Ethernet Communication for Real-time Automotive applications (ECRA)

Diary Number:

Project within: FFI Vehicle Development

Author: Hoai Hoang Bengtsson, Volvo Car Corporation

Samuel Sigfridsson, Volvo Car Corporation

Date: 2019-06-17

Contents

FFI in short

FFI is a partnership between the Swedish government and automotive industry for joint funding of research, innovation and development concentrating on Climate & Environment and Safety. FFI has R&D activities worth approx. €100 million per year, of which half is governmental funding. The background to the investment is that development within road transportation and Swedish automotive industry has big impact for growth. FFI will contribute to the following main goals: Reducing the environmental impact of transport, reducing the number killed and injured in traffic and Strengthening international competitiveness. Currently there are five collaboration programs: **Vehicle Development, Transport Efficiency, Vehicle and Traffic Safety, Energy & Environment and Sustainable Production Technology.**

For more information: www.vinnova.se/ffi

1. Executive summary

The Ethernet Communication for Real-time Automotive applications (ECRA) project started in Jannuary 2016 and finished in December 2018. The project has its basis on the FFI projects ECAE (Dnr 2012-03676) and AVB (Dnr 2013-04722). The ECRA project focused on the recent advanced Ethernet technologies such as Time Sensitive Networking protocols and 1000BASET1, and their applications in Automotive industry. The ECRA project covers both Ethernet hardware, software and real automotive use cases.

The project consisted of five partners including one OEM, three SMEs (Time Critical Network, Pelagicore and ArcCore) and one research institute, Rise Viktoria. Volvo Cars Corporation (VCC) is the only OEM in the project, which is also the project coordinator. VCC also led two work pakages: WP1 and WP4. WP1 focused on application requirements and WP4 on Ethernet HW. Time Critical Network (TCN) ,one of the SMEs in the project, led WP3 and worked with tools in the area of timing analysis for the automotive network system. Pelagicore contributed with Ethernet software implementation experiements and led WP3. WP5 is about real-time analysis and simulations for time sensitive networking. WP5 was led by Rise Viktoria, the research institute of Sweden. ArcCore contributed with software for performance measurements of Ethernet software. Pelagicore were acquired by Luxoft in September 2016.

The results of the project has varied from theoretical analysis, simulation tools to practical implementations of Ethernet HW/SW for automotive use cases. Several proof of concepts about using time sensitive Ethernet for audio applications in vehicle have been done. In the project it has been shown that, using Ethernet as a core network in the vehicle is not only a trend but actually feasible and will be implemented in the next generation of Volvo Car's products.

2. Background

Ethernet has been potential to be the dominant technology for automotive industry. Ethernet has not only been used in the infotainment applications but also become the main candiates for the backbone network inside vehicle as well as connect vehicle with the outside world.

Moreover, moving towards service oriented architectures is a rising trend among many vehicle manufactures. OEMs need to cope with an increasing number of applications, new functionalities, new features and new ways of using cars. That puts strong motivation and challenges for OEMs to introduce the new standard Time Sensitive Networking (TSN) for vehicle communication.

Two previous FFI project about Ethernet has been done with mainly the same partners. These projects provided a very good background as well as technical preparation for the ECRA project. The challenges when using standard Ehernet technology, as defined in the IEEE 802.3 standard in in-vehicle networks as well as using Audio Video Bridging (AVB) protocols in infortainment have been investigated. All partners are familier with the subject, and has worked well together for a long time.

Rise Viktoria was the only academic partner in the project, which is actively participating in the transportation research. Viktoria contributed with important competence in the area of real-time communication in Ethernet for safety critical applications.

3. Objectives

Increased use of in-vehicle Ethernet as a foundation for enhanced functions and services in the areas of active safety, autonomous driving and infotainment is expected to:

- Strengthen the position of the Swedish OEMs compared to international competitors
- Increase the competence level in the Swedish automotive industry as a whole

The main project objective is to *support efficient implementation of reliable, robust and real-time communication over switched Ethernet networks for future advanced automotive applications.* To achieve this the following supporting project objectives are defined:

- Deliveries and Milestones as defined in section "Schedule" below in this document.
- Arrange at least two Project Seminars during the course of the project (see section "Project management" for more information).
- Produce at least two Master Thesis.
- Produce at least two publications for related international conferences or journal.

Comment: We did not have any master thesis connected to the ECRA project but we have recuited two new employees at VCC: one person with competence in real-time Ethernet and communication networking, one person with experience in Ethernet software development.

4. Project realization

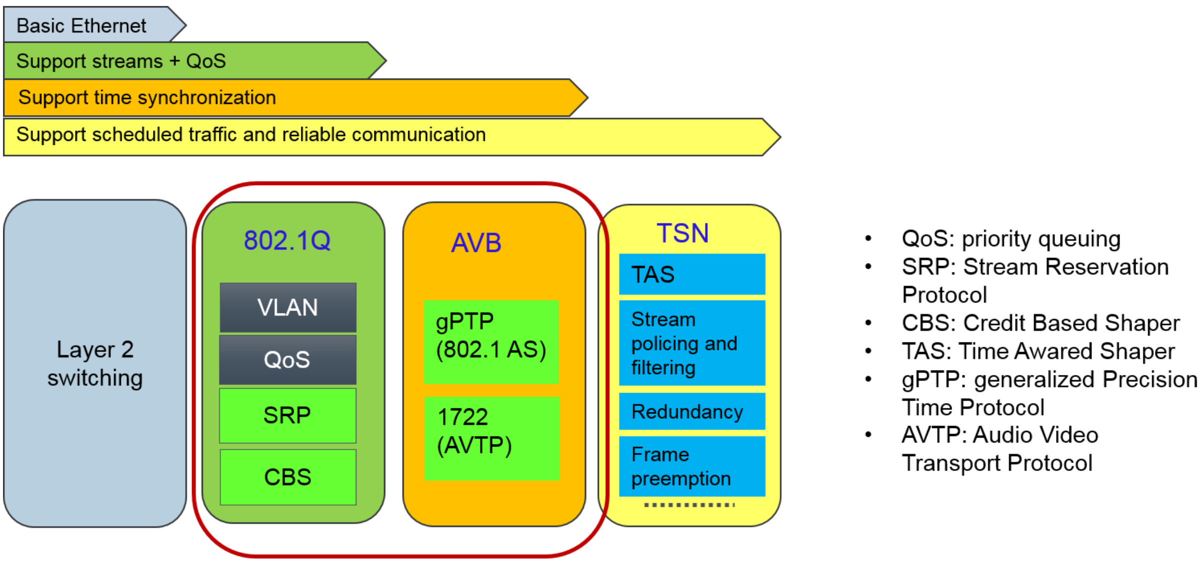


Figure 1. Ethernet Evolution: from legavy Ethernet to AVB to TSN

Figure 1 [9] shows the evolution and the current state of Ethernet standard to achive reliable deterministic communication. The basic switching features offered by IEEE802.1D standard were not sufficient for time critical traffic in the automotive domain. 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. 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.

As the standardization moved towards AVB 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. 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. 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. 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. More detail about the Ethernet related work can be found in the master thesis done by Sivakumar Thangamuthu at Eindhoven University and NXP [9].

The project direction has been following the trend of Automotive Ethernet development. All the partners have work with time sensitive Ethernet at different aspects and different levels.

4.1 WP1: Requirements and use cases (VCC)

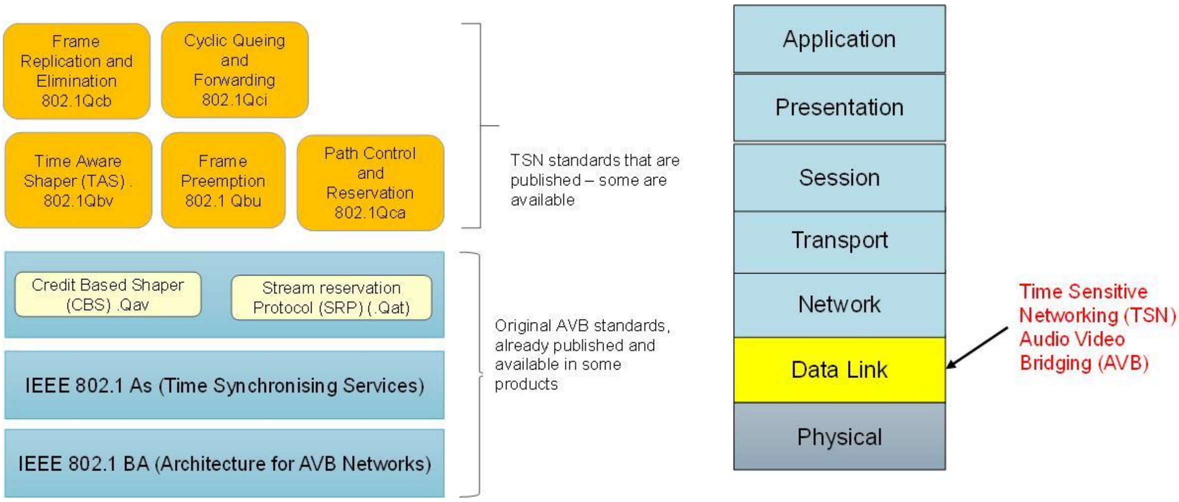


Figure 2. Overview of Time Sensitive Network Protocol stack.

The objective of the WP1 is to identify, analyze and consolidate requirements and uses cases serving as a basis for the project. This is achieved by identifying specific feature(s) that is most probable to be introduced in the market in coming years. The set of requirements needed for these features, technologies, protocols and concepts that are evolving in Autosar, IEEE and OPEN, will be evaluated, adapted and applied in an automotive context.

4.1.1 Theoretical analysis of concepts in Autosar

Purposes of this task is to analysis concepts such as EthTSync, EthSwitch, EthDriver, LargeCOM, etc. used for Ethernet communication and collection of requirements needed for both Autosar and non Autosar ECUs to support these concepts

Comment: A limited effort has been done in the project with focus on Ethernet concepts in Autosar. It has mainly been the SOME/IP and SOME/IP Service Discovery protocols and serialization that has been studied and incorporated in the set of requirements delivered in WP1.

In additional to what have been described in the project description, we have done a comprehensive study about the Time Sensitive Networking protocols and how they are supported in the Autosar stack. The result of this task is a set of requirements for Ethernet ECUs in our next vehicle platform – result can be found in deliverable [8] (LC-TSN) including:

- Time synchronization protocols: generalized precision time protocol gPTP as defined in the IEEE 802.1 AS [3]
- Stream Reservation Protocol (SRP) for time sensitive applications as defined in the [IEEE 802.1 Qat [4]
- Forward and Queuing for Time Sensitive Streams (FQTSS) as defined in the [IEEE 802.1 Qav [5]

Figure 2 illustrates protocols included in the IEEE TSN protocol stack. Results of this task has been presentated at the third project seminar and also Automotive Ethernet Congress 2019 [3].

4.1.2 Theoritical analysis of concepts for fault tolerant systems in the vehicle network

This task is about theoretical analysis of concepts that can be used to achieve a fault tolerant system in the network. Protocols used to achieve redundancy will be evaluated, e.g. PRP - Parallel Redundancy Protocol and HSR - High Availability Seamless Redundancy

Comment: This task has been done together with Rise Viktoria. Main task has been to investigated the Frame Replication and Elimination for reliability IEEE 802.1 Qcb. Two reports have been produced by Rise Viktoria and VCC. IEEE 802.1Qcb required a network with ring topology while we mainly focused on the centralized architecture. Therefore, it require a lot of modification in order to apply the protocol for our topology.

4.1.3 Ethernet physical layers for robust communication system

Theoretical analysis of different physical layers that can be used in a robust communication system, e.g. IEEE 1000BASE-T1, GEPOF, and PoDL

Comment: The analysis has focused on different physical layers that could be suitable for gigabit Ethernet networking. 1000BASE-T1, and GEPOF (or 1000BASE-RH)have been studied. HDBaseT is a third possible technology that could be used ant that partly was studied together with the other two technologies. The theoretical analysis has included meetings with providers of these technologies, studies of materials such as data sheets and presentations and resulted in a report.

4.2 WP2: Tool Development and Assessment (TCN)

In WP2 TCN's simulation tool, TCN TimeAnalysis™, has been extended with improved possibilities to model and simulate ECU task management performed by the RTOS scheduler in an automotive ECU conforming to the OSEK/VDX specification. Together with the previously added simulation models that describe the behavior of the Ethernet, CAN and LIN transmission protocols, TCN simulation engine is now much closer for being used to perform a holistic timing analysis of an automotive electrical architecture, which was an important objective of task T2.1.

The following summarizes the task management modeling improvements that have been implemented during T2.1.

4.2.1 Task management modeling improvement and implementation

- **Task scheduling and task states**

The control software inside an ECU typically must perform different monitoring and computational duties exhibiting real-time requirements, which collectively can be modeled in the form of tasks.Tasks execute one or more software functions and the order in which different tasks are executed by the OSEK operating system is determined by the scheduler. The extended task model of the OSEK operating system was implemented to simulate task scheduling. The extended task model is comprised of four different task states:

- **running**
  In the running state, the CPU is assigned to the task, so that its instructions can be executed. Only one task can be in this state at any point in time, while all the other states can be adopted simultaneously by several tasks.
- **ready**
  All functional prerequisites for a transition into the running state exist, and the task only waits for allocation of the processor. The scheduler decides which ready task is executed next.
- **waiting**
  A task cannot continue execution because it shall wait for at least one event
- **suspended**
  In the suspended state the task is passive and can be activated.

- **Task priorities**

Task priorities can be used to influence the scheduler's task management by giving certain tasks precedence when the scheduler is to select a task that is ready to be transferred to the running state. Task priorities are now supported in TCN's simulation engine by the introduction of a task FIFO queue for each priority level into which tasks are sorted when they transition to the ready state.

- **Periodic tasks**

Periodically activated tasks are frequently used in traditional hard real-time applications and automotive ECU implementations, so this was considered the most basic task model to support when adding task modeling to TCN TimeAnalysis™.

- **Chained tasks**

Besides periodically activated tasks, OSEK also allows tasks to be chained together thereby ensuring, that a specified, succeeding task is activated just after the running task is terminated. Chaining itself thus transfers a specific successor task from the suspended state into the last element of one of the ready state priority queues. After this the scheduler performs a rescheduling.

- **Task execution times**

During a simulation, when a task is selected by the scheduler to run on a CPU, the task will then occupy the CPU for a duration of time equal to its assigned execution time. As a first approximation, a task can be assigned a constant execution time, which is selected solely based on the user's intuition or experience.

However, as a result of work performed in WP2, a statistical collection of execution time measurements can now be used as a probabilistic execution time model for a specific task. Then, every time such a task is transferred to the running state during a simulation, a new execution time is selected from the statistical distribution. This can give a much more realistic execution time model than just a constant value.

- **Task interrupts**

During the ECRA project, support for modeling a specific category of interrupts have been added to TCN's simulation engine in which the CPU continues processing the original task that were interrupted exactly where it left off when the interrupt occurred and thus no task rescheduling occurs.

4.3 WP3: Efficient Implementation (Pelagicore/ Luxoft)

The complex and heterogeneous nature of in-vehicle network systems and their evolution requires a deep, holistic understanding in order to identify requirements on future use-cases. Based on identified corner use-cases, we propose to model end-to-end configurations of the entire network data path, from source to sink. The model should traverse the entire data path and demonstrate for instance, use-cases such as rear view cameras and mirrorless side-mirrors. The model shall incorporate requirements, challenges and options for ECU-internal hardware- and software as well as for the network topology and protocols that connects the ECUs

4.3.1 ECU internal hardware and software architecture

In order to support the requirements of the identified use-cases, ECUs may need to support hardware acceleration of some protocols/formats, while other parts may be possible to support in software. Depending on the node configuration, available memory, processing power different partitioning of the protocol stack have to considered. Therefore, it is necessary to examine and understand possible hardware and software architectures for each node of the end-to-end model.

Comment: We decided to focus on a Gateway ECU, as this was considered a highly relevant use-case for Ethernet in a vehicle. To evaluate different partitioning of the protocol stack, we used the ARP hardware (D3.1), which as an on-board FPGA to investigate the differences between performing bridging in hardware and software (on a Linux system). The results were documented in the D3.2 and D3.3 technical reports.

4.3.2 Data formats and impact on model

Different data formats and protocols that may be transmitted over the network will pose different challenges due to varying requirements on for instance, encoding, latency, fault tolerance 20

and bandwidth. Both existing and future protocols and formats need to be considered to determine their impact and requirements.

Comment: As we decided to focus on a Gateway ECU, we looked at suitable protocols for encapsulating data received on legacy automotive buses (CAN, LIN, FlexRay) on Ethernet. We made the decision to focus on CAN, as this is most widely used in cars and since the protocols and the methods used can be mostly apply to LIN and FlexRay as well. Specifically, we evaluated the IEEE1722a protocol for encapsulating CAN packets in Ethernet frames, and this is described in the D3.2 technical report. As part of this work, we also implemented support for 802.1AS (gPTP), using the on-board FPGA on the ARP board (D3.1)..

4.3.3 Network topologies and protocol configurations

A car network can be configured in a number of topologies, such as point-to-point, bus, star, ring, mesh, tree, hybrid and daisy-chain. The advantages and disadvantages of each must be explored and understood.

Comment: Due to the shifting of focus on realizing the HW & SW for a Gateway ECU, only a star topology was evaluated since it was deemed most appropriate for the use-case investigated.

4.3.4 Performance spectrum: what is achievable in latency, bandwidth, throughput, fault-tolerance

The current network technologies and protocols pose a number of hard and soft limitations on what is possible to achieve in latency, bandwidth, throughput and fault tolerance. The limitations of

existing and proposed future implementations must be identified and their impact on the model and possible use-cases understood. Possible ways of improving these areas should be also considered.

Comment: This was documented in the technical report D3.3, where we compared latency, bandwidth and throughput for two different methods of bridging CAN to Ethernet using the 1722a protocol and the ARP hardware (D3.1).

4.4 WP4: Automotive Ethernet Hardware Aspects (VCC)

The objective of this WP is to provide technical detail for realisation of requirements that have been given in WP1.

4.4.1 Analysis and Evaluate protocols for supporting QoS for streaming real-time communication over Ethernet

The focus in this task will be to analyse and evaluate protocols providing quality of service (QoS) required for streaming real time Ethernet data streaming. More specially, the focus will be on redundancy control protocols and enhanced time sensitive networking protocols. This is needed when there is a fault in the network and the logical active path is interrupted. The protocol should detect this and possibly reconfigure the active topology. The biggest challenge here is the configuration time which can vary depending on the specific use cases. The communication will be interrupted during reconfiguration and Ethernet frames may be dropped and lost in that process. The choice of redundant protocol used is largely dependent on the specific application requirements. Redundancy by means of hardware will also be investigated which is typically achieved by having redundant communication media. This is specifically important when safety critical data is transported requiring fault tolerance. Usually, this is always a trade-off between cost and capability.

Comment: the purpose of this task is to evaluate and provide technical detail for all the requirements proposed in WP1. We have focused on quality of services for time critical communication streams in the vehicle network such as control, audio and video traffic. We have developed a proof of concept for an audio scenario using time sensitive Ethernet both by simulation and HW implementation. Results of this task have been used for research and development at VCC, as well as for design of the audio use case in our next vehicle platform.

4.4.2 Measurement signal quality (Electromagnetic Compatability – EMC) of Ethernet Physical layers

In this task, the physical layer aspects of Ethernet technologies, like signal quality and EMC will be analyzed and validated. Pros and cons for the different technologies will also be studied.

Comment: A lot of practical and theoretical work has been done in this task. The Ethernet communication team and the EMC team have worked together to perform the EMC testing of the new Ethernet physical layers 1000BASE-T1,1000BASE-RH and HDBaseT. All ECUs and communication links in the car must be tested before they can be used in the vehicle, therefore the result of those EMC tests are important to be able to introduce new technologies in coming vehicle platfomrsat Volvo Cars.

A study about possibilities and challenges with gigabit Ethernet technologieshas been carried out and is reported in the deliverable D4.2.

4.5 WP5: Safety-critical Real-time Ethernet (Rise Viktoria)

4.5.1 Investigate real-time requirements

This task focuses on analysis of timing requirements such as delay, jitter, WCET (Worst Case Execution Time) from both application use cases: autonomous driving and centralized automotive E/E architecture.

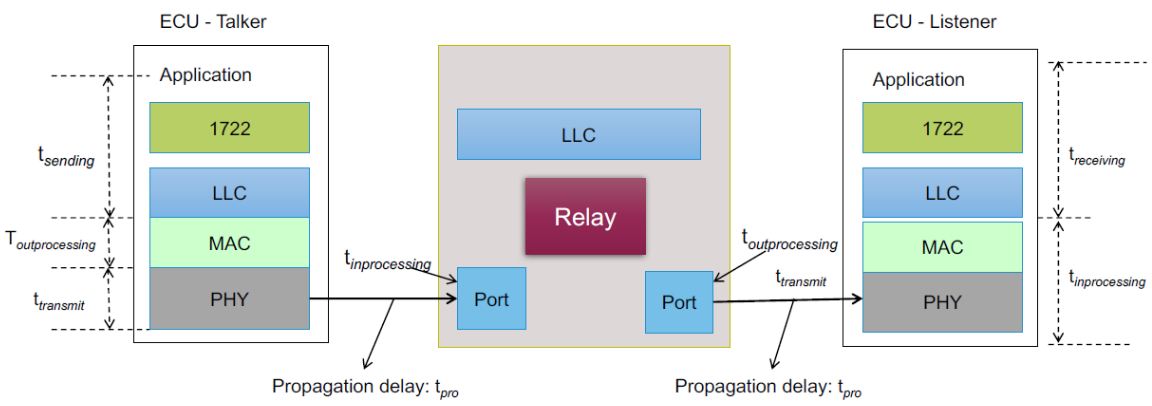
Comment: This WP has cover overall requirements on real-time capable of automotive Ethernet Architecture. Desired properties of real-time communication for safety applications are low latency, low jitter, deterministic, easy to intergration with non-real-time traffic and adaptability to changing network and traffic condition. Delay requirement of a specific scenario depends on both application software implementation and network delay. In order to guarantee end-to-end delay, it is not enough to consider the network delay, in many cases main delay comes from software processing time. When moving toward centralized architecture, more applications cross the network require different communication protocols that put requirement on network scheduling and resource allocation.

4.5.2 Develop methods and tool for timing analysis and simulation of network traffic.

This task requires both a theoretical framework and software development.

- Improve existing methods for supporting real-time communication for automotive applications.
- Develop a framework for end-to-end delay calculation of the traffic over multi-switched Ethernet network. The result of this task will also be used in WP2
- Provide methods to reduce delay and jitter to improve quality of services, especially video traffic over Ethernet networks.

Comment: Worst case delay analysis for different Ethernet traffic shapers such as Time Aware Shaper (TAS) and Credit Based Shaper (CSB) have been studied and reported in the deliverbale D5.2. A complete timing model [9] has been implemented in the network simulation (see Figure 3)



$$t_{end2end}(2\ hops) = t_{talker} + 2 * (t_{transmit} + t_{pro}) + t_{switching} + t_{listener}$$

Figure 3. Timing Model and end-o-end delay calculation for Ethernet communication in vehicle

4.5.3 Provide a real-time communication protocol over standard switched Ethernet network

The legacy Ethernet technology introduced by IEEE in the 802.3 standards were not sufficient for traffic patterns with QoS requirements in automotive applications. The industry has proposed solutions with either a new protocol layers to the regular Ethernet stack or new customized hardware designs for specific purposes. The original purpose of this task was to provide a solution for standard switched Ethernet, which can support both hard real-time traffic, e.g. signal from camera to the control part of the vehicle and soft real-time communication, e.g. multimedia video traffic over standard switched Ethernet network without modifying the standard Ethernet hardware.

Comments: when started this task, there are many work on supporting QoS and real-time streaming applications using Ethernet networks have been carried out by the IEEE Time Sensitive Networking task group [10]. Based on the application requirements provided by VCC in WP1, we have chosen to use some of the protocols in TSN standards:

1. Time Aware Shaper – 802.1Qbv [6]
2. Forward and Queing of Time Sensitive Stream, Credit Based Shaper – 802.1Qav [5]

Evaluation of these propotocols have been done at both Rise Viktoria and VCC (WP1+WP4)

4.5.4 Investigate a network monitoring mechanism for supporting robust communication over Ethernet network.

A complete network simulator has been developed to evaluate chose protocols for support real-time streaming application in vehicle network [5][6].

Assumption:

- All nodes driven by a system clock, resolution is 1µs
- All traffic are periodically generated
- Time aware shaper period is 500 µs, guardand is 26 µs, minimum slot for control traffic is 60 µs and time slot for other traffic is 324 µs.

We measured delay of each flow in the network when every nodes start to send traffic at the same time (a) and when there are offset between traffic flows (b). Detail of the simulation set up can be found in [11], simulation scenario is illustrated in the Figure 4 and Figure 5.

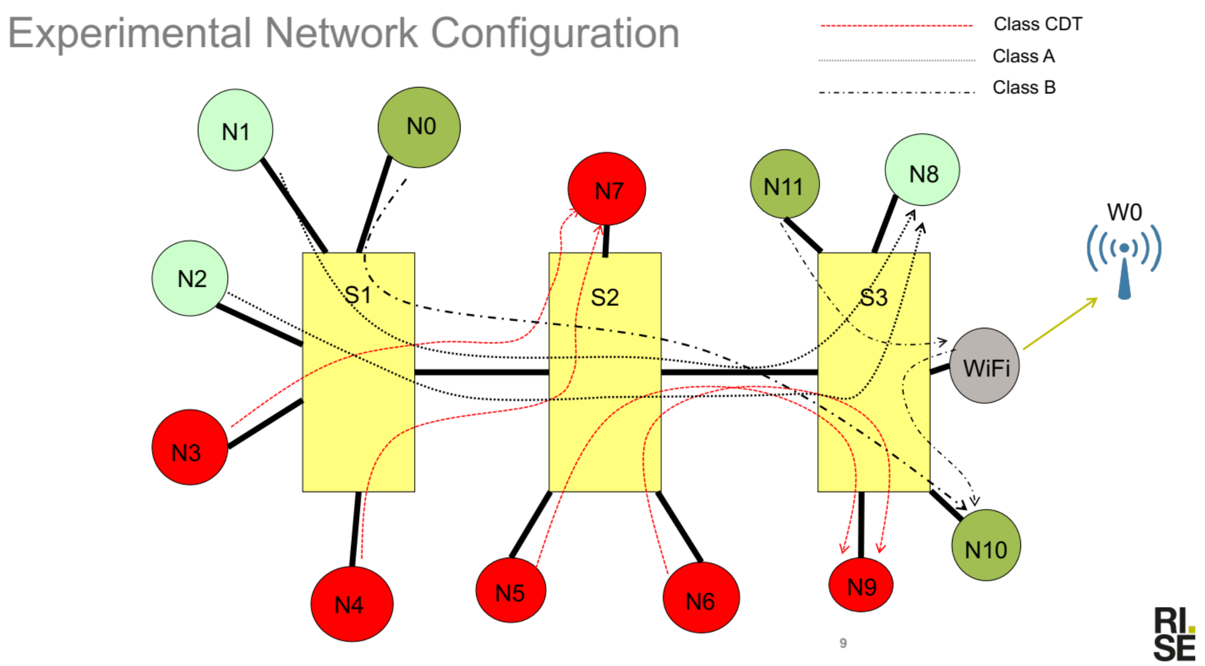


Figure 4. Network simulation scenario

Scenario Applications

Node	Node type	Stream type	I/O
N_0	Back camera	Video class B (AVB_B)	I
N_1	Voice assistant	Audio class A (AVB_A)	I
N_2	Media Audio	Audio class A (AVB_A)	I
N_3	Speed sensor	CDT	I
N_4	Engine Angular Rotation sensor	CDT	I
N_5	Wheel angle sensor	CDT	I
N_6	Steering wheel angle sensor	CDT	I
N_7	Engine speed actuator	CDT	O
N_8	Sound output device	Audio class A (AVB_A)	O
N_9	Steering wheel actuator	CDT	O
N_{10}	Video output device	Video class B (AVB_B)	O
N_{11}	Front camera	Video class B (AVB_B)	I
N_{12}	WiFi communication module	Video class B (AVB_B)	I/O

Figure 5. Scenario applications

4.5.5 Investigate a network monitoring mechanism for supporting robust communication over Ethernet network.

By using network simulation and probes, provided in WP2, we will investigate the application of statistical models for early fault prediction in vehicular network monitoring.

Comments: This task has been combined with the previous task described in the previous chapter [4.5.4]. We have extended our network simulator with a module to simulate and evaluate worst case delay of a Ethernet communication over a wireless link. In addition, Rise Viktoria has contributed to investigation of communication protocol for supporting redandance in in-vehicle network. Result is presented in the section 5.2.5.3.

4.6 WP6: Demonstrator AcrCore

The concepts developed in other work packages may be of little value unless they can be evaluated and verified in an ECU context. AUTOSAR is becoming the de facto standard at most automotive OEMs when it comes to real-time applications while Linux is often preferred for infotainment applications. The demonstrator will include both to provide an accurate setup that resembles a simplified vehicle. Therefore, it is necessary to examine and understand possible hardware and software architectures for each node of the end-to-end model. One of the goals in the project was to develop a performance analyzer for evaluation efficiency of implementations. This analyzer will enable to measure CPU load and also the measure time between executions points in the software.

In this workpackage ARCCORE provided an AUTOSAR embedded platform, ARCTIC Core, that has been used to set up the ECU measurements in the demonstrator. The Arctic Core is an embedded platform developed by ARCCORE, which includes a full set of features required in an Automotive Electronic Control Unit including communication services, diagnostic services, safety services and a real-time operating system. It is based on the industry standard AUTOSAR and brings the benefit from the goals of the AUTOSAR which enables true re-use of software investments, benefiting from a focused time-to-market approach and leveraging on multiple COTS products available. All provided in an open approach providing a full control of both the development and the production phases of the project in a cost-efficient way.

4.6.1 Performance analyzer

To make it possible to realize the demonstrator ARCCORE has also developed a performance monitoring module for the ARCTIC Core platform. The module implementation has been performed in ARCCORE development teams and is fully integrated into Arctic Core. The PerfMon module is an extended BSW(Basic Software Modules) in the Arctic Core which compliant with AUTOSAR standard. The PerfMon module measures:

- The maximum, average execution time and the CPU loads of specified function and the nested interrupts.
- The execution time between two points within the SWCs (Software Component).

4.6.2 Demonstrator

During the project we investigated the possible setups for a demonstrator and decided to use a simulation of ECU and network modelling. To be able to set up a correct model of the signaling in an ECU ARCCORE has enabled mesaurements in ECUs using the performance monitoring module that has been developed in the project. This work has been done together with WP2 by TCN.

5. Results and deliverables

5.1 Project contribution to overaching FFI objectives

5.1.1 Increasing the Swedish capacity and innovation, thereby insuaring competiveness and jobs in the field of vehicle industry

- Automotive Ethernet is one of the attractive area for many engineering. Making Ethernet becomes main technology in the vehicle is a bis step that VCC has achived after the ECRA project. Partners in the project have employed more people to work in the area of automotive Ethernet technology.

5.1.2 Developing internationally interconnected and competive research and innovation environment in Sweden

- The project has enabled the project partners to work in a cross-disciplinary research and development environment, which is internationally interconnected through the partners' connections.
- Volvo Cars and TCN are members of the OPEN Alliance and have participated in workshops arranged by OPEN. Volvo Cars is also a steering committee member and have been active in the steering committee during the project. The OPEN Alliance (One-Pair Ether-Net) is a non-profit, open industry alliance of mainly automotive industry and technology providers working to encourage wide scale adoption of Ethernet-based networks as the standard in automotive networking applications.
- International collaboration has been increased during the project, i.e., TCN has estabilished business partner with Spirent and Technica, two main Ethernet test tool providers for automotive Industry
- VCC successfully co-organised Automotive Ethernet Congress (AEC) 2019, which is one of the biggest events in automotive networking industry. The congress attracts people from high level managers, technical experts, product managers and engineers from OEMs, SMEs, suppliers and academy.

5.1.3 Promoting the participation of small and medium-sized companies, promoting the participation of subcontractors, promoting cross-industrial cooperation

- Project has three partners are SMEs. All of them have competence development in the project transfer into their product development. Some collaboration between partners has been created and potentially extend further after the project duration.

5.1.4 Promoting cooperation between industry, universities and higher education institutions

- Volvo Cars and Rise Viktoria, academic partner in the project, have been working together in several WPs. As the results, three publications have been published (details found in the Section 6.3).
- A staffs have moved from Rise Viktoria to VCC to work with advanced engineering project in Time Sensitive Ethernet for in-vehicle communication.

5.2 Examples of the project results

5.2.1 WP1: Requirements and Use Cases

5.2.1.1 Base Technology Specifications

We have updated and created all the requirements for physical layer and data link layer of Ethernet components in the vehicle platform including:

- Requirements for 100BASE-T1 Physical layer (updated)
- Requirements for 1000BASE-T1 Physical layer (created)
- Requirements for Ethernet Datalink layer (updated)
- Requirements for Time Sensitive Ethernet (created)
- Requirements for 1000BASE-RH Physical layer (created)

5.2.1.2 TSN proof of concept for VCC Audio use case

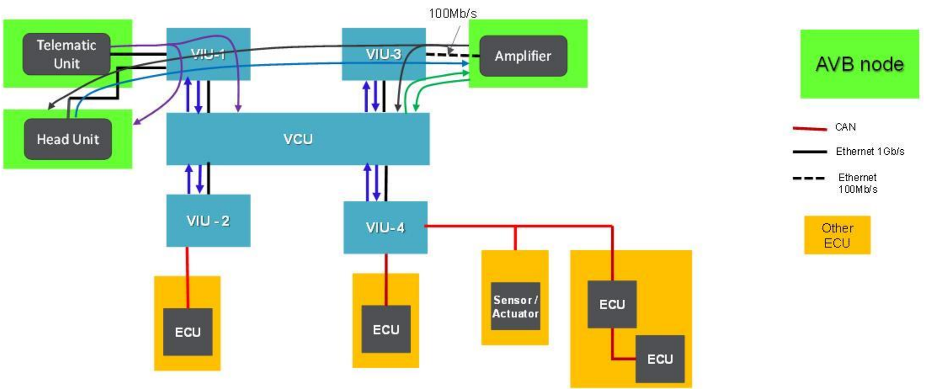


Figure 6. VCC Audio Scenario

Figure 1. illustrates Volvo infotainment scenario. Audio signals are sent from the Head Unit to Amplifier, feedback signal is sent from the Amplifier to the Head Unit. Video streams are broadcasted from Telematic Unit to the Head Unit and the VCU (vehicle computational unit). We have done a proof of concept to find the solution of using Ethernet TSN for audio/ video signals also for gateway signal from CAN ECUs to the Ethernet ECUs. The proof of concept has been done both by mean of network simulation (developed in WP5, WP4) and also HW implementation (WP4). By using stream reservation protocol (SRP) together with credit based shaper (CBS), we guarantee that all the time sensitive signal can obtain the maximum latency of 180µs, no packet loss in the network and very low jitter. All stream paramettters are presented in the Figure 7, and simulation results are showed in Figure 8 below.

Stream	Listener	Talker	priority	Payload (bytes)	Header (bytes)	Interval (µs)	SR class
1	Head Unit	Amplifier	6	600	74	256	B
2	Amplifier	Head Unit	6	360	74	256	B
3	Amplifier	VCU	6	360	74	256	B
4	VCU	Amlifier	5	312	74	1333	C
5	Amplifier	VCU	5	432	74	1333	C
6	VIU-1	VCU	7	1171	66	125	A
7	VCU	VIU-1	7	1171	66	125	A
8	VIU-2	VCU	7	1171	66	125	A
9	VCU	VIU-2	7	1171	66	125	A
10	VIU-3	VCU	7	1171	66	125	A
11	VCU	VIU-3	7	1171	66	125	A
12	VIU-4	VCU	7	1171	66	125	A
13	VCU	VIU-4	7	1171	66	125	A
14	Telematic Unit	Head Unit	4	1480	42	740	BE
15	Telematic Unit	VCU	4	1480	42	740	BE

Figure 7. VCC Audio scenario – streams parameters

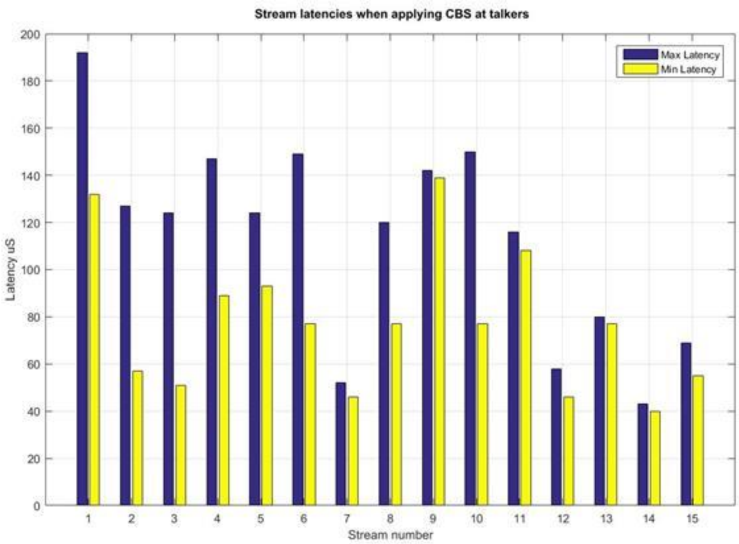


Figure 8. VCC Audio Scenario – Simulation Results

5.2.2 WP2. Tool Development and Assessment

5.2.2.1 TCN Simulator

In ECRA task T2.3, the new modeling functionality that has been implemented in the simulation engine of TCN TimeAnalysis™ was applied when trying to construct a virtual model of a simple, physical CAN-to-Ethernet gateway realized by ARCCORE as part of ECRA WP3. Below we present how the new task modeling mechanisms were used to construct the model of the physical gateway as well as a simple architecture including it.

When modeling the CAN-to-Ethernet gateway using the new task modeling functionality added to the simulation engine of TCN TimeAnalysis™, it was easiest to make the virtual counterpart of ARCCORE's gateway part of a simple, virtual architecture. In this architecture, Host1 is the source of data frames that are to be forwarded to a target ECU, Host3. Host2, positioned between Host1 and Host3, must re-package the data found in incoming CAN frames into Ethernet packets and Host2 thus corresponds to ARCCORE's CAN-to-Ethernet gateway.

ARCORE tried two different task configurations in the gateway. As a result example, we explain below how the tasks were modeled in one of these cases:

5.2.2.2 First gateway configuration

Specific for this task configuration in the CAN-to-Ethernet gateway, or Host2, was that the basic software (BSW) implementing the CAN-stack and the application receiving the data values carried in CAN frames were run in the same task. From a modeling perspective, it seemed appropriate to then use the concept of chained tasks in the OSEK operating system. This implied that the tasks would be running back to back during a simulation.

The reader is now referred to Figure 9, in which a Gantt-chart is used to visualize how the very first job instances of different hardware- and software tasks were run during a short simulation. In the Gantt-chart, the left column lists the processing units or hardware resources responsible for executing the different tasks and the x-axis scale is in microseconds. To make it easy for the reader, all execution times of tasks run by CPUs have been set to a constant 100 µs.

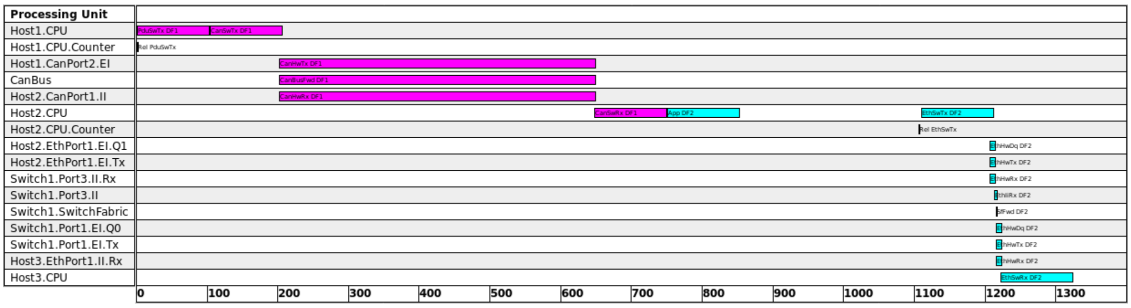


Figure 9. First Gateway Configuration

Now, with reference to Figure 9, we guide the reader through the different tasks in the Gantt-chart and explain which constructs were used to implement the task model:

The first task in the upper left corner of Figure 9 is PduSwTx. It collects some signal or data values into a data frame, DF1, and then instructs the CAN stack to send DF1 to the target ECU, Host3, via the gateway Host2.

Task PduSwTx is configured to be a periodic task. As explained in [1], while a simulation is launched, such a task is associated with an alarm and a counter. The counter will periodically trigger an interrupt, which then causes the scheduler to release task PduSwTx from the waiting state. As there are no other periodic tasks running in Host1, the task will directly be transferred to the running state, i.e. mapped to the Host1 CPU and executed. The occurrence of the interrupt signal, that triggers the release of PduSwTx, is visualised in the row below and occurs at time 0.

Chained to task PduSwTx is task CanSwTx that corresponds to the CAN-stack placing DF1 into a CAN frame and instructing the CAN egress interface to transmit the frame onto the CAN bus.

The following three boxes correspond to the egress interface (EI) of the CAN port of Host1 transmitting the CAN frame, the CAN frame carrying DF1 occupying the CAN bus for a period of time and the CAN ingress interface (II) receiving the same CAN frame, respectively.

Due to the short length of CAN-busses in vehicle architectures, these three boxes overlap in time. However, in an optical fibre having a length of several miles transmitting an Ethernet frame, the progress of the electromagnetic wave carrying the information from one end of the cable to the other end will be non-negligible on a microsecond scale, so in that case the Tx- and Rx-boxes would actually be separated in time.

When the CAN-frame has been fully received by the CAN ingress interface in the gateway, Host2, a hardware interrupt triggers the BSW CAN stack to run as an interrupt service routine (ISR) and process the CAN frame. Thereby, the data values carried by data frame DF1 will be extracted and placed in memory. Next a user application, called App in Figure 9, is to place the new data in an Ethernet frame and send it out on an Ethernet network. In a real scenario, this would likely be an Ethernet backbone connecting to a number of sub networks via gateways. Then, the application App would be collecting a number of data frames arriving via different CAN ingress interfaces. When either a maximum number of data frames have been collected, or a maximum relative time has passed, an Ethernet frame packing all the collected data frames would be sent. In this simple example, however, we send a single data frame, DF2, per Ethernet frame.

According to ARCCORE, the processing of the a new Ethernet frame by the Ethernet stack had to be placed in a separate task. Therefore, task EthSwTx in Figure 9 was configured to run independently as a periodic task and thus asynchronously in relation to tasks CanSwRx and App in Host2. This explains the gap between the termination of task App and the commencing of EthSwTx, the size of the gap being the result of a randomly chosen offset.

The Ethernet frame then traverses a small Ethernet network comprised of a switch and two Ethernet links, being temporarily stored in queued buffers in the Ethernet egress interfaces of Host2 and the switch, respectively. However, as there is no other traffic contending for shared resources in this simple example, the Ethernet frame experiences no queuing delay.

Finally, the Ethernet frames arrives att Host3, were the IP-stack processes the Ethernet frame after having been triggered by a hardware interrupt.

5.2.3 WP3. Efficient Implementation

5.2.3.1 Luxoft/Pelagicore Automotive Reference Platform

The automotive hardware platform was documented in the “ARP 1.1 User’s Guide” as deliverable D3.1, while two technical reports served as deliverables D3.2 and D3.3 respectively.

ARP – Automotive Reference Platform designed for prototyping advanced digital cockpit platforms and capable of driving vehicle instrument cluster, head unit display, cockpit occupant monitoring and driver assistance system simultaneously. The ARP is designed from the ground up to be reconfigurable and expandable [8].

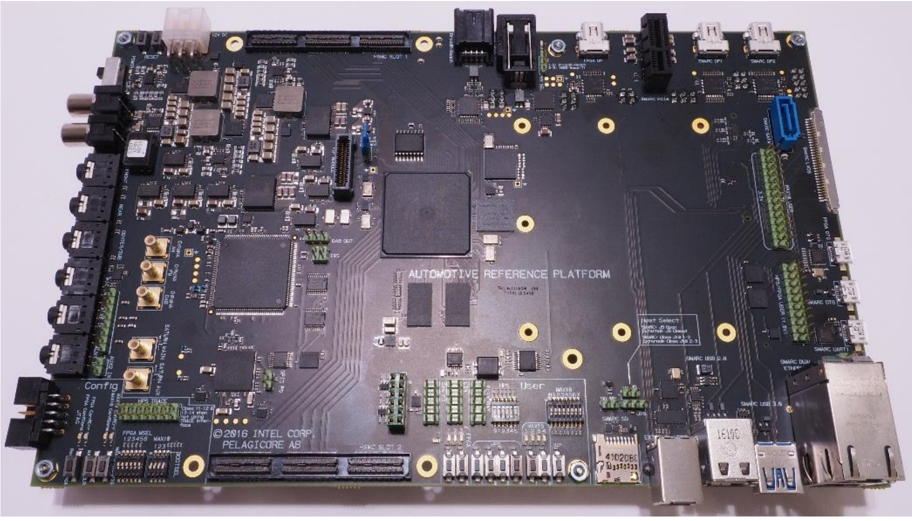


Figure 10. Automotive Reference Platform (ARP)

The strength of the platform is its highly modular system design. Not only is the SoC-module interchangeable – with various Arm and x86 modules available on the market – but by building the board around an Altera Cyclone V FPGA with integrated Arm cores the platform enables high levels of design flexibility.

In addition to USB, Ethernet, SD, SATA, HDMI, DisplayPort, and automotive-specific interfaces like CAN, LIN, FlexRay, Ethernet AVB, the ARP leverages the reprogrammability of the FPGA to add two HSMC slots with massive high-speed I/O expansion capabilities. A wide variety of HSMC expansion boards are available through a partner ecosystem, which allows the ARP to support most existing and emerging connectivity standards. For instance, the platform's four display outputs can be increased to nine, while multiple video cameras and other sensors can be added to enable advanced use-cases such as driver monitoring and assistance.

5.2.3.2 ARP-based Ethernet Gateway Solution and method to map CAN frame into Ethernet packets using IEEE1722a protocol

There are two methods of converting packets received on a Controller Area Network (CAN) bus to Ethernet packets and sending them out on wired network (reffered to as bridging) have been presented in the deliverable D3.2 and D3.2. The two solutions covered are hardware bridging and software bridging.

That HW solution is an implementation done in hardware (Figure 7), using an FPGA (Field Programmable Gate Array) and an implementation done in software, running on a Linux operating system. The hardware implementation is expected to be faster, but since software development is much cheaper than hardware development the interesting question is, will a software implementation be fast enough.

The software solution runs on ARP with an Intel Apollo Lake-based SMARC 2.0 module, specifically an Intel Atom x7-E3950. This SoC has a base frequency of 1.6GHz and is capable of boosting clocks to 2.0GHz, depending on power and thermal conditions. The SoC-module also has 8GB of LPDDR4 with 2400MT/s (Figure 8). The operating system used is Ubuntu 17.10 as operating system, specifically a Linux system using the 4.13 Kernel.

5.2.3.3 Conclusions & Future Research

A hardware solution for bridging legacy automotive buses to Ethernet is without any doubt superior in performance, as we would expect. In the case of CAN, which was examined, the majority of the time is spent on reception of the CAN frame, so the data rate of the CAN bus is the limiting performance factor.

In the software bridging solution, the latency increases by a factor of 10 while sending less than 4 bytes of data in each CAN frame. We found that at this point the time between the timestamps of when Ethernet frames were sent increases while the time difference between when CAN frames are received stay the same. This led us to believe that the issue was with our implementation of the gateway service. However, we did not have time to investigate this.

Further measurements of a CAN-FD implementation should be tested for increased analysis of the highest possible data rate of 8 Mbit/s. This would conclude whether or not the software implementation is limiting the throughput of the software solution. It is expected that it will perform adequately but a round of testing should be performed.

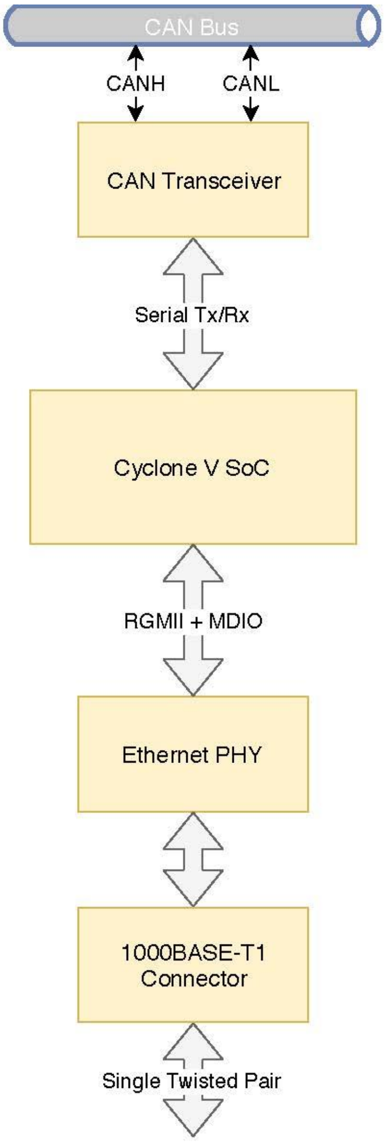


Figure 11. Hardware based Solution of CAN-Ethernet gateway using IEEE 1722a protocol

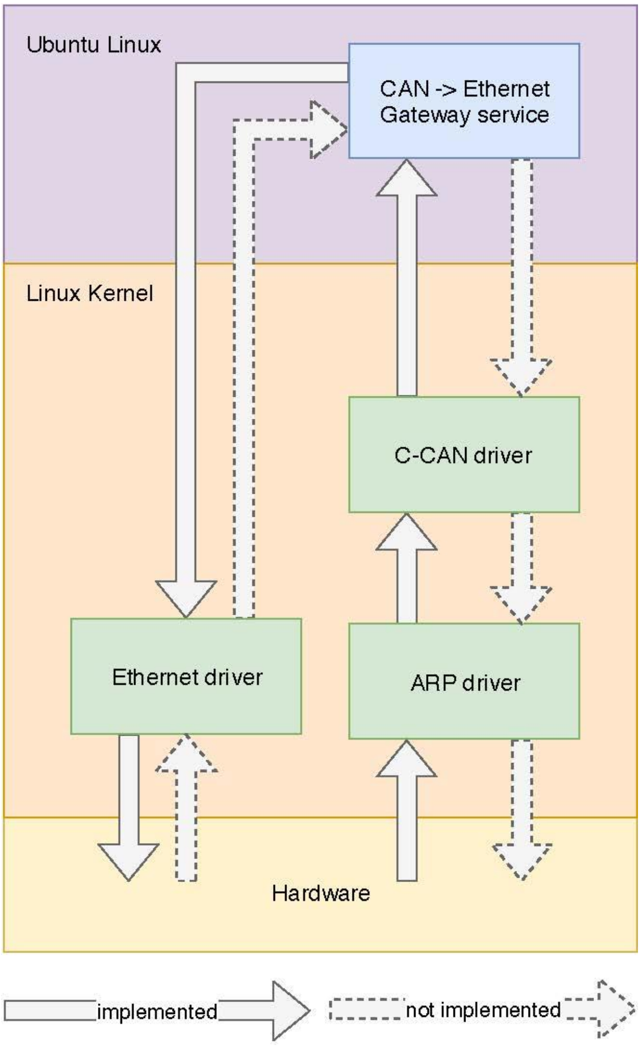


Figure 12. Software components of software-based bridging solution

5.2.4 WP4: HW Aspect of Automotive Ethernet

5.2.4.1 Technical report about protocols for fault handling and redundancy

This work has been done by VCC and Rise Viktoria (in WP5). We have investigated the seamless redundancy protocol defined in the IEEE 802.1cb standard "Frame replication and elimination for reliability". In order to avoid frame loss due to equipment failure, packets are sent on two disjoint paths then combine and delete extras (see Figure 20).

5.2.4.2 Technical report describing possibility and challenges with GEPOF, 1000Base-T1 Ethernet physical layer

GEPOF is Gigabit Ethernet over Plastic Optical Fibre. It is a technology that has been standardized in IEEE in the project 802.3bv. The name of the technology is now 1000BASE-RH. It is a technology that makes it possible to use a plastic optical fibre and photo diodes for 1000Mbit/s Ethernet transmission. The main target for the technology is Automotive. Main advantages with GEPOF is that the optical fibre is immune to electromagnetic interference, the link galvanicly isolates the connected ECUs, and the fibre is light. During this project the specification has been finalized and the specification describing the technology can be found in [11]. The technical report, result of this task, provides a study in cost, power consumption of different Ethernet physical layer technologies. This can be used to evaluate and making decision when it comes to which is best suitable for automotive.

5.2.4.3 A complete network simulation of Time Sensitive Networking applying for VCC E/E architecture

In WP5, a network simutor including TAS and CBS have been developed at Rise Viktoria. In this task, based on the input from WP1 by VCC, we have extended the simulator with:

- a complete stream reservation protocol
- network generator module, which can generate both streaming traffic i.e. audio/video, periodic control traffic (CAN traffic) and internet (TCU/UDP) traffic
- a getway module, which allow to simulate gateway between CAN traffic to Ethernet traffic.
- A new centralized E/E architecture, which is provided by VCC

Detail of developed network simulator has been shown in the Figure 13. Network topology and traffic generator were implemented in Matlab. All the scheduler, switching configuration and main simulation program were done in Java programming language.

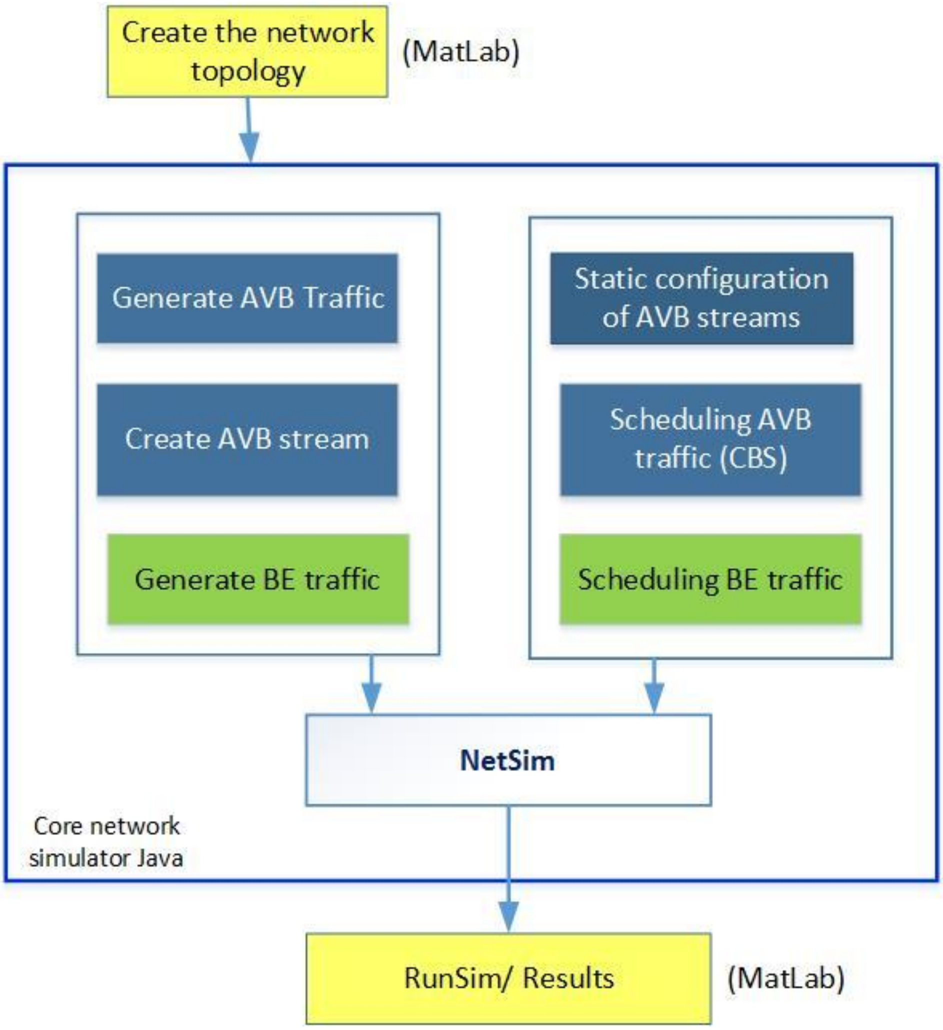


Figure 13. Time Sensitive Networking Network simulator

Figure 14 shows VCC centralized electrical electronic architecture that have been presented as one of the use case for the WP1 and also simulation scenario for WP4.

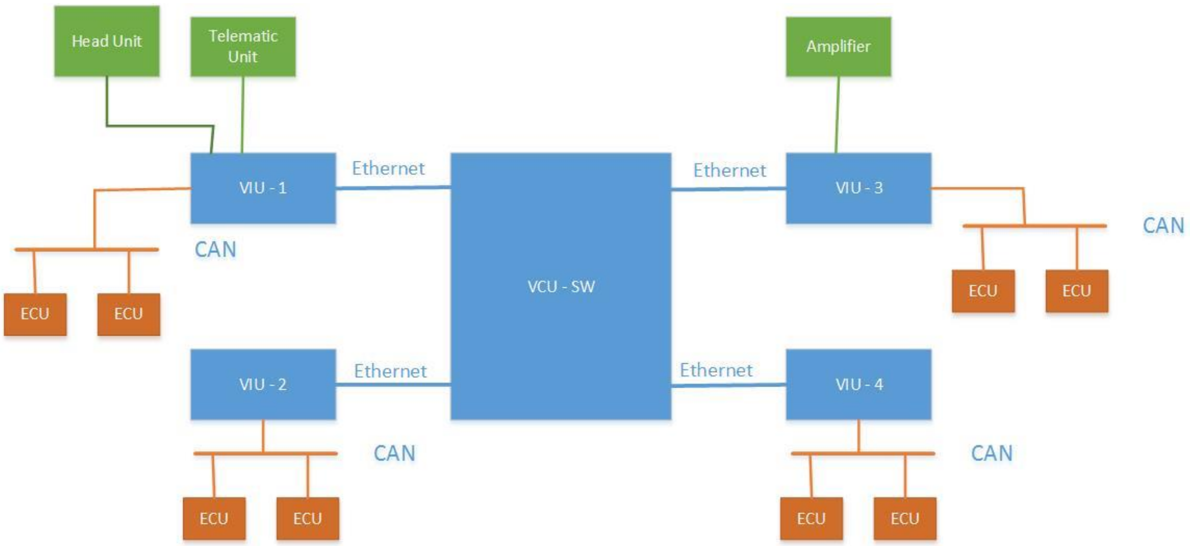
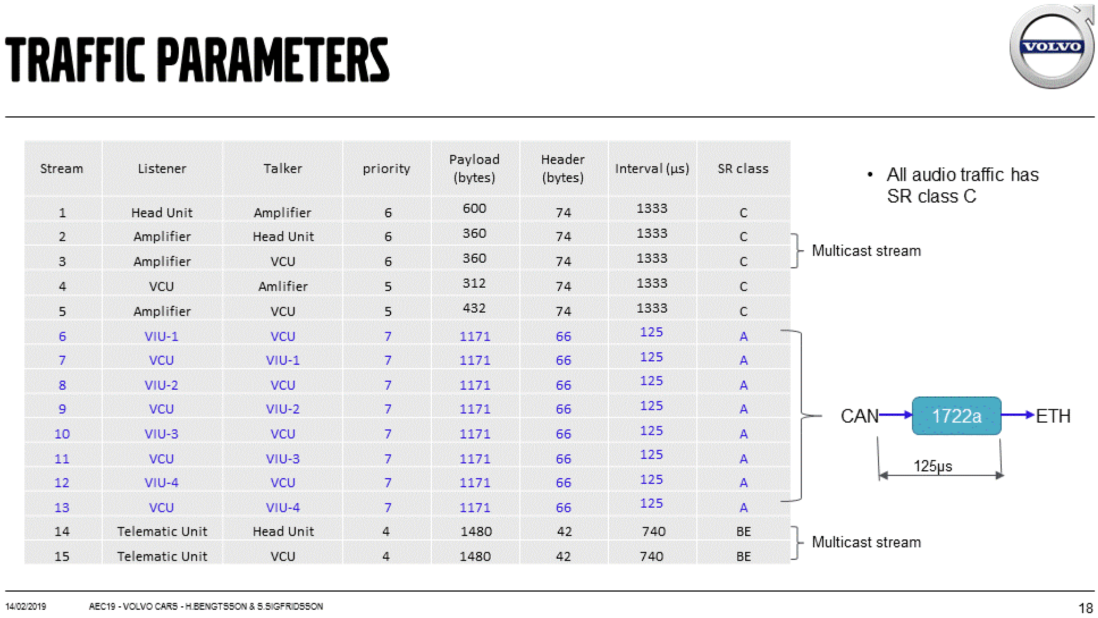


Figure 14. Centralized E/E architecture in automotive

There are two main parts in this architecture:

- Core network consists of VCU (vehicle computation unit) and VIU (vehicle integration unit).
- Mechanical rim contains all ECUs in the vehicle

Mechanical rim connect with VCU via the VIU. VIU also gatewaying all CAN/LIN messages to Ethernet messages to transmit via the core network. There are different way for doing gateway protocol. In this task, we evaluated two different approaches: packing all CAN messages to UDP messages (1) and using 1722 protocol, which is the transport protocol for audio, video traffic in Ethernet AVB, for CAN-Ethernet gateway. The results shows that when the network load is less than 75%, two protocols perform similar (see Figure 16). Network scenario, which has been implemented presented in Figure 15.



Stream	Listener	Talker	priority	Payload (bytes)	Header (bytes)	Interval (μs)	SR class
1	Head Unit	Amplifier	6	600	74	1333	C
2	Amplifier	Head Unit	6	360	74	1333	C
3	Amplifier	VCU	6	360	74	1333	C
4	VCU	Amlifier	5	312	74	1333	C
5	Amplifier	VCU	5	432	74	1333	C
6	VIU-1	VCU	7	1171	66	125	A
7	VCU	VIU-1	7	1171	66	125	A
8	VIU-2	VCU	7	1171	66	125	A
9	VCU	VIU-2	7	1171	66	125	A
10	VIU-3	VCU	7	1171	66	125	A
11	VCU	VIU-3	7	1171	66	125	A
12	VIU-4	VCU	7	1171	66	125	A
13	VCU	VIU-4	7	1171	66	125	A
14	Telematic Unit	Head Unit	4	1480	42	740	BE
15	Telematic Unit	VCU	4	1480	42	740	BE

Figure 15. Network scenario for gateway protocols comparison

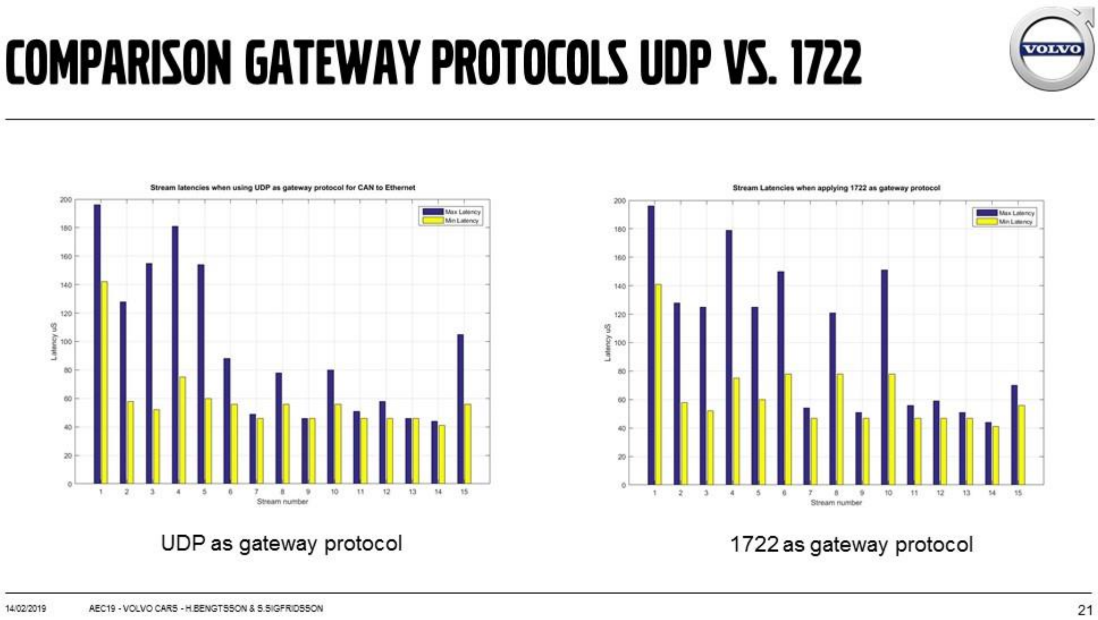


Figure 16. Perfomance comparison UDP vs. 1722

5.2.4.4 Hardware experiment of using AVB for VCC audio scenario

We have done a set of experiments with different automotive hardware. The goal of these experiment is to understand how the chosen protocols in WP1 work with real VCC scenario. The set up has done as in Figure 17. We implement audio application as AVB streams to send over Ethernet network. Two NeoECUs act as AVB talker and listener; the switch Ethernet has all AVB protocol stack implemented.

Figure 17. TSN proof of concept with VCC audio use case

We use Vehicle Spy (provided by Intrepid) to configure all the TSN streams. End-to-end delay of each traffic flows have been measured.

5.2.5 WP5. Safety Critical Real-time Ethernet

5.2.5.1 Requirements on Real-time Capable Automotive Ethernet Architectures – Technical Report

In this technical report, we have made a survey about requirements on real-time capability of Ethernet for Automotive safety-critical applications. The main contributions of this document are:

1. Analysis how Ethernet architecture effect real-time on real-time capability of Ethernet
2. Real-time communication approaches for automotive domain

5.2.5.2 Evaluation of Ethernet AVB/TSN by network simulator; End-to-end latency measurement

Ethernet-based Time Sensitive Networking (TSN) is a key enabling technology for carrying high bandwidth sensor data, such as video for ADAS in the vehicle. The main focus of new IEEE Time Sensitive Network (IEEE 802.1 TSN) is providing deterministic, low latency for Control Data Traffic (CDT). Together with the previous IEEE 802.1 AVB, TSN can also give a good support to in-vehicle video streams between related ECU. However, in order to apply new TSN standard to the next vehicle E/E architecture, we need tools, methods to investigate, test the feasibility of the new proposed schedulers such as Time Aware Shaper and Credit Based Shaper when applying to specific application scenarios.

There are different ways to evaluate end-to-end delay of a message stream over the network. In this deliverable, we use simulation method to measure delay and jitter of each flow under certain network conditions such as link's bandwidth and network load condition.

- **Simulation – software architecture**

We developed a simulator using Java programming language for modeling in vehicle network (IVN) according to IEEE 802.1 TSN standard. The traffic scenario is described in Matlab. The software architect of Ethernet in-vehicle network and simulation components as illustrated in Figure [19].

- *End-node (ECU)*. End-node in the system could be either sender or receiver. It is implemented as traffic generator/ recorder. The layered architecture of the node is described in Figure 18. We simulate communication in layer 2 (MAC address table and Link Logical Control-LLC operation) of both end-nodes and switches. We assum that 802.1AS module for synchronization is included in the LLC sub-layer.
- *Switch*. The switch model consists of number of input ports (ingress ports), output ports (egress ports) and a relay module. Each port has a transmit module and a receive module (full-duplex). The switch has a LLC layer with 802.1AS synchronization support. Furthermore, to deliver the incoming frames to the corresponding output ports in order to route them to the correct destination, the Ethernet switch includes a relay module. Figure 18, a simple example of layered architecture of a switch with one Ingress port and one Egress port is shown.

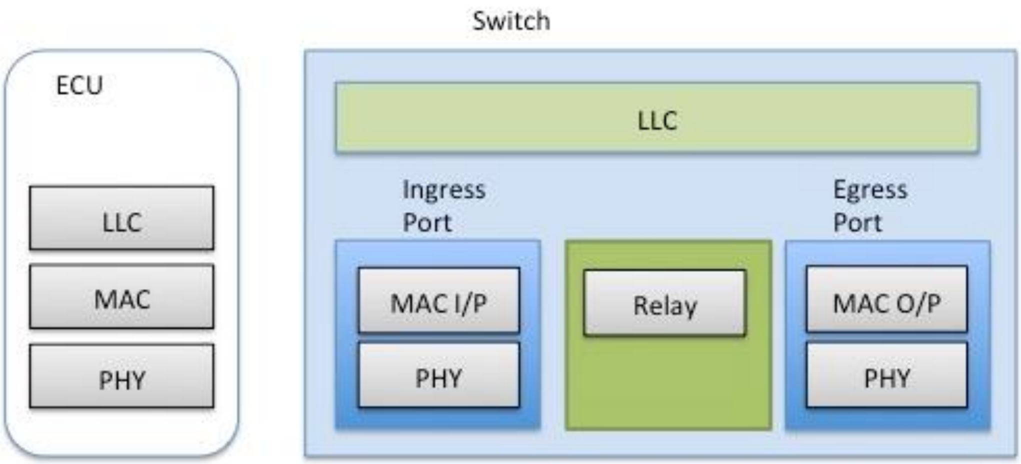


Figure 18. Software Architect in the end node (ECU) and in the Switch

- **Simulator Configuration**

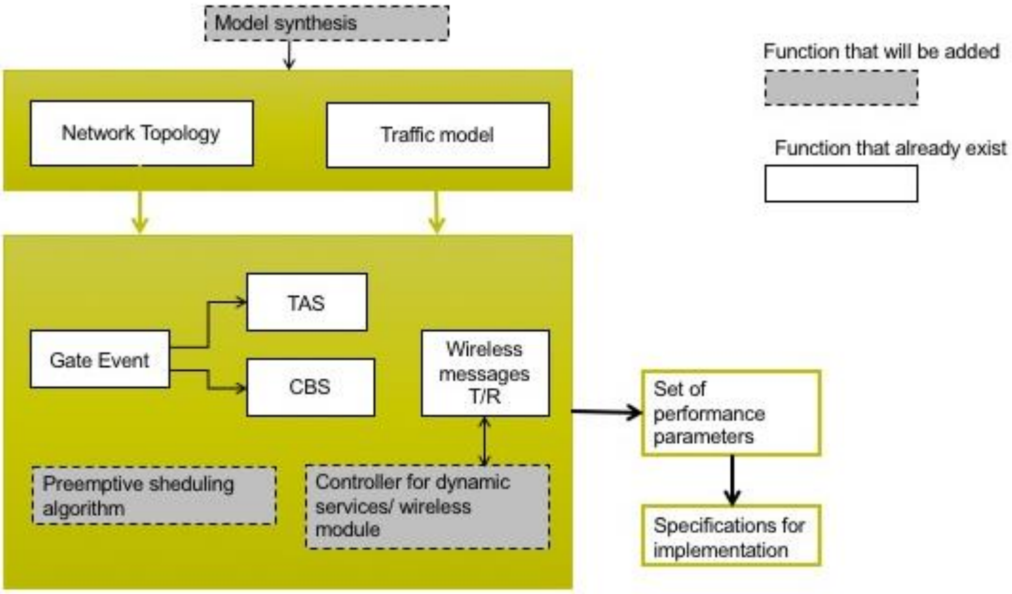


Figure 19. Simulation architecture

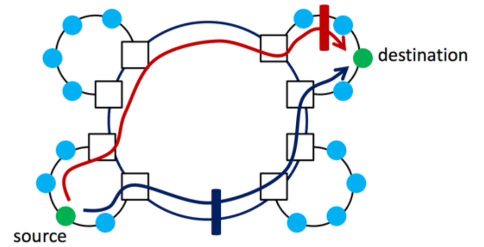
Simulation executable software and results are included in the Deliverable D5.3 (Rise Viktoria)

5.2.5.3 Study of Frame Replication and Elimination for Reliability in the IEEE 802.1cb standard

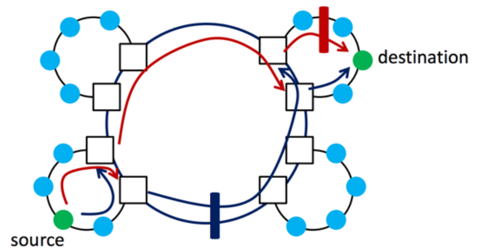
High-assurance applications require reliable delivery of information over the network. To increase the probability of delivering a given packet, IEEE have developed a standard 802.1CB "Frame Replication and Elimination for Reliability" (FRER). With FRER, packets are delivered over multiple paths, which substantially reduces the probability of packet loss due to equipment failures. FRER does not specify how those paths are created, and also relies that those paths are protected against congestion losses. This work package studied in detail and evaluated the standard, and compared it to the previously available solutions.

FRER provides *seamless redundancy*, meaning that there is no reconfiguration time required in case of hardware failure – redundant paths and links are active all the time, and if one path disappears, information is still delivered. In a nutshell, FRER provides each packet in a stream with a sequence number. Then each packet is replicated, and sent over two (or more) fixed paths. Receivers and intermediate nodes keep track of what packets have and have not been seen, and eliminate the duplicates.

FRER supports multiple replication, which provides better resilience to multiple failures, as illustrated in Figure 20 below.



(a) No multiple replication. Two link failures (black and red bars) lead to packet loss.



(b) Multiple replication. The packet is delivered despite two link failures (black and red bars).

Figure 20. Bulk stream with multiple packets in flight.

Implementation complexity of FRER depends on the properties of the network and data streams. For so-called *intermittent streams*, when there is at most one "packet in flight", implementation requires one one additional memeory element per stream. It then can use MatchRecoveryAlgorithm to find out if the arriving packet is a duplicate of a previously seen packet. For streams that can have multiple packets in flight, so-called *bulk streams*, which happen at high sending frequency and long paths, see Figure 21, implementation is much more complicated. Implementor must keep a bit-vector of last-seen sequence numbers and use VectorRecoveryAlgorithm to detect if a packet is a duplicate of another packet.

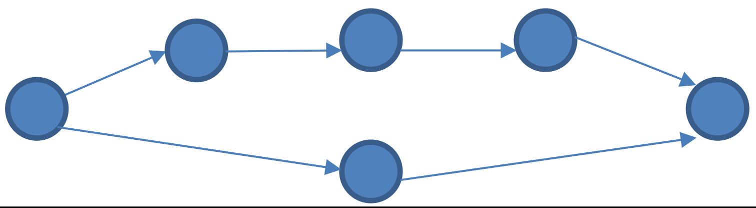


Figure 21. Multiple replication

So how the FRER packets with so-called R-TAG (Redundancy Tag) are different from the ordinary Ethernet II packets. Figure 22 Illustrates the difference.

Ethernet II

MAC destination	MAC source	802.1Q tag (optional)	Ethertype	Payload	Frame check sequence (32-bit CRC)
6 octets	6 octets	4 octets	2 octets	46-1500 octets	4 octets

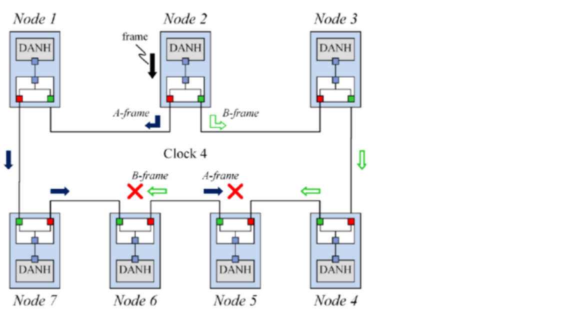
R-TAG (Redundancy Tag)

MAC destination	MAC source	802.1Q tag	Ethertype	Reserved	Sequence Number	Payload Length/Ethertype	Payload	Frame check sequence (32-bit CRC)
6 octets	6 octets	4 octets	2 octets "F1C1"	2 octets	2 octets	2 octets	40-1494 octets	4 octets

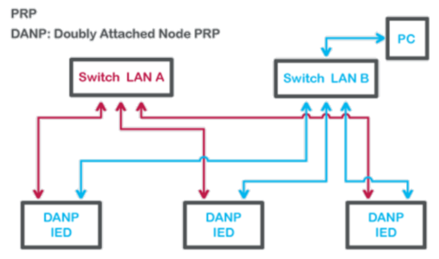
Figure 22. Ethernet packet structure for R-TAG

Since FRER provides seamless redundancy, errors can remain unnoticed, since packets are delivered even when there are errors. To warn about errors, FRER defines mechanisms for *latent error detection*. For a stream with *n* path, there will be *n-1* discarded packets for each packet passed through. If too few packets are discarded within a time window -- signal an error. To prevent occasional random packet losses from accumulating forever, the counters are reset periodically.

Comparison to previous technologies. High-availability Seamless Redundancy (HSR) and Parallel Redundancy Protocol (PRP) are two technologies that provide seamless redundancy, see Figure 23 for illustration of operation principle. Neither HSR nor PRP support multiple replication. FRER is a much more flexible solution for achiving seamless redundancy.



a) High-availability Seamless Redundancy (HSR)



b) Parallel Redundancy Protocol (PRP)

Figure 4. HSR and PRP.

Figure 23. HSR and PRP

5.2.6 WP6: AcrCore demonstrator

5.2.6.1 Performance monitoring module

To make it possible to realize the demonstrator ARCCORE has also developed a performance monitoring module for the ARCTIC Core platform. The module is implemented in C is fully integrated into Arctic Core. The PerfMon module is an extended BSW(Basic Software Modules) in the Arctic Core which compliant with AUTOSAR standard. The PerfMon module measures:

- The maximum, average execution time and the CPU loads of specified function and the nested interrupts.
- The execution time between two points within the SWCs.
- There is also as script impleted in Python available. The script can receive measurement data in runtime and generate performance graphs.

For a more detailed description of the performace monitoring module please see the technical report.

5.2.6.2 ECU measurements

For the performance measurement part, we had an experimental setup with multiple measurements performed. The setup consisted of a MPC5748G ECU running Arctic Core 20 with performance-measuring code enabled, and a PC that was connected to the ECU both through Ethernet and CAN. For Ethernet, a regular connection was used, and for CAN the PC had an attached Kvaser Leaf v2 to act as the actual CAN node.

Performance was measured using the newly implemented PerfMon module, which relies on a high-precision hardware timer. In this particular setup, the timer had 1 microsecond (us) resolution.

The high-level application performed a simple function, to read the data received in a CAN frame and to forward that data to an output Ethernet frame.

The task scheduling was as follows:

A so-called BSW task scheduled every 5 ms. The BSW task performs most of the Autosar stack's functionality, such as sending data that was requested to be sent, managing state machines, etc. The TCP/IP stack runs in a separate task also scheduled for every 5 ms. In some test runs, the application (the data forwarding code) ran in the BSW task, and in other runs it was assigned to a separate task that was then scheduled to run every 5 ms, with an offset of 1 ms after the BSW task.

Two additional parameters were used to differentiate test runs. The "High CAN load" parameter, when enabled, simulated a high network load by sending 10 to 15 CAN frames with a high frequency, between 5 and 15 ms per frame. With the parameter off, CAN load was "normal", with 3 incoming frames. The "High application load" parameter controlled whether the application simulated having additional workload, implemented by random loops of calculations and random transmissions over SPI.

For each of the six possible combination of parameters (presence of a separate application task, high CAN load, high application load), 10 test runs were performed, giving a total of 60 test runs. The measurements were performed between the following measurement points:

CanHwRx - receive interrupt in the CAN driver. This measurement point triggers almost instantly after reception of a message.

AppReceive - the point at which the application code has received the data contained in a CAN frame.

Lwip - measurement at the beginning of the first function in the Lwip TCP/IP stack, or the end of the last function in the stack, depending on what is being measured

EthHwTx - beginning of the trasmit interrupt in the Ethernet driver.

The time of the Ethernet transmit interrupt was also measured.

From the results of the test runs, the main conclusion is that scheduling was the most important factor for the timing, and that the time between tasks is an order of magnitude larger than the other timings. The delay from CanHwRx (physical reception of the CAN message) to AppReceive (reception of the same message by user code) was dominated by the time between the physical reception and the next scheduled run of the user code. Running the user code in a separate task decreased the average and maximum times of the delay.

HiCAN to HIApp different tasks

CanHwRx_AppReceive_avg	CanHwRx_AppReceive_max	AppSend_Lwip_avg	AppSend_Lwip_max	Lwip_EthHwTx_avg	Lwip_EthHwTx_max	EthHwTx_self_avg	EthHwTx_self_max	High CAN load	High app load	Separate BSW/app task
907	2052	2038	2109	37	41	6	11	yes	yes	yes
910	2077	2062	2107	35	41	6	11	yes	yes	yes
901	2055	2062	2106	37	41	5	11	yes	yes	yes

908	2054	2040	2088	36	41	6	10	yes	yes	yes
901	2073	2060	2084	35	41	5	10	yes	yes	yes
912	2077	2039	2088	35	40	6	11	yes	yes	yes
905	2076	2058	2109	36	40	6	10	yes	yes	yes
913	2057	2040	2086	35	41	6	10	yes	yes	yes
904	2076	2040	2107	35	41	6	11	yes	yes	yes
906	2057	2062	2087	35	40	6	10	yes	yes	yes

HiCAN to HiApp same task

CanHwRx_AppReceive_avg	CanHwRx_AppReceive_max	AppSend_Lwip_avg	AppSend_Lwip_max	Lwip_EthHwTx_avg	Lwip_EthHwTx_max	EthHwTx_self_avg	EthHwTx_self_max	High CAN load	High app load	Separate BSW/app task
1172	3052	2052	2077	38	41	6	11	yes	yes	no
1158	3072	2011	2061	37	42	6	10	yes	yes	no
1167	3041	2013	2080	37	41	6	10	yes	yes	no
1169	3043	2055	2072	36	40	6	10	yes	yes	no
1169	3072	2053	2082	36	41	6	11	yes	yes	no
1146	3075	2053	2065	37	42	6	11	yes	yes	no
1172	3072	2009	2067	37	42	6	10	yes	yes	no
1172	3072	2034	2083	36	42	6	11	yes	yes	no
1155	3072	2054	2069	36	41	6	11	yes	yes	no
1155	3051	2009	2066	36	43	6	10	yes	yes	no

HiCAN to LowApp different tasks

CanHwRx_AppReceive_avg	CanHwRx_AppReceive_max	AppSend_Lwip_avg	AppSend_Lwip_max	Lwip_EthHwTx_avg	Lwip_EthHwTx_max	EthHwTx_self_avg	EthHwTx_self_max	High CAN load	High app load	Separate BSW/app task

908	2045	2048	2083	35	41	6	11	yes	no	yes
903	2046	2036	2080	35	41	6	11	yes	no	yes
898	2047	2029	2084	36	41	6	10	yes	no	yes
905	2069	2031	2092	36	41	6	10	yes	no	yes
908	2063	2057	2079	36	41	6	10	yes	no	yes
896	2048	2034	2099	35	41	6	10	yes	no	yes
893	2042	2034	2087	35	40	6	10	yes	no	yes
903	2044	2052	2079	37	42	6	10	yes	no	yes
898	2063	2033	2090	36	41	6	11	yes	no	yes
895	2064	2036	2085	35	42	7	11	yes	no	yes

HiCAN to LowApp same task

CanHwRx_AppReceive_avg	CanHwRx_AppReceive_max	AppSend_Lwip_avg	AppSend_Lwip_max	Lwip_EthHwTx_avg	Lwip_EthHwTx_max	EthHwTx_self_avg	EthHwTx_self_max	High CAN load	High app load	Separate BSW/app task
1208	3074	2011	2070	35	41	6	10	yes	no	no
1205	3013	2031	2068	36	42	6	10	yes	no	no
1214	3044	1991	2071	35	40	6	10	yes	no	no
1200	3074	2031	2060	37	41	5	10	yes	no	no
1213	3074	2031	2064	36	42	6	10	yes	no	no
1210	3044	2031	2059	36	42	6	11	yes	no	no
1209	3013	1991	2073	35	42	5	10	yes	no	no
1202	3104	2011	2074	38	41	6	11	yes	no	no
1205	3013	1991	2075	35	41	5	11	yes	no	no
1208	3013	2011	2071	36	40	6	10	yes	no	no

LowCAN to LowApp different tasks

CanHwRx_AppReceive_avg	CanHwRx_AppReceive_max	AppSend_Lwip_avg	AppSend_Lwip_max	Lwip_EthHwTx_avg	Lwip_EthHwTx_max	EthHwTx_self_avg	EthHwTx_self_max	High CAN load	High app load	Separate BSW/app task
892	2043	2015	2070	36	41	6	10	no	no	yes
896	2048	2014	2090	36	41	6	10	no	no	yes
901	2038	2052	2071	35	41	6	10	no	no	yes
890	2040	2013	2089	36	40	6	10	no	no	yes
892	2038	2036	2081	36	41	6	11	no	no	yes
895	2042	2030	2082	37	42	6	10	no	no	yes
896	2043	2034	2087	35	40	6	10	no	no	yes
894	2046	2052	2093	36	41	5	11	no	no	yes
900	2039	2036	2073	36	40	6	10	no	no	yes
893	2045	2035	2068	35	40	6	10	no	no	yes

LowCAN to LowApp same task

CanHwRx_AppReceive_avg	CanHwRx_AppReceive_max	AppSend_Lwip_avg	AppSend_Lwip_max	Lwip_EthHwTx_avg	Lwip_EthHwTx_max	EthHwTx_self_avg	EthHwTx_self_max	High CAN load	High app load	Separate BSW/app task
1153	3080	2014	2076	36	42	6	10	no	no	no
1148	3021	2003	2063	35	42	6	11	no	no	no
1156	3034	2021	2083	37	41	6	10	no	no	no
1153	3003	2008	2078	36	40	6	10	no	no	no
1150	3057	2003	2080	36	42	6	10	no	no	no
1162	3064	2008	2068	35	42	6	11	no	no	no
1145	3042	2011	2074	38	43	6	10	no	no	no
1153	3056	2012	2082	35	42	6	10	no	no	no

1151	3022	2011	2065	35	40	6	11	no	no	no
1152	3034	2018	2071	36	41	6	10	no	no	no

5.3 Conclusions & Future Research

Moving towards service oriented architectures is a rising trend among many vehicle manufacturers. OEMs need to cope with an increasing number of applications, new functionalities, and new features as well as new ways of using cars. Time Sensitive Networking (TSN) Ethernet is the main enabling technology for the core network in such systems. However making TSN feasible in the vehicle networking contains number of challenges:

(1) how to guarantee fulfillment of real-time requirements of different application domains across the network
(2) how to minimize the interference on real-time traffic from non real-time traffic in the network, and
(3) CAN-Ethernet bridge strategy for the gateway

In the ECRA project, we have worked with all the above challenges from different perspectives both HW, SW and analysis tools. Our conclusions are below:

(1) In order to guarantee hard real-time constraint for control traffic, we propose to use Time Aware Shaper as defined in the IEEE 802.1 Qbv standard. For audio/video applications, we use AVB protocols with the SRP (stream reservation protocol) (IEEE 802.1Qat standard) and Credit Based Shaper (CBS) (IEEE802.1Qav standard), applied on the static network configuration. Simulation shows that we can achieve end-to-end delay of 56µs for control traffic streams and 180µs for audio traffic streams (SR class A). When we add best effort to the network, there is jitter occurring in AVB streams but at the acceptable level. The proposed network architecture is able to guarantee deliverance of control traffic with an end-to-end delay less than 90µs even without TAS.
(2) By applying TSN features including both TAS and CBS, we can prevent interference of non real-time traffic on real-time traffic and also guarantee fairness in the system.
(3) We evaluate the gateway mechanism by comparing aggregation using 1722a and UDP protocols. When the network utilization is low, both protocols perform similar. However, the UDP protocol does neither provide any fairness guarantees, nor guaranteed exclusive access to the network for high priority traffic. Thus, when the network utilization is too high, packets will be dropped. Moreover, high priority control traffic will be highly susceptible to interference, leading to unpredictable and large variations in jitter. The bottleneck constitute the link between the core processors and the network. In order to avoid packet drops, it is necessary to perform a holistic analysis including both processor scheduling and network scheduling.

Comments:

- There were few changes in the project due to personal moving as well as change in partner’s organization (Pelagicore was acquired by Luxoft and ArcCore was acquired by Vector). Thus, minor change in the scope of WP2 (which was approved by all partners and Vinnova). The project demonstrator in WP6 was slightly change as well, Luxoft has their own demonstrator; TCN and AcrCore have collaborate in another simulation tool at TCN. VCC has contributed more than original plan.
- There is engineering gap when introduce new technology to new product. TSN Ethernet is new technology, it take sometime until it can be mature and used in the production.

6. Dissemination and publications

6.1 Knowledge and results dissemination

The main mechanism for dissemination with the project partner organizations were the Project Seminars arranged in the project in cooperation between partners and invited guesses and publications at the conferences. Apart from project team and steering group members, key individuals from the partner organizations participated in these seminars.

The subsections below show the agendas and other relevant information for these seminars. The presentations were, like other results from the project, stored on the Project SharePoint site, administered by VCC and accessible for the other project partners.

6.2 Project seminar

6.2.1 The first internal workshop

- Time and place: 12th December 2016, Rise Viktoria
- Partners: VCC, Viktoria, TCN, AcrCore
- Agenda:

ECRA – 1ST WORKSHOP

Viktoria Swedish ICT, 2016-12-12

- 9.00-9.15: coffee
- 9.15 - 9.45: WP1 (Samuel)
- 9.45- 10.30: WP2 (Jonas-TCN): TAS work on progress + demo?
- 10.30-10.40: Coffe break
- 10.45-11.30: WP5 (Hoai): Requirements on Real-time Capable Automotive Ethernet Architectures
- ~~11.30 - 12.00: WP3 Pelagicore~~
- 12.00 lunch
- 13.00 end

Hoai Hoang Bengtsson

SWEDISH ICT VIKTORIA

6.2.2 The second internal workshop

- Time and place: 7th November 2017, Rise Viktoria, Lindholmen
- Participants: VCC, Rise Viktoria, TCN, ArcCore, Pelagicore
- Agenda:
    - Part 1: go through progress, deliverables of each partner
    - Part 2: discuss project demonstrator between 3 SMEs and Volvo
    - Part 3: Milestones discusssion

6.2.3 The third project external workshop

In this workshop, we have invited speakers from Technica and Spirent. Attendees were from all partners, Volvo Group and CEVT.

- Time and place: 17 May 2018, Volvo Cars Torslanda
- Pariticipants: All partners and invited guesses from Volvo Group Technology, CEVT
- Agenda

Agenda

- 9.00 – 9.20: Introduction – (Hoai & Samuel - VCC)
    - An overview of the project and Time Sensitive Networking from automotive perspective (Hoai - VCC)
- 9.20 – 10.05: Invited talk from Spirent (Thomas Schulze -Director, Marketing & Business Development – Automotive)
    - How to validate AVB/TSN functionalities under real world conditions & Standard Conformance testing
- 10.05-10.15 Coffee break
- 10.15-10.35: Overview of IEEE std 802.1CB "Frame Replication and Elimination for Reliability" and related protocols (Alexey – Rise Viktoria)
- 10.35-11.05: Gigabit Ethernet: HW aspects (Samuel – VCC)
- 1105-11.25 Demo of the ARP driving 3 displays and performing surround-view stitching of 4 cameras (Shahin – Pelagicore/Luxoft)
- 11.25-12.25: lunch
- 12.30 -13.10: Invited talk from Technica (Erick Parra - Business Development Manager)
    - Trends of Automotive Ethernet and Logging Requirements for Autonomous Driving
- 13.10-13.30: Real-time Scheduling on Linux (Jonas-TCN)
- 13.30-13.50: Demo of real-time performance measurement in an Autosar system (AcrCore)
- 13.50-14.00: Wrap up the day!

6.3 Publications

1. Hoai Hoang Bengtsson, Dorin Maxim, Ye-Qiong Song and Samuel Sigfridsson, "Reliable Real-time Ethernet Communication for advanced safety automotive applications", IEEE- SA Ethernet/IP @ Automotive Technology Day, San Jose 2017.
2. Hoai Hoang Bengtsson and Samuel Sigfridsson, "TSN Proof of Concept Evaluation of Multiple SR-Classes and Streams per Talker", Automotive Ethernet Congress, Munich 2019.
3. Björn Bergqvist, Per Ängskog, Mats Bäckström, Jerker Fors and Johnny Larsson, "Automotive-grade Gigabit Ethernet Links Robustness Against UWB Pulses, Automotive Ethernet Congress, Munich, 2019.
4. Hoai Hoang Bengtsson, Martin Hiller and Samuel Sigfridsson, "TSN Ethernet as Core Network in the Centralized Vehicle E/E Architecture: Challenges and Possible Solutions", submitted to "2019 IEEE-SA Ethernet & IP @ Automotive Technology Day".

7. Participating parties and contact persons

Partner	Contact person
Volvo Car Corporation	Hans Alminger hans.alminger@volvocars.com
ArcCore AB	Mathias Kremer (*) [i]Mathias.Kremer@arccore.com
Pelagicore AB (changed to LuxOft in Sept-2016)	Shahin Ghazinouri (**) SGhazinouri@luxoft.com
Time Critical Networks AB	Lars Bröhne lars.brohne@timecriticalnetworks.com
Viktoria Swedish ICT (changed to Rise Viktoria in 2017)	Cristofer Englund cristofer.englund@viktoria.se

(*) when the project started, contact person of ArcCore was Michael Svenstam, 0733-808 702, michael.svenstam@arccore.com

(**) when the project started, contact person of Pelagicore was Mikael Söderberg, 0708-183 485, mikael.soderberg@pelagicore.com

8. Annex

8.1 References

1. ECRA project description
2. IEEE TSN work group
3. IEEE 802.1AS
4. IEEE 802.1 Qat
5. IEEE 802.1 Qav
6. IEEE 802.1 Qbv
7. Deliverable 1.1
8. LC – TSN
9. https://www.embedded-computing.com/hardware/an-open-next-gen-reference-platform-for-tomorrows-digital-cockpits
10. Sivakumar Thangamuthu, “Analysis of Automotive Traffic Shapers in Ethernet In-Vehicular Networks”, master thesis Eindhoven University and NXP, 2015
11. IEEE Time Sensitive Networking Task Goup http://www.ieee802.org/1/pages/tsn.html
12. https://standards.ieee.org/standard/802_3bp-2016.html

8.2 Terminology

AE	Advanced Engineering
AUTOSAR	AUTomotive Open System Architecture
AVB	Audio Video Bridging
AVDECC	AVB Discovery, Enumeration, Connection management and Control
AVTP	Audio Video Transport Protocol
PTP	Precision Time Protocol
gPTP	generalized Precision Time Protocol
QoS	Quality of Service
SRP	Stream Reservation Protocol
CSB	Credit Based Shaper
TAS	Time Aware Shaper
CAN	Controller Area Network
ECAE	Ethernet Communication in Automotive Environment
ECU	Electronic Control Unit
EMC	Electro-Magnetic Compatibility
FFI	Fordonsstrategisk Forskning & Innovation
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
MOST	Media Oriented Systems Transport
OEM	Original Equipment Manufacturer (here automaker)
OPEN	One-Pair Ether-Net
TCP	Transmission Control Protocol
TSN	Time Sensitive Network
UDP	User Datagram Protocol
VLAN	Virtual Local Area Network
WCET	Worst Case Execution Time
MAC	Medium Access Control

8.3 List of contributor authors

Section	Author
1, 2, 3, 4.1, 4.4, 4.5, 5.1, 5.1.1, 5.2.1, 5.2.4, 5.2.5, 6, 7, 8	Hoai Hoang Bengtsson and Samuel Sigfridsson (VCC)
4.2, 5.2.2	Jonas Lext (TCN)
4.3, 5.2.3	Shahin Ghazinouri (LuxOft)
5.2.5.1; 5.2.5.2	Hoai Hoang Bengtsson
5.2.5.3	Alexey Voronov (Rise Viktoria)

4.6, 5.2.6	Kristofer Hallen (ArcCore)

8.4 List of Figures