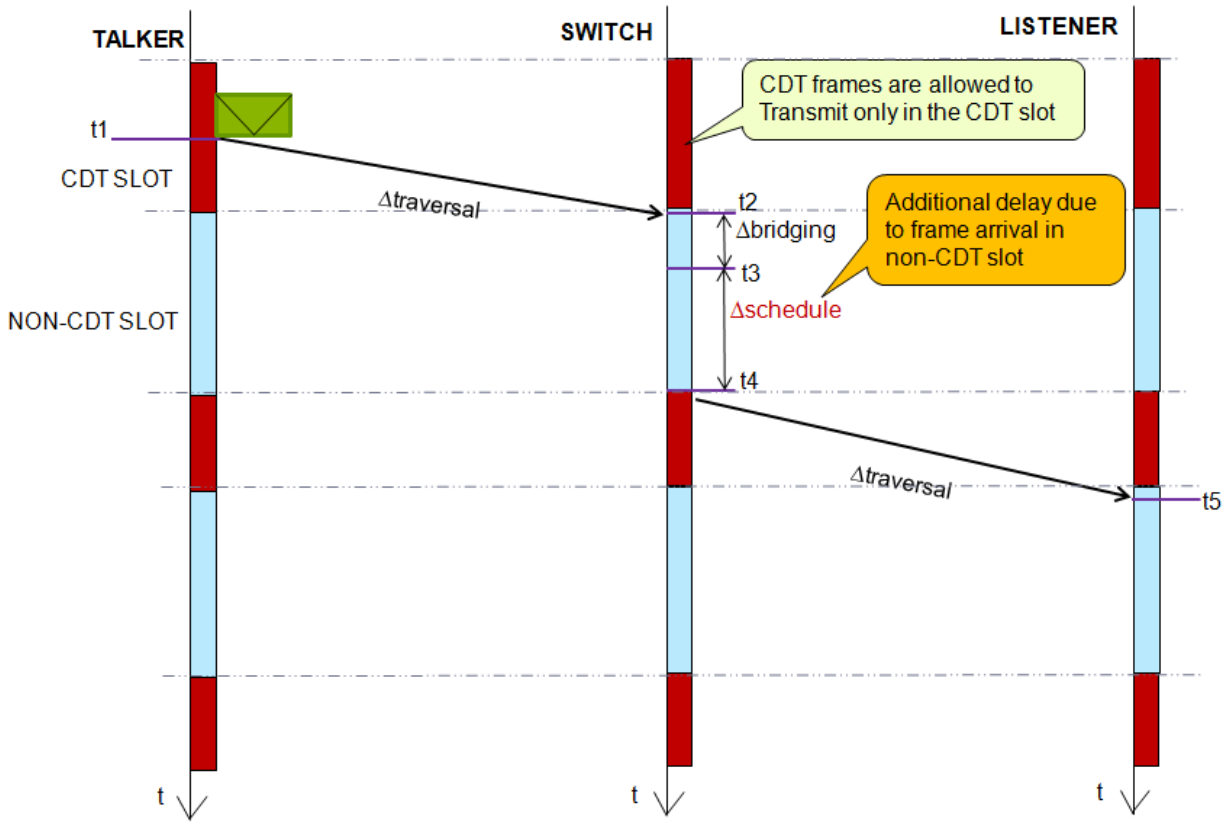


Figure 7.1 Worst Case for Time Aware shaper in a two hop scenario

The maximum residence time for a CDT frame in a switch using Time Aware shaper is given by the equation 7.2 below

$$\Delta maxSwitching_{TAS} = \Delta bridging + \Delta schedule \quad (7.2)$$

So using equation 7.1 and the explanation above, the equation 6.9 for worst case end-to-end latency for a CDT frame in a network can be redefined for Time Aware shaper as follows

$$\Delta max\_endtoend_{TAS\_nhop} = \Delta talker + n \text{ x } (\Delta transmit + \Delta propagation) + \sum_{i=1}^{n-1} (\Delta maxSwitching_{TAS})_i + \Delta listener \quad (7.3)$$

7.2.2 Worst case end-to-end latency in a Burst Limiting shaper

The Burst Limiting shaper described in section 5.2 admits the CDT frame if the credit for the CDT queue is less than a maximum limit. Unlike the Time Aware shaper, there are no dedicated time slots for the CDT frames and there is a possibility of interference with other data traffic. This interference affects the maximum residence time of the CDT frame inside a switch.

The worst case for the Burst limiting shaper is shown in Figure 7.4. The figure shows the working of the Burst Limiting shaper at the output port in a switch. The FramesIn captures the incoming frames and the FramesOut represents the frames that are being transmitted in the output link. It is to be noted that a frame

cannot be selected for transmission until it is completely received. The scenario in Figure 7.4 shows that the credit for the CDT queue is zero which is less than *max_level*. So when a CDT frame is received, it is eligible to be selected for transmission. But when a CDT frame is completely received in the CDT queue at time t1, a non-CDT frame is already using the link. It happens that the non-CDT frame is selected for transmission just before t1. Now, the CDT frame has to wait in the CDT queue until time t2, where the non-CDT frame is completely transmitted and the link is free again. In other words, the blockage time or the interference time, Δ*interference* depends on the size of the interfering non-CDT. During this time, the credit associated with the CDT queue decreases at the rate of *idleslope* and is always less than *max_level*. In this figure the credit remains at zero as the credit cannot be decreased below zero. The credit increases at the rate of *sendslope* when the CDT frame is being transmitted.

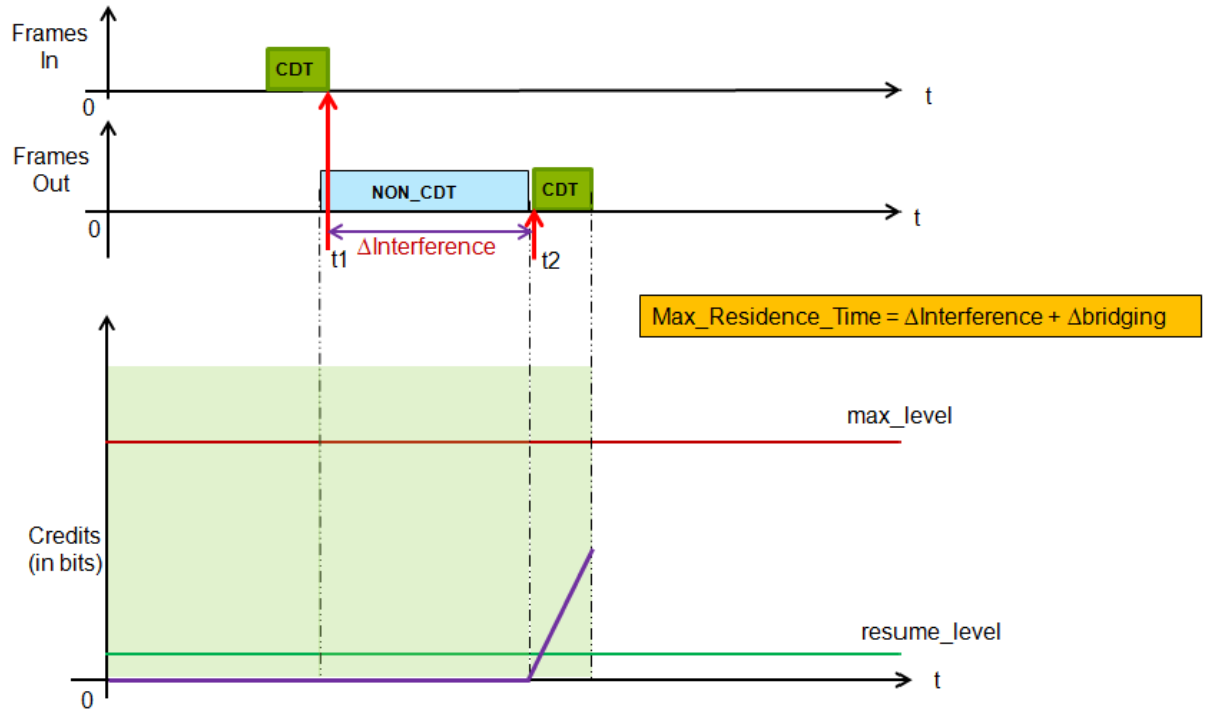


Figure 7.4 Worst case scenario in a Burst Limiting Shaper

It is to be noted that the Δ*interference* is the time to push the non-CDT frame out on the link, i.e, Δ*transmit* time. In addition to this Δ*interference*, there is bridging delay for the CDT frame which is not shown in the figure. So the maximum residence time of the CDT frame inside a switch using a Burst Limiting shaper is given by the following equation

$$\Delta maxSwitching_{BLS} = \Delta bridging + \Delta interference \quad (7.4)$$

And the end-to-end latency is given by

$$\Delta max\_endtoend_{BLS\_nhop} = \Delta talker + n \text{ x } (\Delta transmit + \Delta propagation) + (n-1) \text{ x } (\Delta maxSwitching_{BLS}) + \Delta listener \quad (7.5)$$

7.2.3 Worst case end-to-end latency in a Peristaltic shaper

The Peristaltic shaper described in section 5.3 works on the principle to restrict the residence time of a CDT frame inside a switch to a maximum value. This value depends on the width of the *peristaltic phase*. The width of the peristaltic phase is fixed to 20 µs to achieve an end-to-end latency of 100 µs over five hops. So, the maximum residence time for a CDT frame inside a switch using Peristaltic shaper is 20 µs. This is an ideal situation where there is no interfering traffic at the output port. The worst case scenario for a Peristaltic shaper occurs due to the interference from non-CDT data traffic. Figure 7.5 shows such a worst case scenario at an output port of a switch.

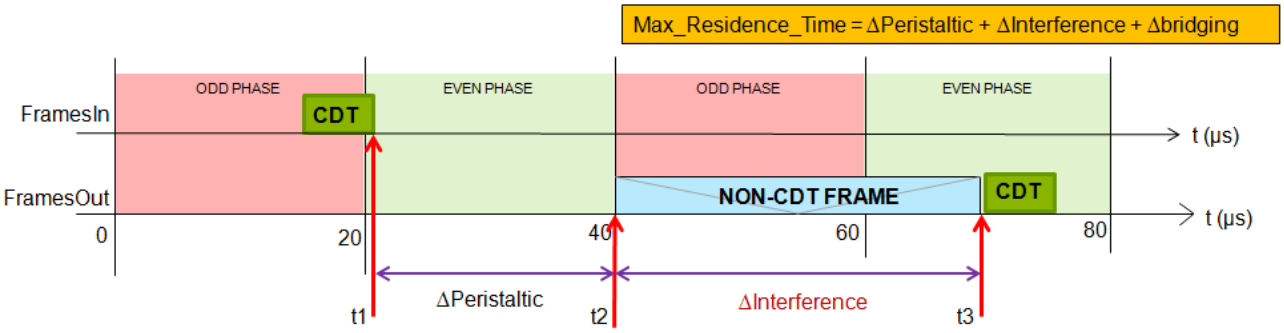


Figure 7.5 Worst case scenario for a Peristaltic shaper

The FramesIn captures the incoming frames and the FramesOut represent the frames being transmitted in the output link. The shaded regions represent odd and even *peristaltic phase*s. The figure shows that a CDT frame is completely received at time t1, which happens to be the beginning of an even *peristaltic phase*. According to Peristaltic shaping algorithm, this frame is selected for transmission only at time t2 which is the beginning of the next *peristaltic phase* (odd phase). This delay, Δ*peristaltic*, is the inherent delay of the Peristaltic shaping algorithm and this can be a maximum of 20 µs. But just before time t2, a non-CDT frame is selected for transmission and it occupies the output link at time t2. Now, the CDT frame is blocked due to the interference from non-CDT frame and can be selected for transmission only at time t3 when the non-CDT frame releases the link. This Δ*interference* due to the non-CDT frame depends on the size of the frame. That is, Δ*interference* is the Δ*transmit* time to push the non-CDT out on the link. Apart from these delays, there is also the bridging delay for the CDT frame which is not shown in the figure. So, the maximum residence time of a CDT frame inside a switch using Peristaltic shaper is given by

$$\Delta maxSwitching_{PS} = \Delta bridging + \Delta peristaltic + \Delta interference \quad (7.6)$$

And the end-to-end latency is given by

$$\Delta max\_endtoend_{PS\_nhop} = \Delta talker + n \text{ x } (\Delta transmit + \Delta propagation) + (n-1) \text{ x } (\Delta maxSwitching_{PS}) + \Delta listener \quad (7.7)$$

7.3 Network scenario for validation

The next step in the validation process is to identify a network scenario that can be used for both theoretical end-to-end latency calculations and simulations. From the literature survey on related works, the scenario used in [10] is chosen. The authors use a similar methodology of comparing the simulation results to the theoretical end-to-end latency calculations. But they analyze only the impact of the Time Aware shaper on AVB Class A data traffic. So the calculations in [10] are for the worst case end-to-end latencies of the AVB Class A traffic rather than for CDT. On the contrary, this thesis work focuses on the impact of non-CDT data traffic (interference) on the end-to-end latency of Class CDT traffic under IEEE802.1TSN frame size restrictions [8]. Since the validation approach used in [10] can be adapted to this thesis work, a similar network scenario is best suited for this validation process. Figure 7.6 shows the network topology of the scenario used for validation

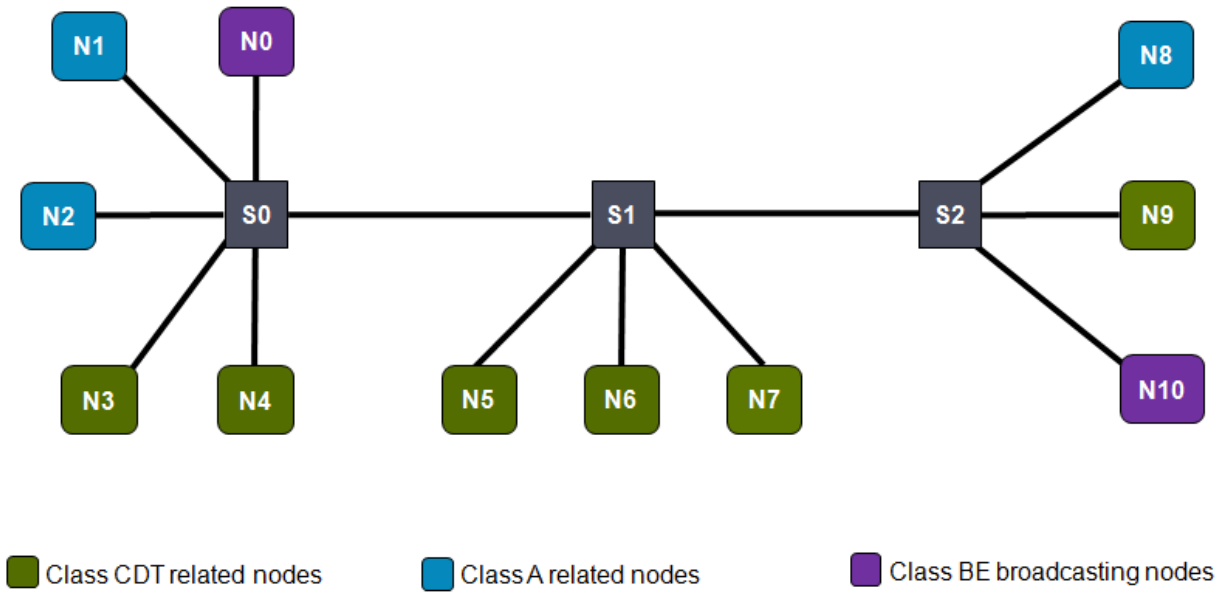


Figure 7.6 Network Topology of the validation scenario

In Figure 7.6, N with a numbered index represents an end node and S with a numbered index represents a switch. The details of all the data traffic in the network including their frame sizes and expected end-to-end latency is listed in Table 7.1.

Source	Destination	Messaging Type	Class	Interval (µs)	Frame Size (Bytes)		Bandwidth with Header (% of total bandwidth)	End-to-End Latency Expectation
					Payload	With Header		
N0	All nodes except N0	Broadcast	BE	Random (poisson)	256	298	#	-
N1	N8	Unicast	AVB_A	125	256	322	20.608	2ms/7 hops
N2	N8	Unicast	AVB_A	125	256	322	20.608	2ms/7 hops
N3	N7	Unicast	CDT	500	128	170	2.72	100 µs/ 5 hops
N4	N7	Unicast	CDT	500	128	170	2.72	100 µs/ 5 hops
N5	N9	Unicast	CDT	500	128	170	2.72	100 µs/ 5 hops
N6	N9	Unicast	CDT	500	128	170	2.72	100 µs/ 5 hops
N10	All nodes except N10	Broadcast	BE	Random (poisson)	256	298	#	-

Table 7.1 Traffic information for Validation Scenario

It is to be noted that the Class BE traffic broadcasted by nodes N0 and N10 is received by all the other nodes in the network. The Class BE traffic is generated at random intervals as explained in section 6.5. AVB Class A traffic is generated every 125 µs interval at both the Class A talker nodes N1 and N2. Class CDT traffic generation and transmission is scheduled among the four CDT talkers (N3, N4, N5 and N6) as shown in the Figure 7.7. The red arrows indicate the time instant at which a particular CDT talker application generates a CDT frame. The four CDT talkers are spaced 100 µs apart. This transmission schedule is common for the simulations with all the three traffic shaping mechanisms.

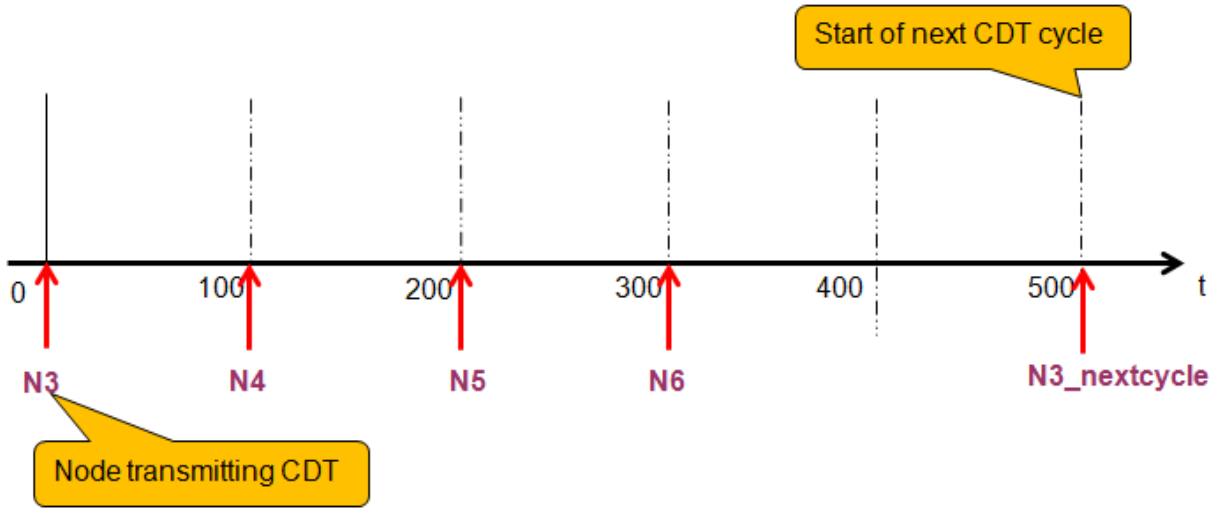


Figure 7.7 Schedule for CDT data traffic transmission

The schedule repeats every 500 µs. The percentage of total bandwidth allocated to Class A and Class CDT traffic is calculated based on the equation 7.5 given below

$$\%\ bandwidth = framesizeInBytes \times \left(\frac{txTime_{byte}}{frameinterval}\right) \times 100 \qquad (7.5)$$

The frame sizes are fixed based on the IEEE802.1TSN standard requirements as explained in section 3.4. The total header size for an AVB frame is 66 bytes (24 bytes of AVTP, 22 bytes of MAC header and 20 bytes of PHY encapsulation). But the Class CDT and Class BE traffic do not include the AVTP header, which gives a total header of 42 bytes for these traffics. The *framesizeInBytes* is inclusive of the header. The $txTime_{byte}$ is the time taken to transmit a byte in the link and this depends on the data rate. Here the data rate is assumed as 100 Mbps for all the links in the network. The total load in the network is 100% with the remaining bandwidth occupied by the Class BE traffic. This is represented as '#' in the Table 7.1. The total bandwidth reserved for each of the traffic class is listed in Table 7.2. AVB Class B traffic is ignored in this configured since the end-to-end latency of the Class CDT traffic affected by the interfering frame size and not the type of the traffic. Since Class B traffic also takes the same size as Class A according to IEEE802.1TSN requirement, only Class A and Class BE traffic is configured for simplicity.

Traffic Class	Reserved Bandwidth (% of Total Bandwidth)
Class CDT	12
Class A	42
Class BE	46

Table 7.2 Bandwidth Reserved for each Traffic Class

The maximum interference for a CDT frame in this configuration is from Class A data traffic with a total frame size of 322 bytes. All the CDT traffic in the network, namely from N3, N4 to N7 and from N5, N6 to N9 takes 3 hops to reach their destination. So, the end-to-end latency requirement for CDT traffic is at most 60 µs according to IEEE802.1TSN requirements (100 µs for 5 hops). The following section describes the computation of theoretical maximum end-to-end latency of a CDT frame for all the three traffic shapers.

7.4 Theoretical maximum for end-to-end latency of CDT traffic and Simulation results

The theoretical maximum value for the end-to-end latency of a CDT frame is computed by considering the worst case scenarios explained in section 7.2.

7.4.1 Time Aware shaper

Time Aware shaping requires scheduled time slots for CDT. Figure 7.8 shows the schedule followed by Time Aware shapers at all switches. The schedule consists of four CDT slots of 15 µs wide which corresponds to the transmission schedule shown in Figure 7.7. The schedule repeats every 500 µs. The width of a CDT slot is fixed based on the bandwidth required for each of the CDT traffics. This is given by Δ*transmit* delay for the CDT frame. The Δ*transmit* delay for a frame of size 170 bytes (inclusive of header) with a 100 Mbps link is 13.6 µs. So a CDT slot size of 15 µs is fixed. This gives a total bandwidth of 12% for all the four CDT traffic. Assuming all the switches are perfectly synchronized, the starts and ends of the slots are also synchronized.

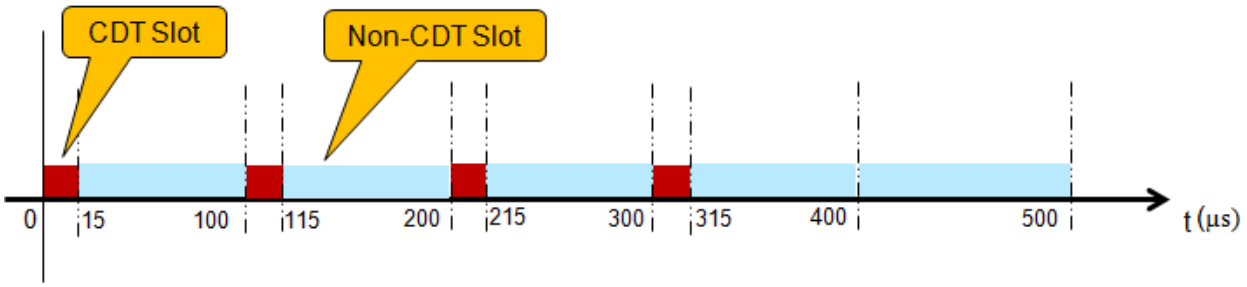


Figure 7.8 Schedule for Time Aware shapers with 4 CDT slots of 15 µs each

In this schedule, a worst case as described in section 7.2.1 occurs at all switches. But the worst case with a maximum Δ*schedule* at a switch occurs for the CDT traffic from node N6 to N9. For example, the CDT frame generated from node N6 at time 300 µs reaches the switch S1 at time 315.178 µs. This is due the sum of the delays Δ*talker*, Δ*transmit* and Δ*propagation* (refer Figure 6.6). So this CDT frame misses the CDT slot by 0.178 µs at switch S1. Now, it has to wait till 500 µs for the start of the next CDT slot. The CDT frame transmitted from switch S1 at 500 µs reaches switch S2 at 515.178 µs. Even here at S2, the CDT frame misses the CDT slot by 0.178 µs. But now, the next CDT slot starts at 600 µs, so the wait time is 100 µs lesser than at the switch S1. So, using the equation 7.3 and the delay values assumed in the simulation model (refer section 6.6), the theoretical maximum end-to-end latency for this scenario is computed as follows

$$\Delta max\_endtoend_{TAS\_3hop} = 1.04 + 3 \times (13.6 + 0.538) + (5 + 185.822) + (5 + 85.822) + 1.02$$

$$\Rightarrow \Delta max\_endtoend_{TAS\_3hop} = 326.118\,\mu s$$

The simulation results of end-to-end latencies of all the four CDT traffic with Time Aware shaper are listed in Table 7.3. The maximum end-to-end latency for the CDT traffic is 316.158 µs, which is less than the theoretical maximum value of 326.118 µs. It is to be noted that the Time Aware shaper gives a constant latency without any jitter since it completely avoids the interference from other data traffic.

CDT Traffic	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
N3 to N7	216.158	216.158	0
N4 to N7	216.158	216.158	0
N5 to N9	316.158	316.158	0
N6 to N9	316.158	316.158	0

Table 7.3 Simulation results for latencies and jitter of CDT traffic with Time Aware Shaper

This worst case end-to-end latency is much higher than the required end-to-end latency of 60 µs. This is due to scheduling and this could be improved by adjusting the schedule at every node in the network. This improvement is explained in Chapter 8.

Table 7.4 lists the simulation results of maximum and minimum end-to-end latency for one of the data streams of all the three different data traffic with the Time Aware shaper. It is to be noted from the table that under worst case scheduling, the maximum end-to-end latency of the CDT traffic can exceed the end-to-end latency of the AVB Class A traffic. Also, the Time Aware shaper has an impact on the end-to-end latencies of Class A data traffic as well. This is discussed in detail in section 9.4 using the simulation results obtained for the Test Case.

Traffic Class	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
Class CDT	316.158	316.158	0.00
Class A	268.318	218.318	50.00
Class BE	7425.160	454.998	6970.16

Table 7.4 Simulation results of end-to-end latency for all traffic with Time Aware shaper

7.4.2 Burst Limiting shaper

The Burst Limiting shaper does not require scheduled CDT slots. It works on the maximum allowed CDT burst. The maximum burst is limited based on the total bandwidth reserved for the CDT traffic as explained in section 6.1. The bandwidth reservations are same as used for the Time Aware shaper (Table 7.2). So using equations 6.1 and 6.2, the *max_level* for credit is derived as 6000 bits.

The worst case scenario for CDT traffic with Burst Limiting shaper occurs due to interference from other traffic. This can occur for all the four CDT traffic in the network. In the network scenario considered, the traffic with maximum interference size is the Class A traffic with a frame size of 322 bytes. This gives rise to a $\Delta interference$ of 25.76 µs. This interference can occur at both switches in the path. So, using equation 7.5 and the delays assumed in the simulation model, the theoretical maximum end-to-end latency for the CDT frame over 3 hops is obtained as follows

$$\Delta max\_endtoend_{BLS\_3hop} = 1.04 + 3 \text{ x } (13.6 + 0.538) + 2 \text{ x } (5 + 25.76) + 1.02$$

$$\Rightarrow \quad \Delta max\_endtoend_{BLS\_3hop} = 105.994\ \mu s$$

The simulation results of end-to-end latencies of all the four CDT traffic with Burst Limiting shaper are listed in the Table 7.5. The maximum end-to-end latency for CDT traffic obtained through simulation is 102.358 µs (for traffic from N6 to N9), which is less than the theoretical maximum value of 105.994 µs. It is to be noted that the minimum end-to-end latency obtained is 51.758 µs, which is less than the theoretical end-to-end latency with zero interference at both the switches (54.474 µs). This is because the observed minimum bridging delay ($\Delta bridging$) at a switch is less than the assumed 5 µs. This difference in end-to-end latency of the CDT gives rise to delivery variations which is expressed as maximum jitter.

CDT Traffic	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
N3 to N7	97.478	51.758	45.72
N4 to N7	98.878	52.158	46.72
N5 to N9	100.958	51.758	49.2
N6 to N9	102.358	53.158	49.2

Table 7.5 Simulation results for latencies and jitter of CDT traffic with Burst Limiting Shaper

This worst case end-to-end latency is much higher than the required end-to-end latency of 60 µs. This is due to interference and this could be improved by pre-empting the interfering frame to transmit the CDT frame. Frame pre-emption is not implemented. However, the effect of frame pre-emption on the end-to-end latency is explained theoretically in Chapter 10.

Table 7.6 lists the simulation results of maximum and minimum end-to-end latency for one of the data streams of all the three different data traffic with the Burst Limiting shaper. The impact of Burst Limiting shaper on Class A data traffic is discussed in detail in section 9.4 using the simulation results obtained for the Test Case.

Traffic Class	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
Class CDT	102.358	53.158	49.20
Class A	245.678	171.918	73.76
Class BE	5472.320	290.398	5181.92

Table 7.6 Simulation results of end-to-end latency for all traffic with Burst Limiting shaper

7.4.3 Peristaltic shaper

Peristaltic shaper requires only a single parameter for configuration, i.e. the width of the *peristaltic phase*. This width depends on the maximum end-to-end latency required for the CDT traffic. It is fixed as 20 µs based on the end-to-end latency requirement from IEEE802.1TSN [8]. The transmission schedule and the bandwidth reservation for the CDT traffic is same as used for the other two shapers.

The worst case scenario for CDT traffic with Peristaltic shaper occurs due to both the peristaltic delay and the interference from other traffic. This can occur for all the four CDT traffics in the network. In the network scenario considered, the peristaltic delay ($\Delta peristaltic$) at a switch is 20 µs. The traffic with maximum interference size is the Class A traffic with a frame size of 322 bytes. This gives rise to a $\Delta interference$ of 25.76 µs. The worst case can occur at both the switches in the path. So, using equation 7.7 and the delays assumed in the simulation model, the theoretical maximum end-to-end latency for the CDT frame over 3 hops is obtained as follows

$$\Delta max\_endtoend_{PS\_3hop} = 1.04 + 3 \times (13.6 + 0.538) + 2 \times (5 + 20 + 25.76) + 1.02$$

$$\Rightarrow \quad \Delta max\_endtoend_{PS\_3hop} = 145.994\ \mu s$$

The simulation results of end-to-end latencies of all the four CDT traffic with Peristaltic shaper are listed in Table 7.7. The maximum end-to-end latency for CDT traffic obtained through simulation is 120.798 µs (for traffic from N5 to N9), which is less than the theoretical maximum value of 145.994 µs. It is to be noted that the minimum end-to-end latency obtained is 56.558 µs, which is close to the end-to-end latency with zero interference and zero peristaltic delay at both the switches (54.474 µs). The case of zero interference and zero peristaltic delay at both the switches is not achieved in the simulation.

CDT Traffic	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
N3 to N7	102.038	56.318	45.72
N4 to N7	102.438	56.198	46.24
N5 to N9	120.798	56.158	64.64
N6 to N9	119.718	56.558	63.16

Table 7.7 Simulation results for latencies and jitter of CDT traffic with Peristaltic Shaper

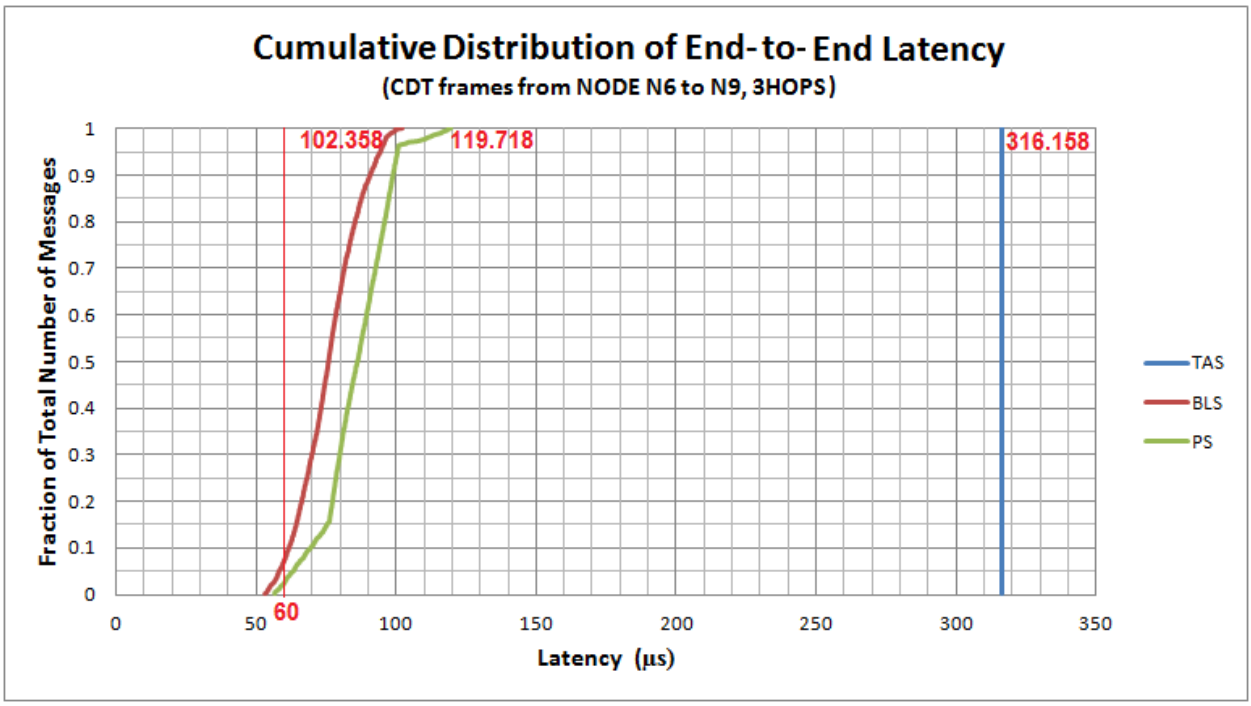
This worst case end-to-end latency is much higher than the required end-to-end latency of 60 µs. This is due to both peristaltic delay and interference. The worst case could be improved by using two techniques, one using a guard band during the peristaltic wait period and other using frame pre-emption. The guard band technique is implemented and the improvement in end-to-end latency is shown with simulation results in Chapter 8. Frame pre-emption is not implemented to improve the performance. However, the effect of frame pre-emption on the end-to-end latency is explained theoretically in Chapter 10.

Table 7.8 lists the simulation results of maximum and minimum end-to-end latency for one of the data streams of all the three different data traffic with the Peristaltic shaper. The impact of Peristaltic shaper on Class A data traffic is is discussed in detail in section 9.4 using the simulation results obtained for the Test Case.

Traffic Class	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
Class CDT	120.798	56.158	64.64
Class A	253.878	184.878	69.00
Class BE	5489.12	341.998	5147.12

Table 7.8 Simulation results of end-to-end latency of all traffic with Peristaltic shaper

For easier comparison of the performance of all the three shapers in terms of worst case end-to-end latency and jitter, the simulation results for the end-to-end latency of all the CDT frames from node N6 to N9, with all the three traffic shapers is expressed in a cumulative distribution (Graph 7.1). The required maximum end-to-end latency of 60 µs is shown in a thin red line and the maximum end-to-end latency from each of the shapers is highlighted.



Graph 7.1 Cumulative distribution of end-to-end latency for CDT traffic from N6 to N9 with all the three traffic shapers

8. Improvements for End-to-End Latency

The validation experiment explained in the previous chapter shows that none of the traffic shaper is able to achieve the end-to-end latency requirement of 60 µs over 3 hops for the CDT traffic in worst case scenarios. But these worst case scenarios can be improved to meet the end-to-end latency requirements. This section describes the mechanisms implemented in Time Aware shaper and the Peristaltic shaper to improve end-to-end latency requirements. Time Aware Shaper is improved by making adjustments in the scheduling technique and sacrificing on overall bandwidth utilization. On the other hand, the Peristaltic shaper is improved by proposing a guard band mechanism to minimize the interference from other data traffic. The end-to-end latency improvements achieved through these mechanisms is also shown in this section.

8.1 Adjustments in scheduling for Time Aware Shaper

The worst case in Time Aware shaper occurs because the scheduling does not take the frame traversal delay, Δ*traversal* into account. Δ*traversal* is the sum of Δ*transmit* and Δ*propagation*

$$\Delta traversal = \Delta transmit + \Delta propagation \tag{8.1}$$

To improve the worst case explained in section 7.2.1, the schedule at each node needs to be adjusted for the frame traversal delay. This requires the schedule to be staggered at each of the nodes.

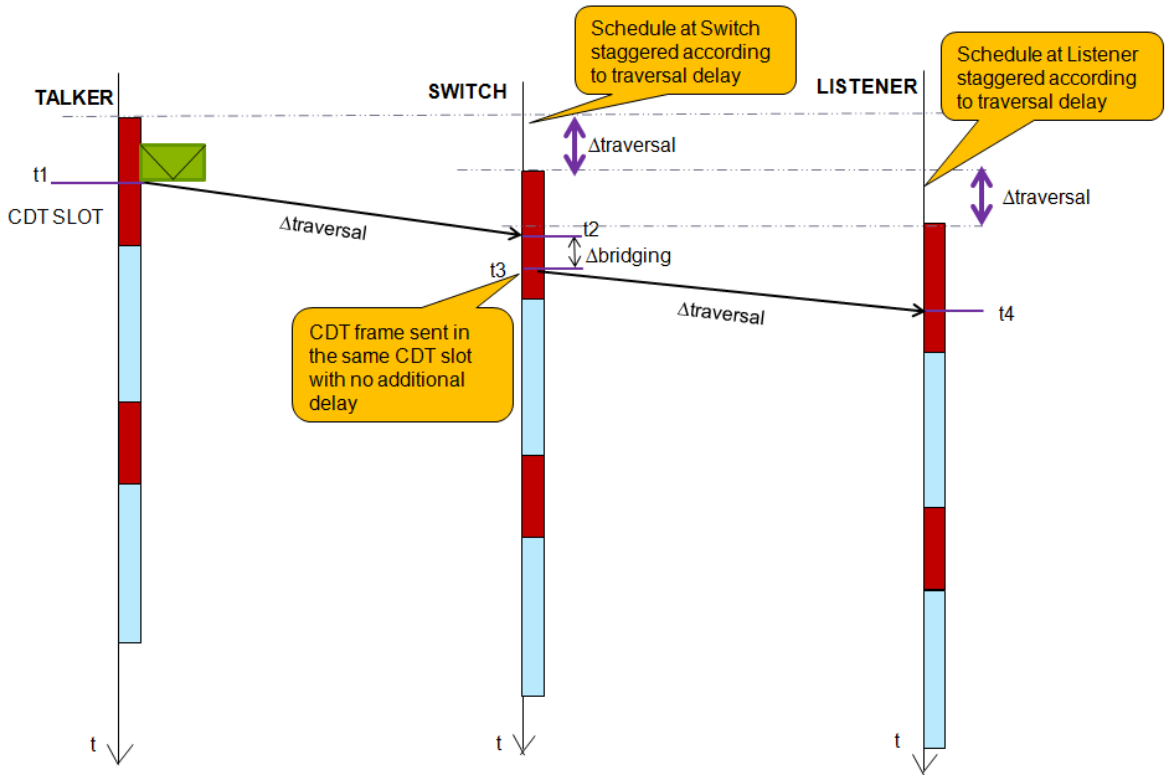


Figure 8.1 Time Aware shaper with staggered schedule to avoid additional delay

Figure 8.1 shows such a staggered schedule for the two hop network considered for explanation in section 7.2.1. The figure shows that the schedule at the switch is offset by the traversal delay and the schedule at the Listener is offset by two times the traversal delay. With this new schedule, the CDT frame transmitted by the Talker at time t1 arrives at the switch at time t2. Because of the staggering, t2 falls in the CDT slot and the CDT frame is transmitted at time t3 after the bridging delay. By this the additional delay (Δ*schedule*) shown in Figure 7.1 is avoided.

The staggered schedule appears to solve the problem, but it is very complex to determine the time to offset the schedule at each node. The traversal delay in equation 8.1 is variable and depends on the size of the CDT frame. IEEE802.1TSN restricts the maximum payload size of the CDT frame to 128 bytes. So applying this restriction, the maximum traversal delay of the CDT frame at each link for a fixed link speed can be identified and the schedules can be staggered using this maximum traversal delay.

The scenario in Figure 8.1 explains the traffic flowing in only one direction. In this case, the staggering could solve the worst case problem. But if the traffic is flowing in both directions, the staggering should account for the traversal delays from both the directions. This gives rise to two different schedules, one staggered for the traffic in forward direction and one for the traffic in reverse direction. Moreover, the staggering offset at a node also depends on its relative position from the Talker. That is, the number of hops from the Talker. This adds to the complexity of configuring the schedule offline. In [44] the authors analyze the complexity of static scheduling and synthesize an optimal schedule for control data traffic taking each branch in the network topology into account. But that results in a table driven scheduling for each of the links.

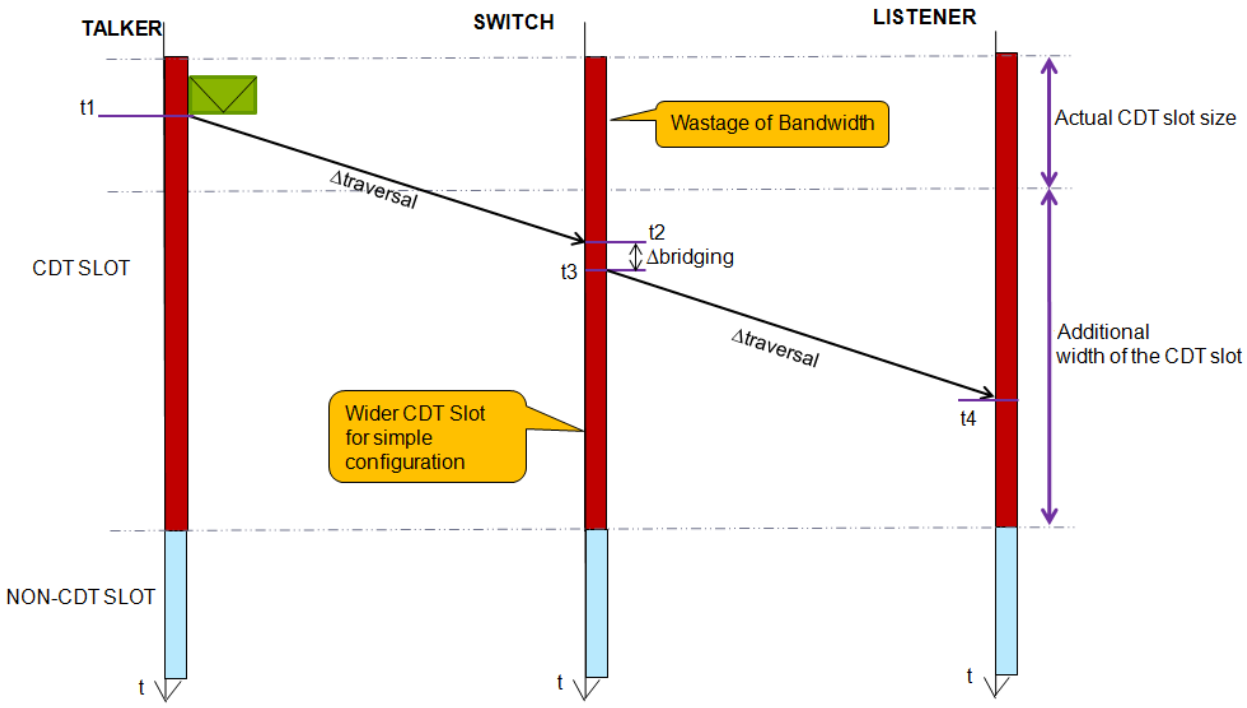


Figure 8.2 Time Aware shaper with a wider CDT slot

One of the simpler solutions is to have a CDT slot wide enough to accommodate the end-to-end traversal delay of the CDT frame. That is, the original CDT slot size is widened to compensate for the end-to-end frame traversal. This is definitely wastage of bandwidth at each node as the rest of the CDT slot remains unused. So, the available bandwidth for the lower priority traffic is sacrificed using this technique. Figure 8.2 shows such a schedule in a two hop network. Here the CDT slot at the switch is wide enough such that the CDT frame transmitted at time t1 from the talker reaches at the switch at time t2 which falls in the CDT slot. So it is transmitted just after the bridging delay at time t3 with no additional wait. The configuration of this type of schedule is as simple as the first schedule shown in Figure 7.1. The schedule at all the nodes is identical with the starts and ends of the CDT slots perfectly synchronized (assuming no synchronization error). The only requirement is to calculate the end-to-end traversal delay of the CDT frame for the maximum number of hops in the network and add this to every CDT slot in the schedule.

Using this improvement, the equation 7.2 changes to

$$\Delta maxSwitching_{TAS} = \Delta bridging \tag{8.2}$$

And using the above equation in equation 7.3, the worst case end-to-end latency for a CDT frame over *n* hops in a network using Time Aware shaper can be redefined as follows

$$\Delta max\_endtoend_{TAS\_nhop} = \Delta talker + n \text{ x } (\Delta transmit + \Delta propagation) + \sum_{i=1}^{n-1} (\Delta bridging)_i + \Delta listener \tag{8.3}$$

8.1.1 Improvement results for Time Aware shaper

The schedule for Time Aware shaper used in the validation experiment consisted of four CDT slots each of width 15 µs (Figure 7.8). This schedule needs to be reconfigured for the improvement. Considering the network scenario for validation, the end-to-end traversal time of a CDT frame of size 170 bytes (inclusive of header) is computed using equation 8.1 as 14.138 µs. This is approximated to 15 µs. This is the traversal delay of a CDT frame for one hop. The worst case number of hops in the validation scenario is 3. So the end-to-end traversal delay for a CDT frame is approximated to 45 µs. This is fixed as the width of each CDT slot. The new schedule with wider CDT slots is shown in Figure 8.3.

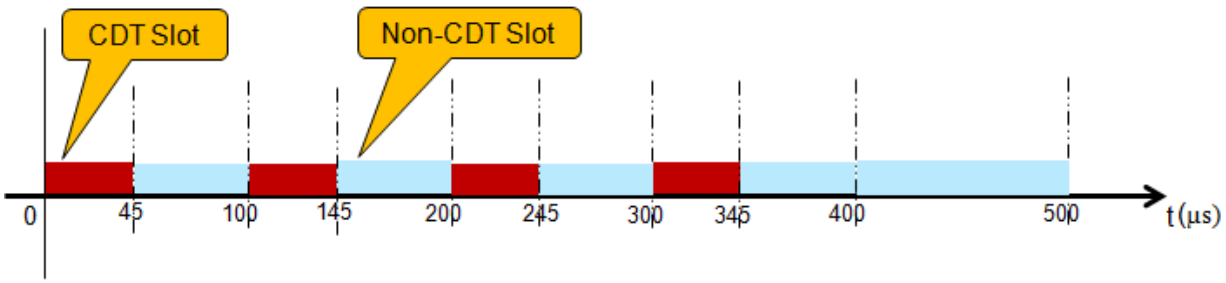


Figure 8.3 Schedule for Time Aware shapers with 4 CDT slots of 45 µs each

The size of the new CDT slot is three times the actual CDT slot size and this is wastage of total bandwidth. The bandwidth reservation for each traffic class changes from the actual reservation used in

the validation experiment (refer Table 7.2). Table 8.1 shows the modified bandwidth reservation for the improved schedule with a wider CDT slot.

Traffic Class	Reserved Bandwidth (% of Total Bandwidth)
Class CDT	36
Class A	42
Class BE	22

Table 8.1 Bandwidth Reserved for each Traffic Class for a schedule with wider CDT slot

It can be seen that the percentage of the total bandwidth reserved for Class CDT increased from 12% to 36%. This affects the bandwidth available for the Class BE traffic, which is reduced to 22% from the actual 46%. In a network with more Class CDT streams, the wider CDT slot mechanism is ineffective as it utilizes the bandwidth available for other lower priority traffic. In that case, the staggering schedule is more effective as it uses the same bandwidth as the actual reservation. In this thesis work, the wider slot mechanism is adapted just to show that it is possible to achieve lower end-to-end latency for Time Aware shaper with adjustments in scheduling.

The theoretical maximum value for end-to-end latency of a CDT frame over 3 hops is computed using the equation 8.3 and the assumed values for the delays in the simulation model as follows

$$\Delta max\_endtoend_{TAS\_3hop} = 1.04 + 3 \times (13.6 + 0.538) + 5 + 5 + 1.02$$

$$\Rightarrow \quad \Delta max\_endtoend_{TAS\_3hop} = 54.474\,\mu s$$

The simulation results of end-to-end latencies for all the four CDT traffic in the validation network scenario using the Time Aware shaper with a wider CDT slot is listed in the Table 8.2. The maximum end-to-end latency for the CDT traffic is 50.598 µs, which is less than the theoretical maximum value of 54.474 µs. This is due to the variation in the actual bridging delay at the switches. It is to be noted that the worst case end-to-end latency is drastically improved from 316.158 µs which was obtained during the validation experiment.

CDT Traffic	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
N3 to N7	50.598	50.598	0
N4 to N7	50.598	50.598	0
N5 to N9	50.598	50.598	0
N6 to N9	50.598	50.598	0

Table 8.2 Simulation results for latencies and jitter of CDT traffic with Time Aware Shaper using a wider CDT slot

This improvement proves that, with proper scheduling it is possible to achieve the IEEE802.1TSN requirement for end-to-end latency (in this case 60 µs over 3 hops) in a network using Time Aware

shaper. Also, the Time Aware shaper gives a constant latency without any jitter since it completely avoids the interference from other data traffic.

The use of the wider schedule has adverse affects on the end-to-end latency of other data traffic. Table 8.3 lists the simulation results of maximum and minimum end-to-end latency for one of the data streams of all the three different data traffic in the validation scenario using Time Aware shaper with a wider CDT slot. It can be noted that the maximum end-to-end latencies of both Class A traffic and Class BE traffic are increased as compared to that of the validation experiment.

Traffic Class	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
Class CDT	50.598	50.598	0.00
Class A	448.318	398.318	50.00
Class BE	14807.9	393.358	14414.6

Table 8.3 Simulation results of end-to-end latency for all traffic with Time Aware shaper using a wider CDT slot

8.2 Guard band for Peristaltic Shaper

The worst case performance of the Peristaltic shaper happens due the interference from non-CDT traffic occurring at the end of the *peristaltic delay* (refer section 7.2.3). The *peristaltic delay* is determined by computing the *peristaltic outphase* (*pTphase*) after a CDT frame is completely received and is available in the output queue. Since the switch is aware that a CDT frame is already available in the CDT queue for transmission, it is possible to avoid any interference from non-CDT traffic during this *peristaltic delay* period. A guard band is enabled such that no other traffic can be selected for transmission until this CDT frame is transmitted. This enforces the blocking of the transmission selection mechanism for the entire *peristaltic delay* period. This could be a maximum of 20 µs, since the worst case *peristaltic delay* is 20 µs. But the guard band ensures that no other traffic interferes the transmission of the CDT frame at the end of *peristaltic delay*. This improvement using the guard band is shown in Figure 8.4. The figure shows the even phase, in which the CDT frame is received, is enabled with a guard band (EVEN_PHASE_GUARD_BAND). The non-CDT data traffic is blocked during this duration but any other CDT data traffic is allowed to be selected for transmission.

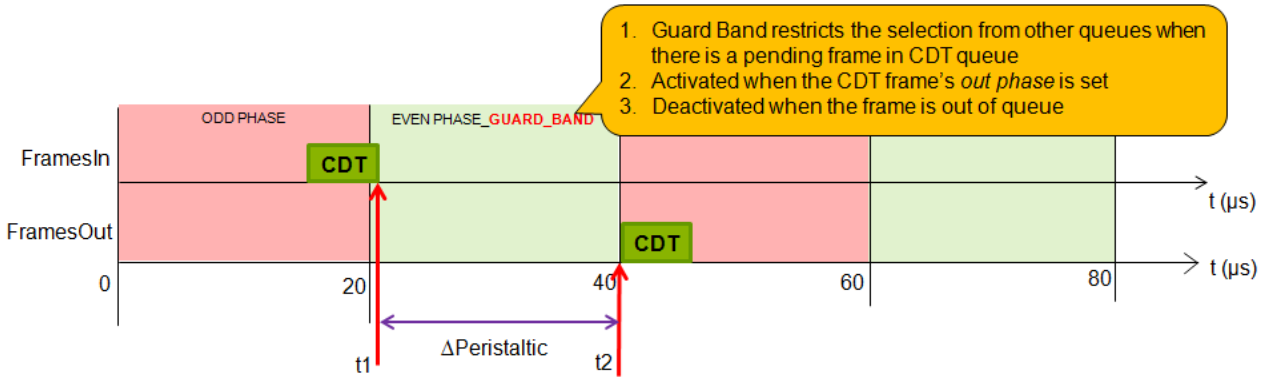


Figure 8.4 Peristaltic shaper with guard band to avoid non-CDT interference

So the guard band avoids interference from the non-CDT traffic but not from the other CDT traffic. Since it is assumed that the CDT traffic is scheduled and cannot interfere with each other, the guard band does not have an impact on other CDT data traffic. Even with the guard band, the interference cannot be completely avoided. This scenario is shown in Figure 8.5. The figure shows that a CDT frame is completely received and available for transmission at time t1. The guard band is enabled at time t1. But at time t1, a non-CDT frame is already being transmitted at the output link and the link is occupied by it. It happens that the non-CDT frame is selected for transmission just before time t1. Now, even with the guard band enabled, the CDT frame has to wait till the complete transmission of the non-CDT frame. The Δ*interference* due to the non-CDT frame could be larger than the *peristaltic delay* of 20 µs. In this case, the maximum residence time of the CDT frame is either the *peristaltic delay* or the delay due to the interference from non-CDT frame whichever is greater. Apart from these delays, there is also the bridging delay for the CDT frame which is not shown in the figure.

Using the guard band improvement, the equation 7.6 for the maximum residence time for a CDT frame in a switch using Peristaltic shaper with guard band can be redefined as follows

$$\Delta maxSwitching_{PS\_GB} = \Delta bridging + max\,(\Delta peristaltic, \Delta interference) \quad (8.4)$$

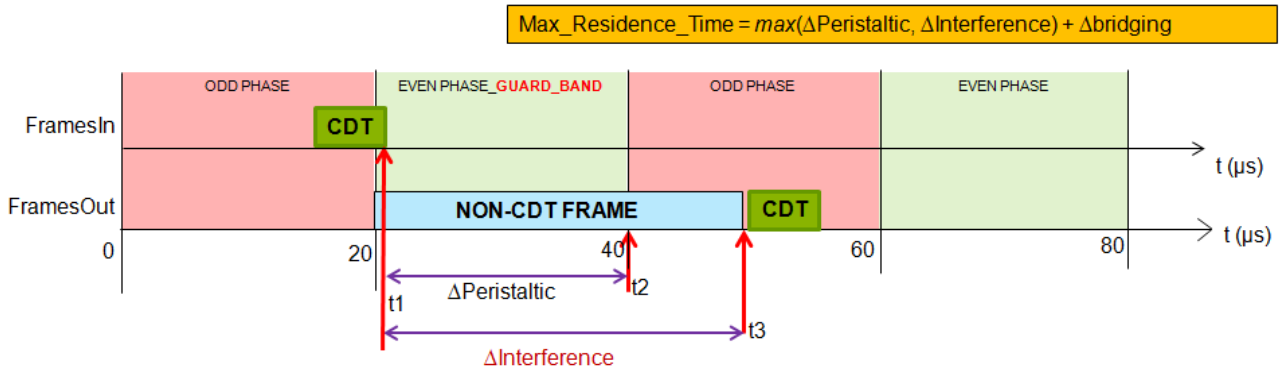


Figure 8.5 Worst case in a Peristaltic shaper with guard band

8.2.1 Results for Peristaltic shaper with guard band

The Peristaltic shaper improved with the guard band mechanism does not require any additional change in the configuration. The only configuration parameter, the width of the *peristaltic phase* remains the same 20 µs as the previous configuration. Unlike the improvement for Time Aware shaper, there is no change in the bandwidth reservation as well. The same bandwidth reservation used for the validation experiment (Table 7.2) is used.

The traffic with maximum interference size is the Class A traffic with a frame size of 322 bytes. This gives rise to a Δ*interference* of 25.76 µs. Since this value is greater than the peristaltic delay, Δ*peristaltic*, the maximum residence time of a CDT frame in a switch depends only on the Δ*interference.* The theoretical maximum value for end-to-end latency of a CDT frame over 3 hops is computed using the equation 8.4 in equation 7.7, including the assumed values for the delays in the simulation model as follows

$$\Delta max\_endtoend_{PS\_GB\_3hop} = 1.04 + 3 \text{ x } (13.6 + 0.538) + 2 \text{ x } (5 + 25.76) + 1.02$$

⇨ $\Delta max\_endtoend_{PS\_GB\_3hop} = 105.994\ \mu s$

It can be noted that this is the same value obtained for the Burst Limiting shaper (refer section 7.4.2). So the Peristaltic shaper with the guard band delivers the maximum end-to-end latency same as that of Burst Limiting shaper.

The simulation results of end-to-end latencies for all the four CDT traffic in the validation network scenario using the Peristaltic shaper with guard band mechanism is listed in the Table 8.4. The maximum end-to-end latency for CDT traffic obtained through simulation is 91.278 µs (for traffic from N4 to N7 and for traffic from N6 to N9), which is less than the theoretical maximum value of 105.994 µs. It is to be noted that the worst case end-to-end latency is considerably improved from 120.798 µs which was obtained during the validation experiment without the use of guard band. Also, the minimum end-to-end latency obtained (60.438 µs) is higher than the end-to-end latency with zero interference and zero peristaltic delay at both the switches (54.474 µs). The case of zero interference and zero peristaltic delay at both the switches is not achieved in the simulation.

CDT Traffic	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
N3 to N7	78.478	72.478	6.00
N4 to N7	91.278	60.438	30.84
N5 to N9	78.478	76.158	2.32
N6 to N9	91.278	65.278	26.00

Table 8.4 Simulation results for latencies and jitter of CDT traffic for Peristaltic Shaper with guard band

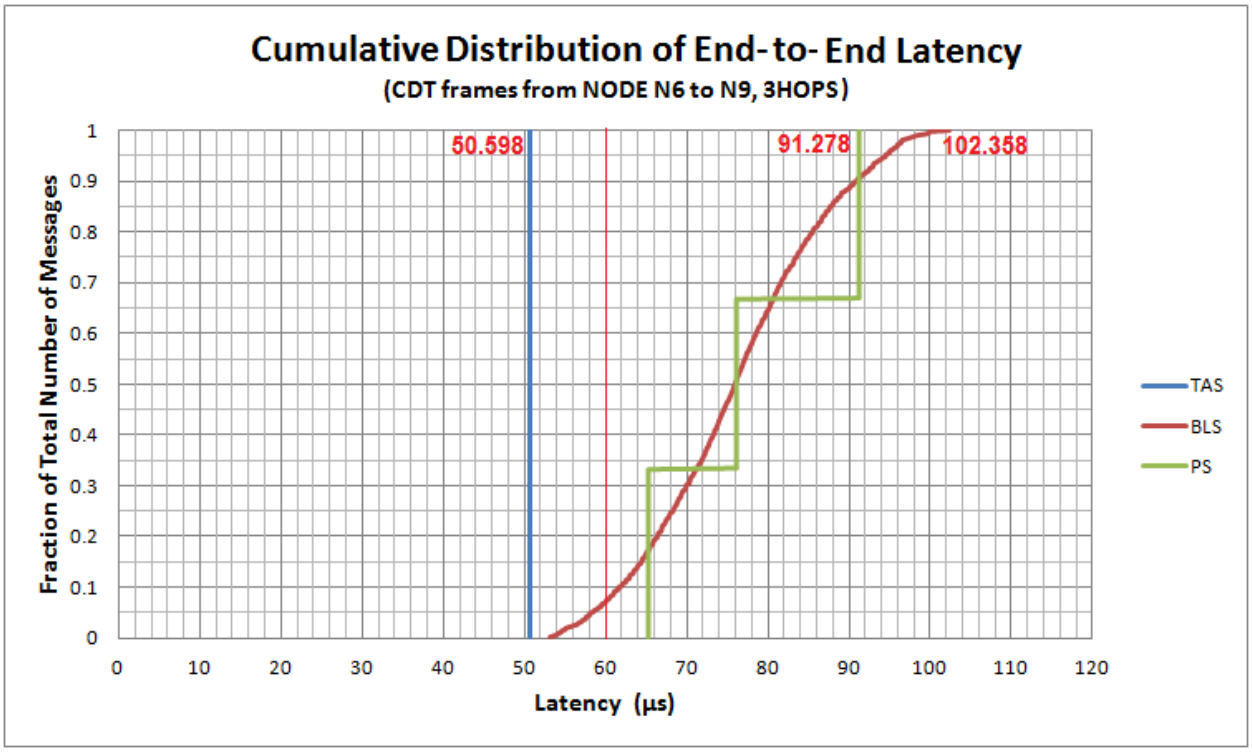
The IEEE802.1TSN requirement for end-to-end latency (60 µs over 3 hops) is not obtained in the Peristaltic shaper even with this improvement. But as the worst case end-to-end latency for the CDT frame depends only on the interference, frame pre-emption mechanism could further improve the latency performance of the Peristaltic shaper. A theoretical analysis of the impact of frame pre-emption mechanism on the improved Peristaltic shaper (with guard band) is given in section 10.1.3.

Traffic Class	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
Class CDT	91.278	60.438	30.84
Class A	257.918	181.558	76.36
Class BE	5511.04	341.998	5169.04

Table 8.5 Simulation results of end-to-end latency of all traffic for Peristaltic shaper with guard band

Table 8.5 lists the simulation results of maximum and minimum end-to-end latency for one of the data streams of all the three different data traffic in the validation scenario using Peristaltic shaper with guard

band. The impact of Peristaltic shaper with guard band on Class A data traffic is discussed in detail in section 9.4 using the simulation results obtained for the Test Case.



Graph 8.1 Cumulative distribution of end-to-end latency for CDT traffic from N6 to N9 with improved Time Aware shaper and Peristaltic shaper

For easier comparison of the improved performance of all the three traffic shapers in terms of worst case end-to-end latency and jitter, the simulation results for the end-to-end latency of all the CDT frames from node N6 to N9, with all the three traffic shapers is expressed in a cumulative distribution (Graph 8.1). Though the Burst Limiting shaper is not improved, the graph includes the results of the Burst Limiting shaper from the validation experiment just to serve as a reference. The required maximum end-to-end latency of 60 µs is shown in a thin red line and the maximum end-to-end latency from each of the shapers is highlighted. The plateaus in the cumulative distribution for Peristaltic shaper are due to the jump in the end-to-end latency. For instance, the plateau at 30% of the messages represent that there is a jump in end-to-end latency from 66 µs (approx) to 76 µs (approx). None of the CDT frames transmitted take an end-to-end latency value between 66 µs and 76 µs.

9. Test Case

This chapter describes simulation results obtained for a test scenario different from that of the validation scenario. The simulation is performed with Time Aware shaper by including scheduling adjustments, with Peristaltic shaper improved using guard band and with normal Burst Limiting shaper. The network scenario used is inspired from a realistic use-case considered at NXP. The sub-sections are organized to explain the network scenario, different data traffic in the network, configuration of the shapers and the simulation results of end-to-end latency for CDT traffic.

9.1 Network Scenario

The network scenario used for this test case represents a close to real in-vehicle Ethernet network. Figure 9.1 shows the topology of the network scenario used.

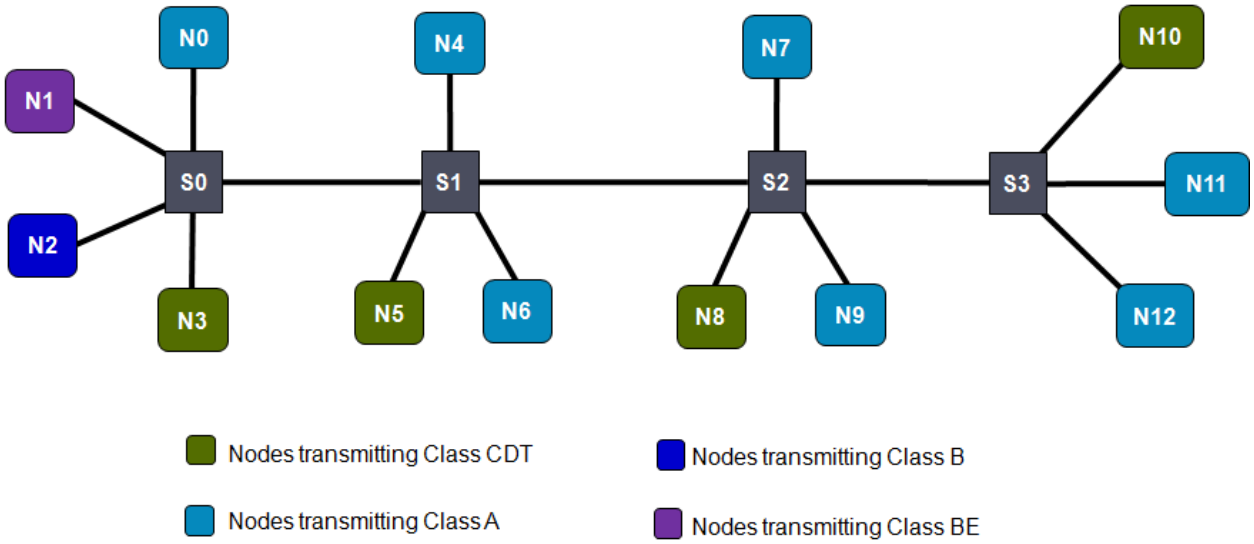


Figure 9.1 Network Topology of the Test case scenario

The topology consists of four switches and thirteen end-nodes connected using 100 Mbps links. In the topology, box N with a numbered index represents an end node and box S with a numbered index represents a switch. The nodes with a green shade are the nodes capable of transmitting Class CDT traffic. And the nodes with a blue shade are the nodes capable of transmitting AVB Class A traffic. Similarly the purple (N1) and the deep-blue shaded nodes (N2) represent the nodes transmitting Class BE and AVB Class B data traffic respectively. Each of the nodes can transmit only one type of data traffic, but they can receive multiple types of data traffic. Table 9.1 shows the details of all the data traffic in the network including their frame sizes, bandwidth requirements and expected end-to-end latency.

It is to be noted that except the CDT traffic from nodes N8 and N10 to node N5, all the other data traffic is multicast. Nodes N6, N9, N11, and N12 listen from most of the other nodes. On the other hand the nodes N0, N1, N2 and N3 are just transmitting nodes and receive no traffic from other nodes.

Source	Destination	Messaging Type	Class	Interval (µs)	Frame Size (Bytes)	Bandwidth with Header (% of total bandwidth)	Latency Expectation
N0	N11, N12	Multicast	AVB_A	125	194	12.5	2ms/7 hops
N1	N11, N12	Multicast	BE	Random (Poisson)	554	#	-
N2	N11, N12	Multicast	AVB_B	250	512	16.4	50ms/7 hops
N3	N4, N5, N7	Multicast	CDT	500	74	1.2	100 µs/5 hops
N4	N6, N9	Multicast	AVB_A	125	194	12.5	2ms/7 hops
N5	N6, N12	Multicast	CDT	500	52	0.8	100 µs/5 hops
N6	N4, N11, N12	Multicast	AVB_A	125	130	8.3	2ms/7 hops
N7	N4, N6	Multicast	AVB_A	125	98	6.3	2ms/7 hops
N8	N5	Unicast	CDT	500	58	0.9	100 µs/5 hops
N9	N4, N11, N12	Multicast	AVB_A	125	130	8.3	2ms/7 hops
N10	N5	Unicast	CDT	500	52	0.8	100 µs/5 hops
N11	N4, N6	Multicast	AVB_A	125	130	8.3	2ms/7 hops
N12	N4, N6	Multicast	AVB_A	125	130	8.3	2ms/7 hops

Table 9.1 Traffic information for Test Scenario

The frame size varies for each of the CDT traffic, but it is within the IEEE recommended payload size of 128 bytes. For instance, the CDT traffic from node N5 and N10 has a payload of only 8 bytes. Since the minimum Ethernet payload size is fixed as 10 bytes in the simulation, this traffic has a total frame size of 52 bytes with the header (10 bytes payload and 42 bytes header). The AVB frames (both Class A and Class B) have a total header size of 66 bytes with an additional 24 bytes of AVTP header.

The frame size for AVB Class A traffic also varies but within the limits of the recommended payload of 256 bytes. But the frame size of the AVB Class B traffic is fixed as 512 bytes inclusive of the header. Also the frame size of Class BE traffic is fixed as 554 bytes inclusive of the header. This definitely violates the IEEE802.1TSN restrictions, but it is considered to verify whether the end-to-end latency of CDT frames of smaller sizes could be affected. Moreover, the Class B which represents the video traffic and Class BE which represents the internet traffic has a payload of 1500 bytes in the IEEE802.1AVB protocol for practical reasons. So an intermediate value of 512 bytes is considered for these traffic.

9.2 Configurations

The Class BE traffic is generated at node N1 at random intervals as explained in section 6.5. AVB Class A traffic is generated every 125 µs interval at all of the Class A talker nodes. AVB Class B traffic is generated every 250 µs interval at node N2. Class CDT traffic generation and transmitted from nodes N3, N5, N8 and N10 as per the schedule shown in the Figure 9.2. The red arrows indicate the time instant at which a particular CDT talker application generates a CDT frame. The four CDT talkers are allowed to

transmit at the same time instant. This transmission schedule is common for the simulations with all the three traffic shaping mechanisms.

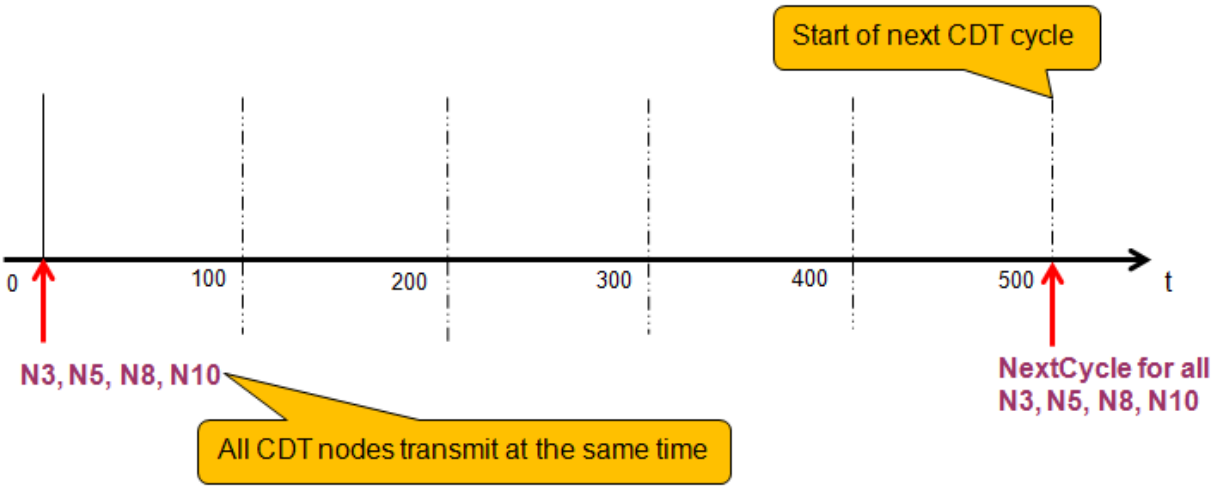


Figure 9.2 Schedule for CDT data traffic transmission in Test Scenario

9.2.1 Time Aware Shaper

The Time Aware shaper requires scheduling to be adjusted with a wider CDT slot for an improved performance in terms of end-to-end latency. The schedule is fixed based on the bandwidth required for the CDT traffic and also based on the end-to-end traversal delay of a CDT frame. The total bandwidth required for the CDT traffic in the network is 3.7 Mbps (from the Table 9.1). This can be approximated to 4 Mbps. The schedule used in the Time Aware shapers of all the switches shown in Figure 9.3 uses only a single CDT slot of 30 µs wide for all the four CDT traffic.

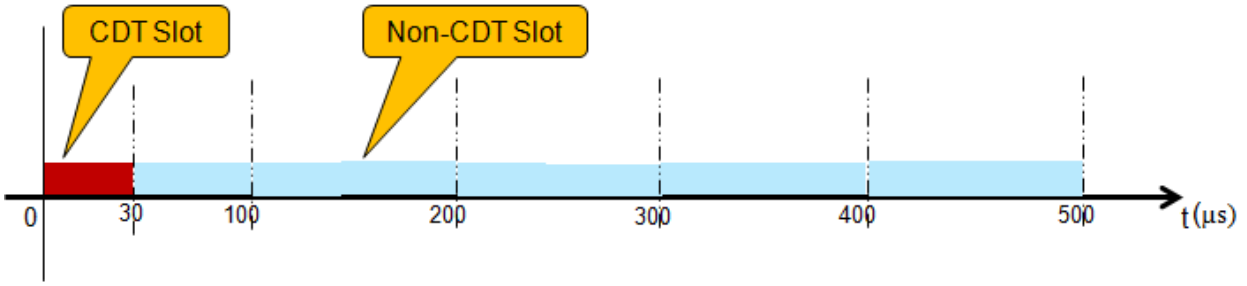


Figure 9.3 Schedule for Time Aware shapers in Test Scenario

This single CDT slot size is sufficient because the CDT traffic from different nodes does not interfere with each other in the path towards their destination. For instance, the CDT frames from node N3 and node N8 both have node N5 as their destination and both the traffics require 3 hops in this case. But they do not collide with each other because of their different frame sizes. The CDT frame from node N8 has a smaller frame size and reaches the switch S1 (where N5 is connected) much earlier than the frame from N3. This can be observed for all other combinations of CDT traffic in the network. So there is no chance of interference among the CDT data traffic. This allows to schedule all the Talkers to transmit at the same time and thereby to have a single CDT slot at all switches in the network. Even though the assumption of

transmitting CDT traffic separately (one at a time) is violated, the collision is avoided since each of the CDT traffics take a separate path towards its destination.

But the entire CDT slot has to be wide enough to accommodate the end-to-end traversal of a single CDT frame. In this scenario, the maximum size of the CDT frame is 74 bytes and the maximum number of hops is 4. This results in an end-to-end traversal delay of 25.832 µs according to equation 8.1. This occupies about 5% of bandwidth in a CDT cycle period. Considering that the bandwidth for a single CDT stream is about 1% (from Table 9.1), the CDT slot width is fixed as 6% of the CDT cycle period, i.e. 30 µs.

9.2.2 Burst Limiting shaper

The configuration required for Burst Limiting shaper is the maximum burst level of the CDT, *max_level*. The *max_level* is depends on the total bandwidth required for the CDT traffic. From Table 9.1, it can be seen the total bandwidth for CDT traffic is around 4%. This results in a *max_level* of 2000 bits using the equations 5.1 and 5.2.

9.2.3 Peristaltic shaper

The configuration required for the peristaltic shaper is the width of the *peristaltic phase*. This remains the same 20 µs because of the IEEE802.1TSN requirements for end-to-end latency of a CDT frame. The Peristaltic shaper improved with the guard band mechanism is used to reduce the maximum residence time for a CDT frame in a switch.

9.3 Worst case for CDT and Theoretical end-to-end latency

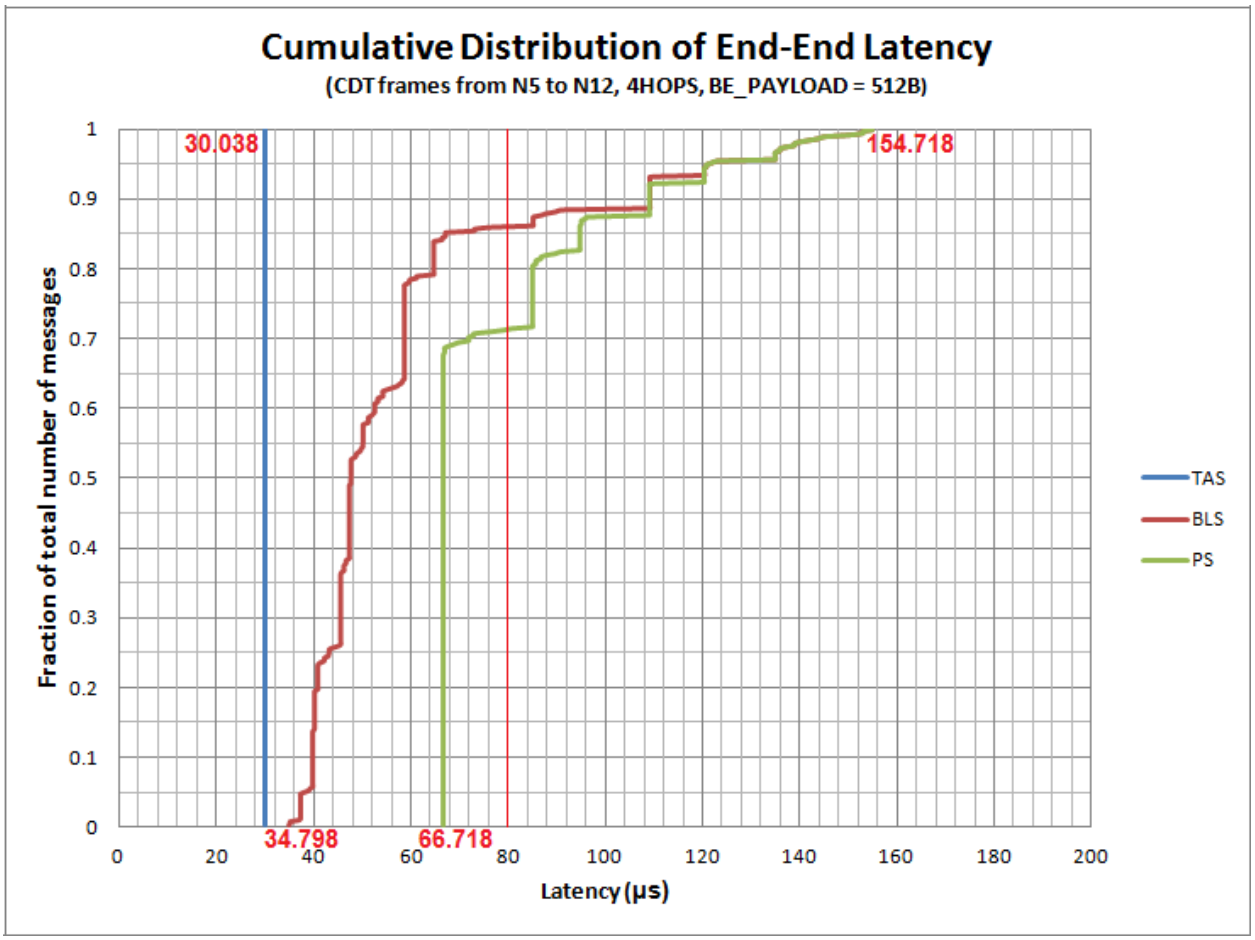
The worst case number of hops for the CDT traffic is 4 and therefore the expected end-to-end latency is 80 µs. The CDT traffic from node N5 to N12 is selected for the latency analysis. The reason for selecting this traffic is due to the receiving node N12. Node N12 receives all types of data traffic. So, the path from N5 to N12 has interference from all other data traffic.

The theoretical maximum value for end-to-end latency of the CDT frame from node N5 to N12 using Time Aware shaper with wider time slot is obtained as 35.852 µs. The theoretical maximum for Burst Limiting shaper and the improved Peristaltic shaper is same as the size of the maximum interference affecting the CDT frame (554 bytes => 44.32 µs) is greater than the *peristaltic delay* (refer section 8.2). So the Peristaltic shaper with the guard band has the same theoretical maximum end-to-end latency for the CDT frame as that of the Burst Limiting shaper. This value computed using the equation 7.5 is 168.812 µs.

9.4 Results

Graph 9.1 shows the cumulative distribution of end-to-end latency of the CDT traffic from node N5 to N12 for all the three traffic shapers. The maximum and minimum end-to-end latencies for each of the traffic shapers are highlighted in the graph. The Time Aware shaper delivers a constant end-to-end latency of 30.038 µs with no jitter. This is less than the theoretical maximum value of 35.852 µs obtained in the previous section. The Peristaltic shaper and the Burst Limiting shaper deliver an identical

maximum end-to-end latency of 154.718 µs which is less than the theoretical value of 168.812 µs obtained in the previous section.



Graph 9.1 Cumulative distribution of end-to-end latency for CDT traffic from N5 to N12 with all the three traffic shapers

Also the minimum end-to-end latency obtained using the Burst Limiting shaper is close to that of the Time Aware shaper. This indicates that the Burst Limiting shaper can match the performance of Time Aware shaper if there is no interference. The minimum end-to-end latency obtained using the Peristaltic shaper is higher than that of Burst Limiting shaper because of the inherent peristaltic delay, $\Delta peristaltic$. But the performance of the Burst Limiting shaper and the Peristaltic shaper converges towards the maximum end-to-end latency since the maximum end-to-end latency in both these shapers is affected only by the interfering frame size.

It can also be noted that the end-to-end latency using Time Aware shaper is always less than the expected maximum value of 80 µs (indicated using a thin red line in the graph). On the other hand, in the Burst Limiting shaper 85% of the CDT frames have end-to-end latencies less than the expected maximum value of 80 µs. For the Peristaltic shaper, only about 72% of the CDT frames have end-to-end latencies less than the expected maximum value. Since the Burst Limiting shaper has the maximum end-to-end latency variations (from 34.798 µs to 154.718 µs), it has the maximum jitter for the CDT traffic. The Peristaltic shaper has relatively lesser value for maximum jitter since its minimum end-to-end latency is restricted by

the peristaltic delay. The Time Aware shaper has the best performance in terms of maximum jitter since it completely avoids the interference.

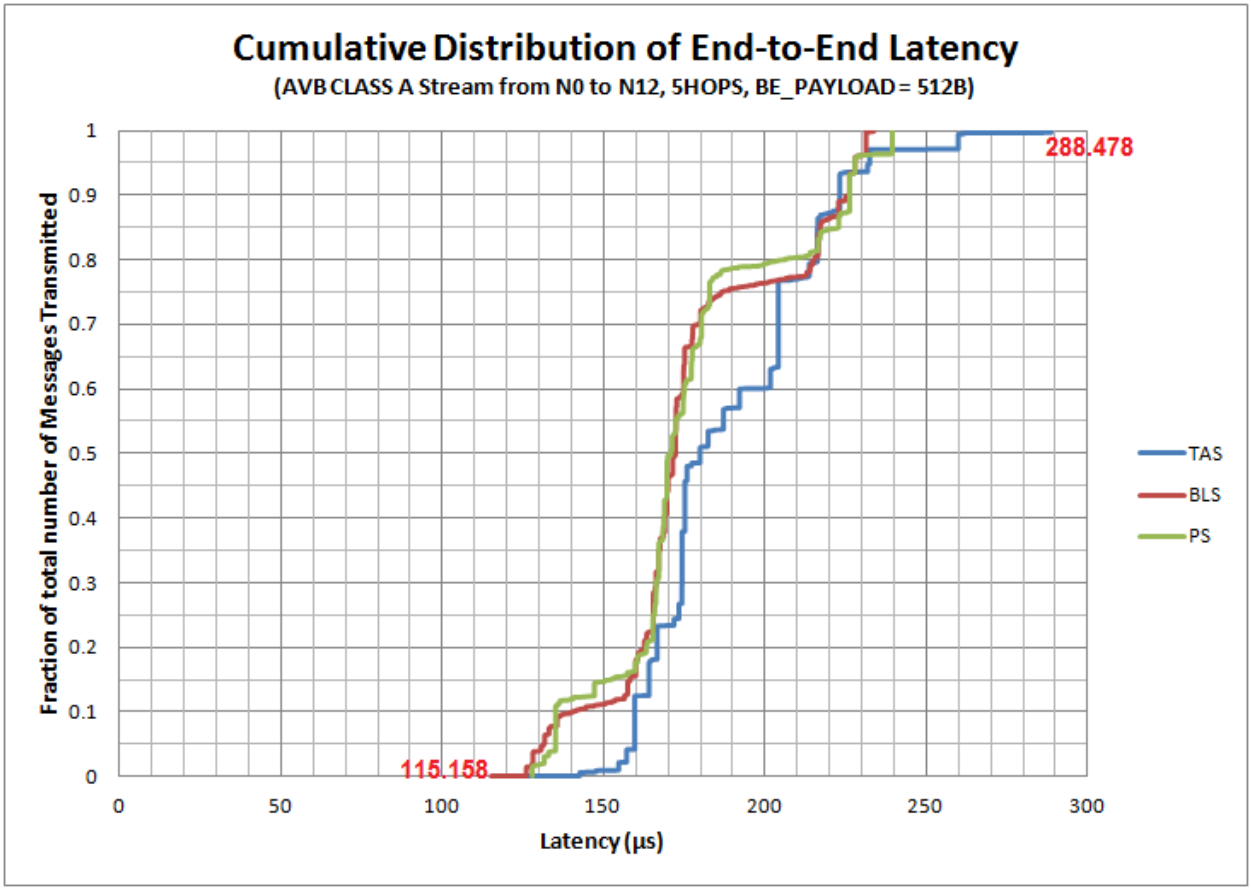
Table 9.2 the simulation results of maximum and minimum end-to-end latencies obtained using all the three traffic shapers for one of the data streams of all the four different types of data traffic. The traffics shown in the table are as follows

1. Class CDT Traffic : From N3 to N7 (4 hops)
2. Class A Traffic : From N0 to N12 (5 hops)
3. Class B Traffic: From N2 to N12 (5 hops)
4. Class BE Traffic: From N1 to N12 (5 hops)

Shaper	Traffic Class	Maximum end-to-end latency (µs)	Minimum end-to-end latency (µs)	Maximum Jitter (µs)
Time Aware Shaper	Class CDT	37.078	37.078	0.000
	Class A	288.478	124.878	163.600
	Class B	496.678	221.798	274.880
	Class BE	449.878	142.758	307.120
Burst Limiting Shaper	Class CDT	120.198	38.838	81.360
	Class A	233.558	126.038	107.520
	Class B	395.798	221.798	174.000
	Class BE	306.398	238.598	67.800
Peristaltic Shaper	Class CDT	128.478	68.478	60.000
	Class A	239.358	128.078	111.280
	Class B	409.398	221.798	187.600
	Class BE	341.518	238.598	102.920

Table 9.2 Simulation results of end-to-end latency of all traffic for all three shapers

It can be noted in Table 9.2, that the end-to-end latency requirements (refer Table 9.1) of both AVB Class A and Class B traffic are met by all the three traffic shapers. However, the three shapers for CDT traffic have a considerable impact on the AVB traffic. The CDT traffic has the highest priority in the Ethernet IVN network and the traffic shapers associated with CDT queues restrict other lower priority traffic in order to transmit the CDT traffic as soon as possible. This has a significant impact on the end-to-end latency of the rate-constrained AVB Class A and Class B traffic. Graph 9.2 shows the cumulative distribution of end-to-end latencies of one of the AVB Class A traffic in the Test Case scenario. The AVB Class A traffic taken for consideration is the traffic from node N0 to N12 (5 hops). The maximum and minimum values for end-to-end latencies are highlighted in the graph. Even though the maximum end-to-end latency value stays well below the expected latency of 1.429 ms (2ms over 7 hops), the impact of each of the traffic shapers on the latency of Class A traffic can be clearly seen.



Graph 9.2 Cumulative distribution of end-to-end latency for AVB Class A traffic from N0 to N12 with all three shapers

The maximum impact for Class A data traffic is from the Time Aware shaper. This is reasonable, since the Time Aware shaper completely avoids the interference from Class A traffic and blocks its transmission during the CDT slot. Moreover, the Class A traffic is not allowed to transmit if the time for the transmission of the entire frame exceeds the start of the subsequent CDT slot.

The Burst Limiting shaper and the Peristaltic shaper follows almost identical pattern for the impact on Class A traffic. This is due to the fact that the performance of both the improved Peristaltic shaper with guard band and the Burst Limiting shaper is affected by the interfering traffic. The Peristaltic shaper has slightly higher impact on the Class A traffic just because of the guard band. The guard band is activated when a CDT frame is received and it does not allow other interfering traffic to be selected until this CDT frame is selected for transmission. So a Class A frame which is normally allowed to transmit is blocked because of the guard band to avoid interference for the CDT frame (refer Figure 8.4). Now, the Class A frame has to wait until the CDT frame is completely transmitted. This waiting time is a maximum of 20 µs at a switch since the maximum size of the guard band is 20 µs. This adds to the end-to-end latency of the Class A frame. This can be noted from the difference between the maximum end-to-end latencies for Class A traffic in both of these shapers. So, the Burst Limiting shaper has the minimal impact on the AVB Class A data traffic. A similar distribution is seen for the AVB Class B data traffic as well.

10. Discussions and Recommendations

The overall performance of a traffic shaper (for CDT) depends not only on the end-to-end latency. To identify the best possible traffic shaper for automotive requirements, the traffic shapers considered need to be compared for other performance aspects such as configuration flexibility, hardware impact, impact on other data traffic and many more. This chapter provides a discussion on the performance of all the three traffic shapers over different aspects. Also certain key recommendations are pointed out for each of the traffic shapers. Based on the results obtained through experiments, observations made and arguments discussed, one of the traffic shapers is recommended for the automotive Ethernet IVN.

10.1 Discussions

10.1.1 End-to-End Latency for CDT

The Time Aware shaper delivers the best performance in terms of end-to-end latency for Class CDT traffic and it is the only shaper that is able to meet the IEEE802.1TSN requirements for end-to-end latency. But this is achieved only through proper scheduling adjustments. Without proper scheduling adjustments, the Time Aware shaper gives the worst performance for end-to-end latency among all the three shapers. The Burst Limiting shaper and the Peristaltic shaper are not able to meet the IEEE802.1TSN requirements for end-to-end latency due to the size of the interfering traffic. Peristaltic shaper without guard band mechanism delivers the maximum end-to-end latency for CDT traffic, only next to the Time Aware shaper without scheduling adjustments. The performance of the Peristaltic shaper with the guard band mechanism and that of the Burst Limiting shaper converge for the worst case maximum end-to-end latency as both the shapers depend on the interfering frame size for their latency performance.

10.1.2 Maximum Jitter for CDT

The Time Aware shaper always delivers a constant end-to-end latency for CDT frames with no jitter. The performance of Time Aware shaper in terms of jitter is ideal because of the assumptions that all the nodes are perfectly synchronized and there is no interference among the CDT traffic. If the synchronization error and the CDT interference are modeled, then the maximum jitter in the Time Aware shaper will be non-zero. The Peristaltic shaper has the next best performance for maximum jitter since its minimum end-to-end latency is restricted by the peristaltic delay. The Burst Limiting shaper has the least performance for maximum jitter since the end-to-end latency variation corresponds to the size of the interference at a switch. The size of the interference at a switch varies from zero to the maximum size of the interfering frame.

10.1.3 Effect of Frame Pre-emption

The frame pre-emption technique has no effect on the end-to-end latency of CDT frames in a Time Aware shaper, since it inherently avoids the interference from other data traffic. But the end-to-end latency of a non-CDT frame is improved using frame pre-emption. Without frame pre-emption, the Time Aware shaper completely blocks non-CDT frames, if their transmit time exceeds start of the subsequent CDT

time slot. With frame pre-emption, at least a fragment of the larger non-CDT frame can be transmitted before the start next CDT slot. This effectively reduces the end-to-end latency of the non-CDT frames.

However, the performance of the Burst Limiting shaper and the Peristaltic shaper can be improved using the frame pre-emption mechanism. Since the maximum size of the interfering frame determines the worst case end-to-end latency of the CDT in these two shapers, pre-empting the interfering frame reduces the end-to-end latency of CDT. The minimum size of a pre-empted Ethernet frame fragment is limited to 64 bytes [45] [46]. This reduces the maximum $\Delta interference$ to 5.12 µs (for 100 Mbps links) at all the switches. So according to equation 7.5 and the delay values assumed in the model, the maximum end-to-end latency over 5 hops for the Burst Limiting shaper is computed as 113.23 µs. This shows that the IEEE802.1TSN requirement for end-to-end latency (100 µs over 5 hops) is not met. This requirement is possible with the Burst Limiting shaper only if the payload size of the CDT data traffic is restricted to 94 bytes rather than 128 bytes. Frame pre-emption also reduces the maximum jitter delivered by the Burst Limiting shaper for CDT traffic as it limits the maximum end-to-end latency.

Considering the Peristaltic shaper, the maximum $\Delta interference$ (5.12 µs) after pre-emption is less than the peristaltic delay, $\Delta peristaltic$. So, the maximum end-to-end latency of a CDT frame in a peristaltic shaper is influenced only by the peristaltic phase width (20 µs). The theoretical maximum end-to-end latency of a CDT frame over 5 hops is computed according to equation 7.7 as 172.75 µs. This is much higher than the expected 100 µs. The CDT frame size restriction or the peristaltic phase width reduction does not have a significant improvement in the latency delivered by the Peristaltic shaper. The only option for improving the Peristaltic shaper is to increase the link speed from 100 Mbps to 1 Gbps. But with the link speed of 1 Gbps, the maximum CDT payload size restriction increases to 256 bytes according to IEEE802.1TSN requirements [8]. Now with equation 7.7, the theoretical maximum end-to-end latency of a CDT frame over 5 hops is obtained as approximately 25 µs using a Peristaltic shaper in 1Gbps Ethernet IVN. This is well below the required 100 µs limit.

10.1.4 Addition of New CDT Streams

The automotive Ethernet IVN requires flexibility in adding and removing new CDT streams dynamically. Though this is handled by the IEEE802.1Qat (Stream Reservation Protocol) [35], the impact of dynamic addition of new CDT streams on the traffic shapers needs to be studied. The Time Aware shaper works on statically configured schedule. So, an addition of new CDT stream requires modification in the schedule table entries at all nodes involved in the path of the new CDT stream. This becomes a seriously complex task for the Stream Reservation Protocol because problems like dynamic schedule computation and distribution need to be addressed.

However, addition of new CDT streams does not affect the Burst Limiting shaper much since it works on the same Credit Based shaping principles. The Stream Reservation Protocol used for dynamic reservation of AVB Class A and Class B streams in IEEE802.1AVB protocol can be used with minor modifications.

On the other hand, addition of new CDT streams has no effect on the Peristaltic shaper. Since the Peristaltic shaper uses only the ingress time stamp to compute the peristaltic phase, adding a new CDT stream requires no modification in the configuration parameters of the Peristaltic shaper.

10.1.5 CDT Streams with Multiple Transmission Intervals

The control data traffic in the present day automotive network has streams with multiple transmission intervals [9]. This enforces the need to configure the CDT streams with different transmission intervals. The schedule for Time Aware shaper is configured based on the arrival of each of the CDT streams for one complete CDT period (500 µs). This schedule is repeated at an interval of one CDT period. To include streams with multiple CDT periods, the schedule needs to be extended to include the arrival of all CDT streams. The repetition interval of the schedule is a common CDT period which is given by the least common multiple of all the different CDT periods. This increases the size of the schedule table as the common repetition interval increases for new CDT streams with greater transmission intervals.

The Burst Limiting shaper limits the maximum burst of CDT data traffic within a CDT period. The configuration of the maximum level for the CDT burst depends on the CDT transmission period. However, if there are CDT streams with multiple transmission periods, a common transmission period determines the maximum burst. This common transmission period is given by the least common multiple of all different CDT periods. This implies that the maximum burst limit increases with the addition of new CDT streams with multiple transmission intervals.

The operation of the Peristaltic shaper does not depend on the CDT transmission interval. Also, there is no configuration parameter involving the CDT transmission interval. This makes the operation of the Peristaltic shaper completely independent of the CDT transmission interval. So, the Peristaltic shaper inherently supports CDT streams with multiple transmission intervals.

10.1.6 Bandwidth Utilization

The width of the CDT slot determines the bandwidth allocated for the CDT traffic in a Time Aware shaper. Scheduling adjustments play a vital role in bandwidth utilization. The link wise staggered scheduling utilizes the total bandwidth efficiently. But the schedule with a wider CDT slot wastes the useful bandwidth available for other data traffic in order to achieve better end-to-end latency for the CDT traffic.

The Burst Limiting shaper shares the bandwidth according to the configuration. The maximum burst level for the CDT traffic limits the over usage of bandwidth by the CDT traffic. So, other data traffic gets their configured share of the total bandwidth.

The Peristaltic shaper does not have a mechanism to limit the bandwidth usage of the CDT traffic. In case of misbehaving CDT Talker, the CDT traffic over uses the bandwidth and due to this other low priority data traffic could suffer bandwidth starvation. For example, if there are more CDT frames to be transmitted in a single peristaltic out phase and if the sum of frame sizes of all these frames exceeds the size of a single peristaltic out phase, they are still transmitted continuously one after the other. This is because the Peristaltic shaper holds a CDT frame only for the peristaltic delay time. If this time expires for a particular CDT frame, it is selected to be transmitted before other data traffic.

10.1.7 Impact on hardware

The traffic shapers are mechanisms to be integrated as hardware functional units in each of the network elements (both switches and end nodes) of the Ethernet IVN. It is crucial that the traffic shapers have minimal impact on the hardware to which they are integrated.

The performance of Time Aware shaper highly depends on perfect synchronization of all the nodes in the network. So, the hardware time synchronization protocol described in IEEE802.1ASbt is essential for the working Time Aware shaper. Apart from this, the schedule table at each node requires memory. Each entry in the schedule table, called the *gateEvent* is a pair of the width of the time slot and the gate states of all the output queues during this time slot (refer Figure 5.3). The width of the time slot takes 10 bytes since time is represented using 10 bytes in IEEE802.1AS protocol [34]. The gate states can be represented in a single byte with each bit representing the open or closed state of one of the eight available queues in an output port. So, the memory requirement increases in a multiple of 11 bytes with the increase in the size of the schedule table. The Time Aware shaper also requires logical units to act as gates and logic circuitry for driving these gates. These features require considerable hardware modifications in the existing IEEE802.1AVB based switches and end nodes.

The principle of operation of the Burst Limiting shaper aligns to that of the Credit Based shaper specified by the IEEE802.1AVB standard. The computational units for incrementing and decrementing the credits in the Credit Based shaper, can be re-used for the Burst Limiting shaper. The only addition is the signal required to change the priority of the CDT queue when the credit reaches the *max_level* (refer Figure 5.2). This is relatively a minor hardware modification to the existing IEEE802.1AVB based switches and end nodes.

The Peristaltic shaper requires the implementation of time stamping mechanisms at the input ports in order to compute the peristaltic input phase (*pRphase*) for the incoming CDT frames. Also a highly synchronized time provided by hardware time synchronization protocol IEEE802.1Asbt is required to compute the peristaltic output phase (*pTphase*). A CDT frame received in the odd phase is transmitted in the even phase and vice versa. A simple solution to implement this mechanism at the output ports is to use two separate queues for the CDT namely odd and even queue. The CDT frames are placed in either odd or even CDT queue based on their *pTphase*. A phase trigger mechanism should be included to switch the selection of frames from odd and even CDT queues for transmission after every peristaltic phase width. All these mechanisms require major hardware modifications in the existing IEEE802.1AVB switches and end nodes.

The length of the CDT queue required is minimal for all the three CDT shapers as compared to the queue sizes of other data traffic. This is due to the fact that CDT frames have smaller size, higher priority and all the CDT shapers try to transmit the CDT frames as soon as possible. This implies that the maximum size of the CDT queue depends on the incoming CDT traffic and the maximum residence time of a CDT frame. This shows that the CDT queue can be relatively small compared to the AVB Class A or Class B queues.

To summarize, the Time Aware shaper and the Peristaltic shaper have relatively higher impact on the existing hardware than the Burst Limiting shaper.

10.1.8 Overall Comparison

Table 10.1 shows the overall comparison of all the three traffic shapers for different aspects. Each of the aspect is given a star rating based on the observation and the results obtained. The star rating has a maximum value of 4. The overall performance ranks the three shapers based on the total performance ratings.

Aspect	Time Aware Shaper	Burst Limiting Shaper	Peristaltic Shaper
End-to-End Latency	★★★★	★★★	★
Maximum Jitter	★★★★	★	★★
Flexibility & Configurability	★	★★★	★★★★
Minimal Hardware Impact	★	★★★★	★★
Overall Performance	★★	★★★	★

Table 10.1 Comparison ratings for all the three CDT shapers on different performance aspects

Though the Time Aware shaper gives the best possible end-to-end latency and maximum jitter for the CDT, it offers very little flexibility and has high configuration complexity and hardware impact. On the other hand, it is very easy to configure a Peristaltic shaper and it is highly flexible for the addition of new CDT streams but it is not possible to achieve the required end-to-end latency for CDT streams in a 100Mbps network. Also it requires major modifications in the hardware. The Burst Limiting shaper has a minimal hardware impact since it re-uses most of the hardware units from the Credit Based shaper. Also it provides flexibility to dynamically add new CDT streams through existing Stream Reservation Protocol. Though the end-to-end latency is not achieved for the IEEE802.1TSN frame sizes, with additional frame size restrictions on CDT and with frame pre-emption the Burst Limiting shaper can meet the end-to-latency requirements with a limited maximum jitter. So, considering the overall performance the Burst Limiting shaper is better than the other two traffic shapers.

10.2 Recommendations

Based on the simulation results and the arguments in the previous section, the following recommendations are listed for the use of the three traffic shapers considered

1. The implementation of hardware time synchronization protocol (IEEE802.1ASbt) is essential for both Time Aware shaper and Peristaltic shaper.
2. A protocol to compute the staggering of the schedule and to distribute it to different nodes is important for the effective functioning of the Time Aware shaper.
3. Implementation of frame pre-emption mechanism is required to achieve end-to-end latency in a Burst Limiting shaper.
4. Implementation of the guard band mechanism is recommended to reduce the end-to-end latency of a CDT frame in a Peristaltic shaper.
5. A link speed of 1Gpbs is required to meet the end-to-end latency requirements of IEEE802.1TSN using a Peristaltic shaper.

11. Conclusion

The primary goal of this project was to compare three different CDT traffic shaping mechanisms proposed by IEEE802.1TSN standardization committee for the automotive Ethernet IVN and to verify the possibility to achieve the specified end-to-end latency requirements for Control Data traffic (100 µs over 5 hops). The details of the three traffic shaping mechanisms namely, the Time Aware shaper, the Burst Limiting shaper and the Peristaltic shaper are collected from the IEEE802.1TSN standardization committee presentations. These details are organized and formulated as corresponding traffic shaping algorithms. The algorithms are implemented in SystemC and integrated as plug-n-play software modules to an existing IEEE802.1AVB based automotive Ethernet IVN simulator available at NXP. Implementation in compliance with the evolving IEEE802.1TSN standard is achieved. A mathematical analysis of worst case behavior of all the three shapers is presented. The simulation results for end-to-end latencies are validated against the mathematical calculations. Based on theoretical analysis, improvements for enhancing the worst case end-to-end latency of a CDT frame are implemented for Time Aware shaper and Peristaltic shaper. A realistic Ethernet IVN scenario is simulated to evaluate the performances of all the three traffic shaper models. The simulation results are analyzed and the three traffic shaping mechanisms are compared for performance on different aspects.

The simulation results show that it is possible to achieve the IEEE802.1TSN latency requirements only using the Time Aware shaper with proper scheduling. The Burst Limiting shaper and the Peristaltic shaper are not able to meet the IEEE802.1TSN requirements for end-to-end latency due to the size of the interfering traffic. But further analysis shows that these two traffic shapers too can meet the IEEE802.1TSN requirements on certain conditions. The Burst Limiting shaper can meet the IEEE802.1TSN latency requirements with frame pre-emption and additional frame size restrictions. Also, the Peristaltic shaper can meet the latency requirements with increased link speeds. Considering these improvements and on the basis of overall performance on end-to-end latency for CDT, configuration complexity and hardware impact, the Burst Limiting shaper is recommended for implementation in automotive Ethernet IVN.

12. Future Work

The traffic shaping mechanism for Control Data traffic in automotive Ethernet IVN is under standardization process. Each of the traffic shaping mechanisms proposed by the IEEE802.1TSN standardization committee has its own advantages and disadvantages. All the three traffic shaping mechanisms considered in this thesis work have the potential for further enhancements to suit specific automotive requirements. This thesis work provides a basic simulation platform where these enhancements can be implemented for further comparison. Some of the general improvements for this thesis work are listed below

1. The hardware synchronization protocol (IEEE802.1ASbt) [47] is to be included in the simulation model to obtain the study the performance of Time Aware shaper and Peristaltic shaper under synchronization errors
2. The frame pre-emption mechanism (IEEE802.1Qbu) [46] is to be implemented to study the end-to-end latency performance improvement in Burst Limiting shaper and Peristaltic shaper
3. The Stream Reservation Protocol (IEEE802.1Qcc) [48] needs to be included in the model for dynamic stream reservation
4. The application module for the end nodes need to be improved to facilitate the generation of multiple data traffic and also the generation of response to the incoming frames
5. The effect of CDT streams with multiple transmission periods needs to be evaluated
6. The impact of interference from other CDT traffic due to simultaneous transmission from different CDT Talkers needs to be analyzed
7. The performance of all the three traffic shapers for networks with 1Gbps link needs to be evaluated

Bibliography

[1] E. Eckermann, in *World History of the Automobile*, SAE Press, 2001, p. 14.

[2] N. Navet, Y. Song, F. Simonot-Lion and C. Wilwert, "Trends in Automotive Communication Systems," in *Proceedings of the IEEE, Vol. 93, No. 6, pp. 1204-1224*, 2005.

[3] H. Lim, L. Volker and D. Herrscher, "Challenges in a future IP/Ethernet-based in-car network for real-time applications," in *In Proceedings of the 48th Design Automation Conference (pp. 7-12). ACM.*, June 2011.

[4] Micrel, "Micrel and Marvell Deliver World's First Standards-Compliant Ethernet PHYs for In-Vehicle Networking," 24 September 2012. [Online]. Available: http://investor.micrel.com/releasedetail.cfm?ReleaseID=708948.

[5] "History Of Ethernet," [Online]. Available: http://standards.ieee.org/events/ethernet/history.html.

[6] A. Abaye, "From OPEN Alliance BroadR-Reach® PHYs to IEEE802.3bp RTPGE," [Online]. Available: http://standards.ieee.org/events/automotive/03_Tazabay_Broadcom_RTPGE.pdf.

[7] M. Teener Johas, A. Fredette, C. Boiger, P. Klein, C. Gunther, D. Olsen and K. Stanton, "Heterogeneous Networks for Audio and Video: Using IEEE 802.1 Audio Video Bridging," in *Proceedings of the IEEE, 101(11), 2339-2354*.

[8] D. Pannell, "Automotive IVN specifications for IEEE802.1TSN," [Online]. Available: http://www.ieee802.org/1/files/public/docs2013/avb-pannell-gen2-assumptions-1113-v17.pdf.

[9] M. Joachim, "Automotive industry requirements for IEEE802.1TSN," [Online]. Available: http://www.ieee802.org/1/files/public/docs2013/new-tsn-jochim-aaa2c-requirements-for-control-traffic-0713-v01.pdf.

[10] P. Meyer, T. Steinbach, F. Korf and T. C. Schmidt, "Extending IEEE 802.1 AVB with time-triggered scheduling: A simulation study of the coexistence of synchronous and asynchronous traffic," in *VNC (pp. 47-54)*, 2013.

[11] SAE - AS-2D Time Triggered Systems and Architecture Committee, "Time-Triggered Ethernet (AS 6802)," 2009. [Online]. Available: http://standards.sae.org/as6802/.

[12] IEEE 802.1 AVB Task Group, "IEEE 802.1 Audio/Video Bridging (AVB)," [Online]. Available: http://www.ieee802.org/1/pages/avbridges.html.

[13] R. Marau, L. Almeida and P. Pedreiras, "Enhancing real-time communication over cots ethernet

switches," in *WFCS (Vol. 6, pp. 295-302)*, June 2006.

[14] G. Alderisi, A. Caltabiano, G. Vasta, G. Iannizzotto, T. Steinbach and L. L. Bello, "Simulative Assessments of IEEE 802.1 Ethernet AVB and Time-Triggered Ethernet for Advanced Driver Assistance Systems and In-Car Infotainment," in *VNC*, 2012.

[15] H. T. Lim, D. Herrscher and F. Chaari, "Performance comparison of ieee 802.1 q and ieee 802.1 avb in an ethernet-based in-vehicle network," in *Computing Technology and Information Management (ICCM), 8th International Conference on (Vol. 1, pp. 1-6)*, 2012.

[16] F.-J. Goetz, "Traffic Shaper for Control Data Traffic," 15 July 2012. [Online]. Available: http://www.ieee802.org/1/files/public/docs2012/new-goetz-CtrDataScheduler-0712-v1.pdf.

[17] IEEE 802.3 Residential Ethernet Study Group, "IEEE 802.3 Residential Ethernet," [Online]. Available: http://www.ieee802.org/3/re_study/index.html.

[18] IEEE Std., " IEEE Standard for a High-Performance Serial Bus 1394-2008," IEEE, 2008.

[19] L. Zhang, "Virtual Clock: A New Traffic Control Algorithm for Packet Switching Networks," in *Proceedings of the ACM Symposium on Communications Architectures*, 1990.

[20] J. Specht, "Urgency Based Scheduler," December 2013. [Online]. Available: http://www.ieee802.org/1/files/public/docs2013/new-tsn-specht-ubs-avb1case-1213-v01.pdf.

[21] C. Boiger, "How Many Transmission Selection Algorithms Do We Need?," May 2013. [Online]. Available: http://www.ieee802.org/1/files/public/docs2013/new-tsn-boiger-new-shaper-0513.pdf.

[22] F.-J. Götz, "Alternative Shaper for Scheduled Traffic in Time Sensitive Network," January 2013. [Online]. Available: http://www.ieee802.org/1/files/public/docs2013/new-goetz-TSN-4-Industrial-Networks-20130115-v1.pdf.

[23] D. Pannell, "AVB Gen2- Latency Improvement Options," March 2011. [Online]. Available: http://www.ieee802.org/1/files/public/docs2011/new-pannell-latency-options-0311-v1.pdf.

[24] D. Pannell, "AVB-Generation 2 Latency Improvement Options," November 2011. [Online]. Available: http://www.ieee802.org/1/files/public/docs2011/new-avb-pannell-latency-options-1111-v2.pdf.

[25] C. Boiger, "Class A Latency Issues," January 2011. [Online]. Available: http://www.ieee802.org/1/files/public/docs2011/ba-boiger-class-a-latency-issues-0111.pdf.

[26] C. Boiger, "Class A Bridge Latency Calculations," November 2010. [Online]. Available: http://www.ieee802.org/1/files/public/docs2010/ba-boiger-bridge-latency-calculations.pdf.

[27] M. Kießling, "Some Sources of Latency and Jitter," May 2013. [Online]. Available:

http://www.ieee802.org/1/files/public/docs2013/bv-kiessling-Some-sources-of-Latency-and-Jitter-0513-v01.pdf.

[28] R. Cummings, "802.1Qbv: Performance / Complexity Tradeoffs," September 2012. [Online]. Available: http://www.ieee802.org/1/files/public/docs2012/Qbv-cummings-performance-tradeoffs-0912-v2.pdf.

[29] M. Rahmani, K. Tappayuthpijarn, B. Krebs, E. Steinbach and R. Bogenberger, "Traffic shaping for resource-efficient in-vehicle communication," in *Industrial Informatics, IEEE Transactions on, 5(4), 414-428*, 2009.

[30] "IEEE Standard for local and metropolitan area networks — Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks," IEEE Std 802.1Q-2011.

[31] "IEEE Standard for local and metropolitan area networks-Media Access and Control(MAC) Bridges and Virtual Bridged Local Area Networks," IEEE Std 802.1Q-2011.

[32] "Introduction to Livewire Systems," [Online]. Available: http://axiaaudio.com/tech/.

[33] "IEEE 802.1 AVB Task Group," [Online]. Available: http://www.ieee802.org/1/pages/avbridges.html.

[34] "IEEE Standard for Local and Metropolitan Area Networks—Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks," IEEE Std 802.1AS-2011.

[35] "IEEE Standard for Local and Metropolitan Networks – Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP)," Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005).

[36] "IEEE Standard for Local and Metropolitan Networks – Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams," IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Qav-2009).

[37] "IEEE Standard for Layer 2 Transport Protocol for Time Sensitive Applications in a Bridged Local Area Network," IEEE Std 1722-2011.

[38] S. Kerschbaum, F.-J. Goetz and F. Chen, "Towards the Calculation of Performance Guarantees for BLS in Time-Sensitive Networks," November 2013. [Online]. Available: http://www.ieee802.org/1/files/public/docs2013/bv-kerschbaum-TSN-Latency4CDT-112013-v4.pdf.

[39] "IEEE Standard for Bridges and Bridged Networks— Amendment: Enhancements for Scheduled Traffic," IEEE P802.1Qbv/D1.1.

[40] M. Johas Teener, "Peristaltic Shaper in Clause 8 style," [Online]. Available: http://www.ieee802.org/1/files/public/docs2013/avb-tj-peristaltic-shaper-in-clause-8-style-0313-

v1.pdf.

[41] M. Johas Teener, "Peristaltic Shaper:updates,multiple speeds," January 2014. [Online]. Available: http://www.ieee802.org/1/files/public/docs2014/new-tsn-mjt-peristaltic-shaper-0114.pdf.

[42] D. C. Black, J. Donovan, B. Bunton and A. Keist, SystemC: From the ground up, Boston,MA: Kluwer Academic Publishers, 2004.

[43] K. Steinhammer, P. Grillinger, A. Ademaj and H. Kopetz, "A time-triggered ethernet (TTE) switch," in *Proceedings of the conference on Design, automation and test in Europe: Proceedings (pp. 794-799). European Design and Automation Association*, 2006.

[44] L. Zhang, D. Goswami, R. Schneider and S. Chakraborty, "Task-and network-level schedule co-synthesis of Ethernet-based time-triggered systems," in *ASP-DAC (pp. 119-124)*, 2014.

[45] F.-J. Goetz, "AVB Extensions for Industrial Communications," January 2011. [Online]. Available: http://www.ieee802.org/1/files/public/docs2011/new-goetz-avb-ext-industrcom-0113-v01.pdf.

[46] IEEE802.1TSN Task Group, "IEEE802.1Qbu - Frame Preemption," [Online]. Available: http://www.ieee802.org/1/pages/802.1bu.html.

[47] IEEE 802.1TSN Task Group, "IEEE802.1ASbt - Timing and Synchronization: Enhancements and Performance Improvements," [Online]. Available: http://www.ieee802.org/1/pages/802.1asbt.html.

[48] IEEE 802.1 TSN Task Group, "IEEE802.1Qcc - Stream Reservation Protocol (SRP) Enhancements and Performance Improvements," [Online]. Available: http://www.ieee802.org/1/pages/802.1cc.html.