IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 27 October 2023, accepted 9 January 2024, date of publication 12 January 2024, date of current version 23 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3353466

RESEARCH ARTICLE

Proposal and Investigation of a Lite Time Sensitive Networking Solution for the Support of Real Time Services in Space Launcher Networks

TIZIANA FIORI, (Graduate Student Member, IEEE), FRANCESCO GIACINTO LAVACCA, FRANCESCO VALENTE, (Graduate Student Member, IEEE), AND VINCENZO ERAMO, (Member, IEEE)
DIET Department, Sapienza University of Roma, 00184 Rome, Italy
Corresponding author: Tiziana Fiori (tiziana.fiori@uniroma1.it)

ABSTRACT Most launcher networks are based on proprietary buses such as MIL-STD-1553B whose low bandwidth limits the introduction of new services of suitable characteristics. Ethernet technology, because of its low cost and high performance, has been considered an excellent candidate for its use in launcher networks. The real time Ethernet solutions based on the Time Sensitive Networking (TSN) standards seem the most suitable because of its multi-vendor product characteristics. In this paper we propose a real time Ethernet solution for aerospace applications in which negligible jitter services has to be guaranteed. The proposed solution is based on the following TSN standards: IEEE 802.1AS/ASrev as synchronization protocol and 802.1Qbv-2015 for deterministic traffic scheduling. To improve both the bandwidth effective and the frame delay the solution is also based on a change in the management of the Priority Code Point field in IEEE 802.1Q standard. The optimal scheduling problem is formulated so as to minimize the makespan, defined as the time needed to deliver all of the messages of an elementary cycle. The problem has been resolved with the CPLEX solver and the proposed solution has been evaluated in terms of both delay and bandwidth effective by comparing its performance with the TTEthernet, FTTEthernet benchmark solutions. The obtained results in a real traffic scenario characterized by the set of messages of the VEGA launcher show how the proposed solution allows for the same performance of TTEthernet, i.e., the solution of proprietary and real-time Ethernet with better performance.

INDEX TERMS Time sensitive networking, TTEthernet, FTTE, scheduling optimization.

I. INTRODUCTION

Future aerospace intra-communication systems must be designed with new, acceptable communication infrastructures intended to overcome the restrictions associated with well-tested aerospace protocols. This allows to overcome the bandwidth constraints of the current most widely used solution, MIL-STD-1553B, whose limited 1 *Mbps* bandwidth is no longer adequate for modern high-speed applications requiring faster data transfers, [1], [2], or large deployment costs and lack in flexibility of other alternatives, like TTEthernet, used as communication system in Ariane 6 [3] and on board for NASA's Artemis I mission [4]. This trend has been verified by the introduction of the new micro-launcher technologies in the context of the so-called "New Space" [5]. These launch vehicles are typically intended to release relatively modest payloads and, given this reduced size of the payload, these projects are frequently driven by the reusability of the launching vehicles themselves or the viability of a quick production. Due to this paradigm shift, manufacturers are attempting to use commercially available off-the-shelf (COTS) components in order to

The associate editor coordinating the review of this manuscript and approving it for publication was Ayaz Ahmad.

minimize development effort and increase interoperability with standards emerging from industrial applications or other domains. As a result, the companies driving the design of micro-launchers are opting for avionics intra-communication systems based on Ethernet technologies, because of its high bandwidth and interoperability. However, traditional Ethernet networks cannot provide the necessary real-time transfer needs, hence they are not suited for aerospace networks [6], [7], [8].

The standardization efforts within the IEEE 802.1 Time-Sensitive Networking (TSN) task group which is part of the IEEE 802.1 Working Group have lead to the introduction of a set of standards for the support of real time and reliable communications in Ethernet networks. TSN is a set of standards covering 4 categories: resource management, shaping/scheduling, time synchronization, and reliability. The TSN is based on some cores standards for supporting synchronization capabilities (IEEE 802.1AS/ASrev), deterministic scheduled traffic (IEEE 802.1Qbv-2015), frame preemption of low-critical frames by high-critical frames (IEEE 802.1Qbu-2016), traffic filtering and policing (IEEE P802.1Qci-2017), redundant frame transmission (IEEE 802.1CB-2017) and network configuration (IEEE 802.1Qcc) [9].

TSN is currently offered for custom designs as well as COTS controllers from several company [10], [11].

The selection and implementation of TSN standards are highly related to the specific application. The TSN system implementation details, such as network topology, device buffers, queue mechanisms, time synchronization precision, need to be determined by designers, and the selected standards should be carefully customized according to the application requirements. By employing only the standards strictly necessary to meet the demands of the particular application, network configuration and management are simplified. In particular we are interested to defining a real solution Ethernet solution for launcher networks able to guarantee negligible jitter services. It will have to satisfy two main requirements: a synchronization of the network elements and a message deterministic scheduling. For this reason the proposed solution, referred to as TSN-Lite is based on the IEEE 802.1AS/ASrev synchonization and IEEE 802.1Qbv scheduling standards. Their application ensures a defined maximum latency for scheduled traffic via switched networks by constructing queues that transmit their messages according to a specified schedule.

The main contributions of the paper are the following:

- The definition of TSN-Lite in two different versions, TSN-Lite-v1 based on the standards IEEE 802.1AS/ASrev and IEEE 802.1Qbv and TSN-Lite-v2 that enriches TSN-Lite-v1 with the addition of a change to the management of the Priority Code Point filed in the standard IEEE802.1Q;
- The definition of the optimization problem for the evaluation of the message scheduling instants for TSN-Lite-v1 and TSN-Lite-v2;
- The evaluation of TSN-Lite-v1 and TSN-Lite-v2, in a real-world network environment using VEGA messages, a launcher developed by the Italian Space Agency (ASI); including a comparison with FTTE and TTEthernet state-of-the-art solutions; in particular we will compare the results of the optimization problems.

The paper is organized as follows. The related work is described in Section II. A general overview of the launcher network is described in Section III. The state of the art real-time Ethernet solutions are described in Section IV. Section V describes the TSN-Lite solutions and Section VI formulates the message scheduling optimization problems. The numerical results are presented in Section VII and Section VIII reports the conclusions.

II. RELATED WORK

The avionics subsystems of space launchers or satellites typically use buses that are based on protocols, such as the well-known MIL-STD-1553B. While this bus ensures the determinism and dependability crucial for aerospace applications, its constrained 1 *Mbps* bandwidth has become inadequate. As aerospace networks now contend with larger data volumes, including images, voice, and video streams, there is a growing requirement to upgrade solutions by increasing data transmission rates and enhancing overall performance. Designing an Ethernet-based solution can achieve performance increase in terms of available bandwidth. Some Ethernet-based solutions have been proposed to conform Ethernet standard to be able to handle real-time features.

A compelling solution is the Time Triggered Ethernet (TTEthernet), which has garnered attention for aerospace applications due to its ability to provide real-time communication with predictable latency [12]. TTEthernet [13], [14], [15], provides synchronization and establishes temporal partitioning for high criticality data flows, but it is a proprietary technology associated with a mono-vendor market and high costs.

To face the temporal requirements, an innovative communication system, the Flexible Time Triggered Ethernet (FTTE), for a next-generation launcher has been proposed in [16]. This solution involves the use of traditional Ethernet switches and the definition of a scheduling algorithm for messages running in the end systems and able to avoid bandwidth resource contention in the network links, [7], [8], [17]. FTTE is a promising solution but it is not mature yet for a short-term application.

The Time Sensitive Networking (TSN) group task was founded in the Institute of Electrical and Electronics Engineers (IEEE) working group 802.1 to study new solutions supporting real-time services in Ethernet networks [18], [19], [20], [21].

TSN focuses on the IEEE 802.1Qcr [22] Asynchronous Traffic Shaper (ATS) and the IEEE 802.1Qbv [19] Time-Aware Shaper (TAS). ATS is a token bucket based approach applied in the end systems and the switches but is not suited for launcher networks because it is not able to guarantee

deterministic delay to periodic traffic flows. Conversely TAS defines a mechanism for time-driven control and scheduling of data frames. It offers the opportunity, for every TSN device to have up to 8 traffic shapers associated to up to eight transmission queues per port; Time Aware Shaper allows defining time windows in which frames can be emitted. The time-aware shaper is essentially a gate enabling or disabling the transmission of frames for a queue following the specification of a periodic schedule. Here it is important to note that the schedule is defined on the level of traffic classes and not of individual frames like in TTEthernet. Scheduling entire traffic classes as opposed to individual frames provides more flexibility for use-cases where strict timing constraints and determinism on the level of streams are not the most important aspects. Since the traffic class is defined in the PCP code of the VLAN tag of frames, using only IEEE 802.1Qbv cannot enable a fined-grained identification and control on the level of streams. Additional mechanisms like the per-stream identification and filtering (defined in IEEE 802.1Qci/802.1CB), allowing identification of frames based on a stream identifier and overriding of the traffic class encoded in the PCP code, are necessary if we want to achieve the same level of determinism as in TTEthernet [23], [24].

To avoid the use of the IEEE 802.1Qci/802.1CB standards, we propose a real time Ethernet solution, referred to as TSN-Lite and based on two standards IEEE 802.1AS/ASrev and IEEE 802.1Qbv. Performance is evaluated in terms of makespan defined as the time needed to deliver all of the messages of an elementary cycle. Other papers have assessed and minimized makespan for TSN, such as [25], [26], and [27], but none of the cited works have conducted evaluations of minimized makespan for lightweight versions of TSN, as proposed in this study with TSN-Lite. We carry out a comparison of TTEthernet, FTTE, and TSN-Lite in a case study of a real avionic application.

III. LAUNCHER COMMUNICATION SYSTEM

First we report a generic overview of a launcher communication system, the next section describes real-time Ethernet options that were considered for the particular space application. A general overview of the communication system of a launch vehicle is illustrated in Fig. 1. A launcher executes various phases and the main high-level operations of a mission are listed below:

- Ground: the vehicle is getting ready for its launch campaign during this phase. The launcher components are powered up, all system and payload components, such as propulsion and avionics, are verified at the launch pad, tanks are filled and pressurized, and the ground activities are carried out until the vehicle control is transferred to the On-Board Computer (OBC).
- Lift off and flight: the umbilical is freed from the launcher when the engine has been started and the motor has been ignited, the Lift off of the vehicle has started and the launcher begins the ascent phase where it is controlled (primarily by the Trust Vector Control (TVC) system) to follow an ideal trajectory profile.

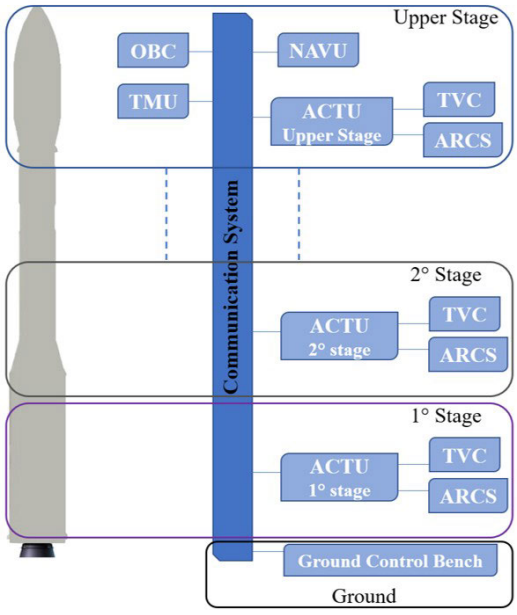


FIGURE 1. Generic launcher communication system and equipment distribution.

A division of the flight itself stands out and implies that the avionic architecture has been broken down into modules, each of which has features that are closely tied to the stage to which it is attached. The launcher needs to use a comprehensive avionics system to execute the task. As a result, each stage has its own avionics system and it is composed by a Guide Navigation & Control (GNC) subsystem, On-Board Software, Data Handling subsystem, responsible for gathering sensor data, operating valves, and formatting telemetry, and other subsystems. The GNC is the module in charge of maintaining the launcher's target mission trajectory and attitude, primarily through attitude control via Trust Vector Controls. In order to determine the launcher state that the GNC requires, the navigation function receives the measurement data from the inertial sensors. The angular rates and accelerations that the inertial sensor measures are directly integrated into the navigation algorithms. The launcher communication system is therefore equipped with a Navigation Unit (NAVU) in the upper stage. To comply with the mission maneuvers and schedule, the launcher must maintain the reference attitude and reference angular velocity calculated by the guidance function. The control function computes the attitude control for the TVC and Attitude and Rate Control System (ARCS) based on the information received by the navigation and guiding system. The guidance provides the reference thrust direction, and reference attitude. The actuation units (ACTUs), one in each stage of the rocket, receive this data from the OBC, which is located in the upper stage. Finally, the master stage located in the upper stage operates during the entire mission and

supervises the other stages, so it is also equipped with a Telemetry Master Unit (TMU).

Overall, the avionic intra-communication system supports two types of messages:

- Time Triggered (TT) messages: they represent the cyclic GNC messages and the sporadic messages. The GNC must be transmitted in real-time and error-free because the predefined orbital accuracy performances are guaranteed by the cyclic GNC mechanism. The GNC algorithms specifically process the attitude, velocity, and position data, which have already been transformed into engineering format, and compute new command data for the actuators in order to follow an optimal trajectory. In addition, there are commands that are only executed to trigger crucial events, thus the sporadic messages must be issued without designating predetermined time intervals (such as the separation of a stage, the ignition of an engine). Nevertheless, due to the fact that they relate to important events, even these occasional signals must be transmitted with a deterministic delay.
- Telemetry messages: they allow for ground visibility and supervision of onboard activities; this is possible using the TMU. Some telemetry data can be simply supplied when it is possible, while other telemetry data may need to be sent promptly in order to identify abnormalities and determine the progress of the launch. Two categories of data will be managed by the TMU and Ground: i) the Contrôl Visuel Immediate (CVI) messages, transmitted to the ground and made available to mission control right away on the screen; ii) the Contrôl Visuel Différé (CVD) messages, sent to the ground as soon as possible and stored there for post-processing.

The majority of launchers organize message transmission into time blocks known as Elementary Cycles (EC) of duration T_{EC}. For instance, the VEGA launcher created by ASI has defined EC of duration 5 ms [28]. Two terminals exchanging GNC messages provide a periodic message flow with a period multiple of T_{EC}. The GNC messages of the same flow must be transferred with the same network delay since all GNC processes are synchronized, and as a result, they require a service with zero jitter. The deterministic nature of the traffic in launcher networks causes an admission control procedure to only be used in the initial launch phase, where bandwidth resources are statically allocated to each GNC flow to ensure deterministic delays while only maximum delay constraints are verified for telemetry messages.

IV. REAL TIME ETHERNET SOLUTIONS FOR A LAUNCHER COMMUNICATION SYSTEM

A. TIME-TRIGGERED ETHERNET

Time-Triggered Ethernet (TTEthernet) adds robust services to traditional Ethernet to satisfy the demands of fully deterministic communication. These services include fault-tolerant synchronization, guaranteed constant latency for multi-hop communication pathways in a network and

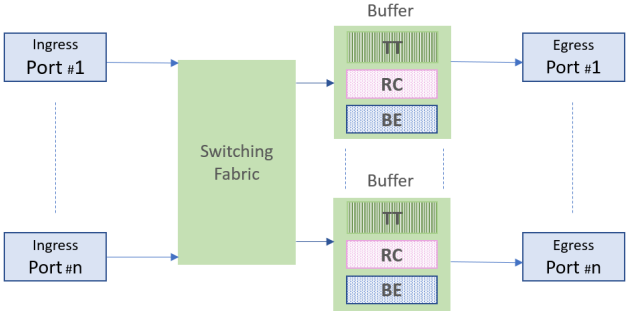


FIGURE 2. TTEthernet switching architecture.

traffic segregation and protection. In application sectors like aerospace, TTEthernet can be used as a single real-time network and backbone network solution. In this scenario, a single network can incorporate the entertainment system, electronic navigation and guidance system, and in-seat internet access. In TTEthernet, Time Triggered (TT) traffic produced by critical applications will always take precedence over non-critical application traffic. The message temporal behavior can be modified based on the desired quality and is predictability. If the network is expanded, due to the change of other apps, an existing critical application does not need to be altered in terms of functionality or scalability. By synchronizing the local clocks of all Time-Triggered Ethernet devices, these time-triggered methods create and uphold a universal time. The implementation of temporal segmentation, accurate diagnosis, effective resource use, or composability all start with the global time. TTEthernet uses the global time Universal Coordinated Time (UTC) as the basis for synchronization, which is compatible with the time format of the IEEE 1588 standard. It relies on the Precision Time Protocol (PTP) [29] to send synchronization packets and measure transmission times, aligning device clocks through hardware timestamping that instantly records packet reception and transmission times, aiding in precise network delay measurements. The following three traffic classifications are supported by this solution: i) TT traffic, sent in a time-triggered way; each TTEthernet sender node has a transmit schedule, and each TTEthernet-Switch has a receive and forward schedule; ii) Rate-constrained (RC) traffic, controlled by shaping and policing processes; iii) Best Effort (BE) traffic, transmitted with no timing guarantees.

The frame format is compatible with the IEEE 802.3-compliant frame format for Ethernet. Switches play a crucial part in processing communication data in TTEthernet. The switch processes TT messages in accordance with a predetermined timetable. RC messages are not transmitted with respect to a system-wide synchronized time base, in contrast to TT messages. As a result, various communication controllers may send RC signals to the same receiver at the same time. The RC messages could therefore accumulate in the network switches, increasing transmission jitter. The TT messages take precedence over the RC messages if they

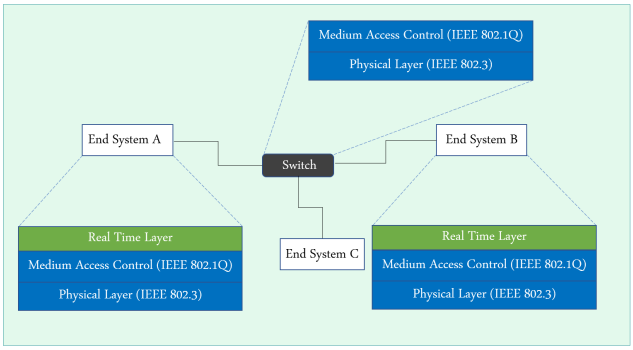


FIGURE 3. FTTE architecture.

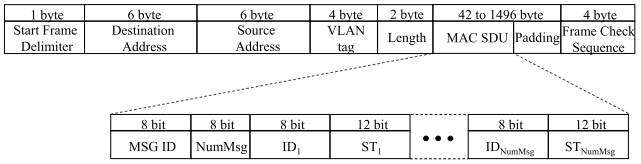


FIGURE 4. Trigger message in a frame 802.1Q.

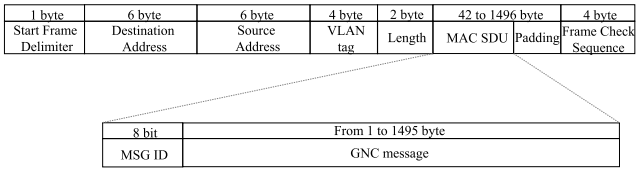


FIGURE 5. GNC message in a frame 802.1Q.

are to be sent simultaneously over the same switch outgoing port as shown in Fig. 2. BE messages are forwarded as classical Ethernet messages. It is crucial to keep in mind that TTEthernet devices dispatch frames in accordance with the stated schedule, meaning that the TTEthernet schedule is set at the level of individual frames.

B. FLEXIBLE TIME TRIGGERED ETHERNET (FTTE)

The Flexible Time Triggered Ethernet (FTTE) solution is based on the use of 802.1Q [30] traditional Ethernet switches, which ensure low costs, especially if COTS Ethernet switches are used. The 802.1Q standard allows for the configuration of VLANs regarding communications involving two any devices and the telemetry system. The use of traditional Ethernet switches, however, would not allow for obtaining the time guarantees required by a space application, for this reason, the FTTE solution is based on two aspects:

- The implementation of a Real-Time Layer, whose primary role is medium access control that prevents bandwidth usage and enables the support of a service with zero jitter; in order to prevent link bandwidth contention, the Real Time layer is only implemented in the network end systems as shown in Fig. 3. A Master End System (MES), which can be the OBC, evaluates the GNC messages scheduling times.
- The MES distributes the scheduling instances to all end systems. They are inserted in a message known as the Trigger Message (TM), which is also periodically transmitted to synchronize all of the end systems and may be sent at the start of each EC.

The Service Data Unit (SDU) of the TM, contains a list of identifiers of all messages which can be sent in that EC and, for each TT message, the time delay from the beginning of the EC at which that message shall be sent, determined by the a priori scheduling algorithm, is indicated. Each end system then receives the TM, and whenever it locates a message identifier in the list of TM messages identifiers for which it is the sender, it transmits the message at the time specified by the scheduling and indicated by the TM. This is the reason why the TM frame has a different structure from other GNC messages. As shown in Figs. 4 and 5, the TM and GNC messages are transmitted on conventional Ethernet frames according to IEEE 802.1Q [30]. For all message typologies, frames will be characterized by the presence of standard Medium Access Control (MAC) frame fields (Destination Address, Source Address, Length/Type, Frame Control Sequence) and VLAN tag, since IEEE 802.1Q standard is used. The structure of the TM is composed by the following main fields:

- MSG_{ID}: 8 bits field which identifies the Trigger Message.
- *NumMsg*: a field of 8 bits that indicates how many messages are scheduled to be sent in the EC where the TM is being sent.
- ID_i $(i = 1, \cdots, NumMsg)$: 8 bits field that reports the identifier of the i-*th* $(i = 1, \cdots, NumMsg)$ GNC message scheduled in the EC in which the TM is sent.
- ST_i $(i = 1, \cdots, NumMsg)$: 12 bits field that reports the sending time of the i-*th* $(i = 1, \cdots, NumMsg)$ GNC message scheduled in the EC in which the TM is sent.

When an end system delivers the GNC message in the planned instant, as shown in Fig. 5, the GNC message and its MSG_{ID} are conveyed in the MAC SDU.

C. TIME SENSITIVE NETWORKING (TSN)

In order to increase the determinism of Ethernet-based networks, the Institute of Electrical and Electronic Engineers (IEEE) established the Time Sensitive Networking (TSN) Ethernet extension [9]. is based on a set of standards that allow for a precise synchronization, packet prioritization, bandwidth guarantee, and fault tolerance mechanisms. Precise synchronization relies on high-precision protocols to align network device clocks, enabling real-time coordination. Packet prioritization assigns priority levels to packets, ensuring timely transmission of critical data. Bandwidth guarantee reserves sufficient bandwidth for critical traffic, preventing network congestion. Fault tolerance mechanisms provide redundancy and recovery from failures, ensuring continuous communication. For the support of negligible jitter services, two standards defined by the TSN group are of interest: IEEE

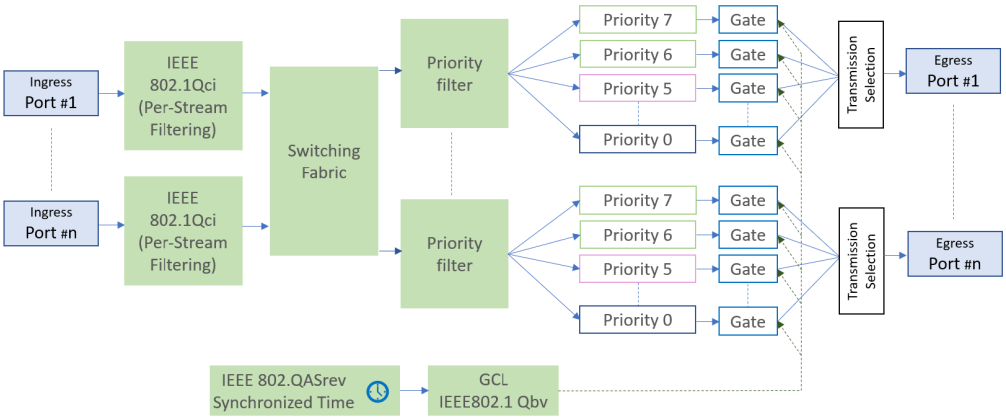


FIGURE 6. TSN switching architecture.

802.1ASrev and IEEE802.1Qbv [9]. The IEEE 802.1ASrev is a specific application of the Precision Time Protocol (PTP) synchronization protocol for Ethernet networks. This means that IEEE 802.1AS customizes the time synchronization functionalities of PTP specifically for Ethernet networks, ensuring that devices within an Ethernet network can achieve highly accurate time synchronization using the PTP protocol as its basis. The TAS adds time-controlled gates to the queues at the egress ports to provide a Time Division Multiple Access (TDMA) scheme for IEEE Ethernet. These gates allow for the temporary disabling of one or more queues to give high-priority traffic exclusive network access. With enough time synchronization, these gates–which normally operate on a queue basis–can be utilized to schedule frames in a stream-wise manner. The fundamental idea behind TT communication is quite straightforward: the IEEE802.1Qbv creates a communication timetable that tells the end stations when to send specific frames to the network. Instead of explicitly scheduling frame transmissions (as in TTEthernet), IEEE802.1Qbv plans the activation and deactivation of queues. To implement the activation and deactivation of a queue, IEEE802.1Qbv introduces a gate for each queue. The gate is in the open position to activate the queue; conversely, it is in the closed position to stop it. The communication schedule in IEEE802.1Qbv is realized by a Gate Control List (GCL), whose entries specify the points in time when to set the gate state into the open/closed state for each port. It is significant to notice that, unlike TTEthernet, the schedule in this case is established at the level of queues rather than individual frames. The transmission selection with gates is depicted in Fig. 7.

The queue selected by the switch to store an arriving frame is determined by the Priority Code Point (PCP) values of the three bits VLAN TAG field of the standard IEEE802.1Q. The encoding of the VLAN TAG leads to the need to implement up to eight queues.

There is more flexibility when scheduling at the level of queues as opposed to individual frames for use cases where severe time constraints and stream-level determinism

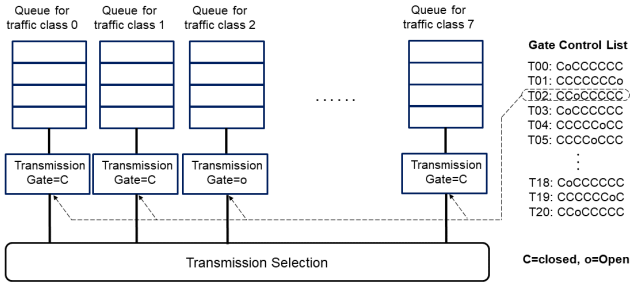


FIGURE 7. Transmission selection with gates.

3 bits	1 bit	12 bits
PCP	DEI	VID

FIGURE 8. VLAN TAG control information format.

are not the most crucial factors. If we want to achieve the same level of determinism as TTEthernet, we can use the per-flow identification and filtering (specified in IEEE 802.1Qci), allowing identification of frames based on a VLAN Id and overriding the PCP field. In particular the IEEE 802.1Q specification establishes a standardized method for implementing VLAN tagging and class of service in Ethernet networks. The fields within the 802.1Q tag are the following, as reported in Fig. 8:

- Priority Code Point (PCP): A 3-bit field is used to denote the level of priority assigned to the frame.
- Drop eligible indicator (DEI): 1-bit field not used to our purpose and allowing for a frame discarding priority of messages.
- VLAN Identifier (VID): A 12-bit field is utilized to designate the VLAN of the frame.

IEEE 802.1Qci enhances determinism in Ethernet networks by providing a mechanism for reserving resources. While it may appear to be focused on stream filtering, its impact on determinism is significant because it helps to create a controlled and predictable network environment, which is crucial for meeting the demanding requirements of

applications like TTEthernet or other real-time systems. Assigning a PCP value to each flow enables precise control over the path of the flow, optimizing resource management and ensuring adherence to quality of service criteria. Leveraging various PCP values facilitates the allocation of flows to separate priority queues, thus enhancing the efficiency of network resource distribution. For each stream, a PCP is assigned, which can vary for each link traversed by the flow. We want to provide the same degree of determinism on the level of frames in a TSN environment. Hence, we need to derive, in addition to the TTEthernet constraints, specific 802.1Qbv constraints. We will later explore the implications of this on scheduling constraints.

To support negligible jitter services in Ethernet network we need to extend it with the protocols defined in the following TSN standards: IEEE 802.1Qci for traffic filtering, IEEE 802.1ASrev for clock synchronization, IEEE 802.1Qbv for traffic scheduling. Next the real time Ethernet solution based on the TSN standards IEEE 802.1Qci, IEEE 802.1ASrev and IEEE 802.1Qbv will be referred to as TSN-Target. The TSN-Target switch architecture is depicted in Fig. 6, incorporating the key IEEE 802 standards (to enable deterministic and time-sensitive communication in aerospace environments.

V. LITE VERSIONS OF TIME SENSITIVE NETWORKING

In this section, we describe TSN-Lite which is based on the essential time sensitive standards for achieving good temporal guarantees. We propose two versions of TSN-Lite referred to as TSN-Lite-v1 and TSN-Lite-v2 respectively. In both cases, the IEEE802.1Qci standard is omitted. The first version, TSN-Lite-v1, includes the traditional standards IEEE802.1ASrev and IEEE802.1Qbv, without further modifications. However, this approach does not achieve the same level of performance as TTEthernet. Conversely, TSN-Lite-v2 involves a modification to the embedded IEEE802.1Q standard to attain the same level of performance as the existing solution currently employed by TTEthernet launch systems. The introduction of TSN-Lite-v2 is crucial to achieve a competitive performance level on par with TTEthernet, without adding further filtering standards, thereby maintaining simplicity.

A. TSN-LITE-V1

As previously stated, to attain an equivalent level of determinism as TTEthernet, we can employ the per-flow identification and filtering mechanisms outlined in IEEE 802.1Qci(802.1CB. However, the use of these additional standards makes the implementation and configuration very complex. Filtering, as stipulated by the IEEE 802.1Qci standards, enables the regulation of data flow paths within the network, facilitating the selection of specific network segments or nodes through which data flows should be transmitted. This regulatory mechanism guarantees the allocation of adequate bandwidth and latency to prioritized or critical data flows, ensuring compliance with determinism and quality of service requirements. The IEEE 802.1Qci standard defines a comprehensive set of rules and criteria often based on the PCP parameter and other frame fields for identifying and filtering data flows within the TSN network. By overriding a PCP value with each flow, it becomes possible to exert granular control over the path of the flow, optimizing resource management and ensuring compliance with quality of service requirements. The utilization of different PCP values allows for the assignment of flows to distinct priority queues, maximizing the efficiency of network resource allocation.

TSN-Lite-v1 is based IEEE802.1ASrev for syncronization and IEEE802.1Qbv for time-aware-shaper scheduling. It uses the capability provided by the IEEE 802.1Q standard to change the PCP value when a frame is received. as reported in paragraph 6.9.4, page 160 [30]. Typically, the relay function of a bridge does not modify priority during transit. However, if useful as in this case, bridges can regenerate the PCP based on signaled information and stored configuration data, enabling PCP changes on each receiving port for efficient resource management. However, assigning different PCP regeneration rules to multiple flows passing through a receiving port is not possible. Thus, achieving the same level of determinism and bandwidth efficiency as TTEthernet is not feasible with TSN-Lite-v1. An example of TSN-Lite-v1 is depicted in Fig. 9, where we have two flows, Flow #1 and Flow #2, passing through the same reception port of the switch. The Flow #1 and Flow #2 are characterized by the Virtual IDentifiers (VID) 7 and 8 respectively and the same Input Priority Code Point (IPCP) equal to 4. Because the current TSN version allows for the support of one PCP regenerative table only per input port, the frames of the Flow #1 and Flow #2 will be constrained to be directed to the same output queue with Egress Priority Code (EPCP) equal to 6.

B. TSN-LITE-V2

TSN-Lite-v2 is an extension of TSN-Lite-v1 with the addition of a change to IEEE 802.1Q in order to attain a comparable level of performance and determinism as the case incorporating the filtering standards.

In typical scenarios, the relay function of a bridge does not alter the priority during transit. However, there might be instances where it is necessary to have control over how the priority is propagated for management purposes. In such cases, the bridges have the capability to regenerate the PCP, as in the case TSN-Lite-v1 by considering both signaled information and configuration data stored within the bridge, and therefore a PCP change can be performed on each receiving port to achieve efficient resource management. However TSN-Lite-v1 lacks flexibility because if multiple flows pass through a receiving port, it is not possible to assign different regenerated PCPs to them. By proposing a modification to IEEE 802.1Q that allows for the regeneration of a distinct PCP for each differential flow at each receiving port, an optimal resource management can be achieved, similar to the case of TSN with IEEE 802.1Qci/CB.

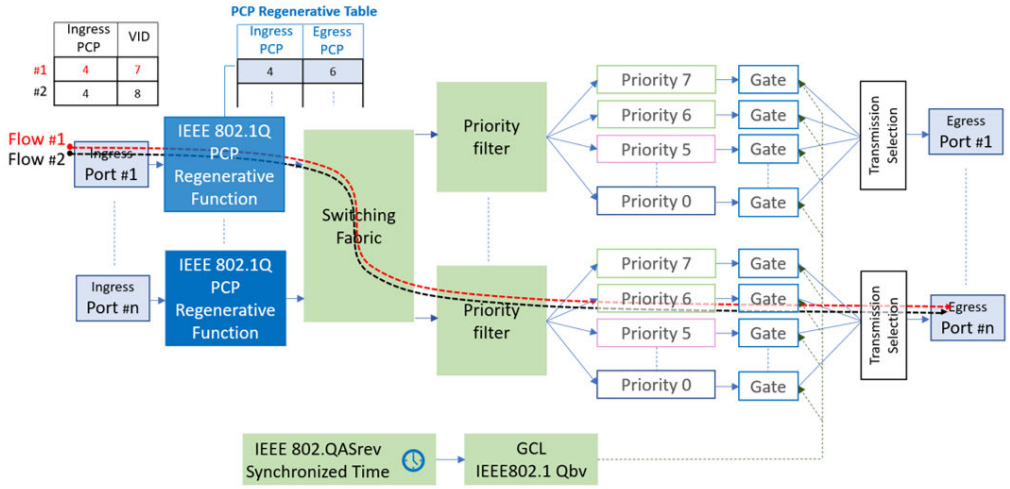


FIGURE 9. Architecture of TSN-Lite-v1 Switch. The two flows, denoted as Flow #1 and Flow #2, traverse the same receiving port. It is possible to assign a regenerated PCP to them, (in this example equal to 6), which must be the same for both since they pass through the same receiving port.

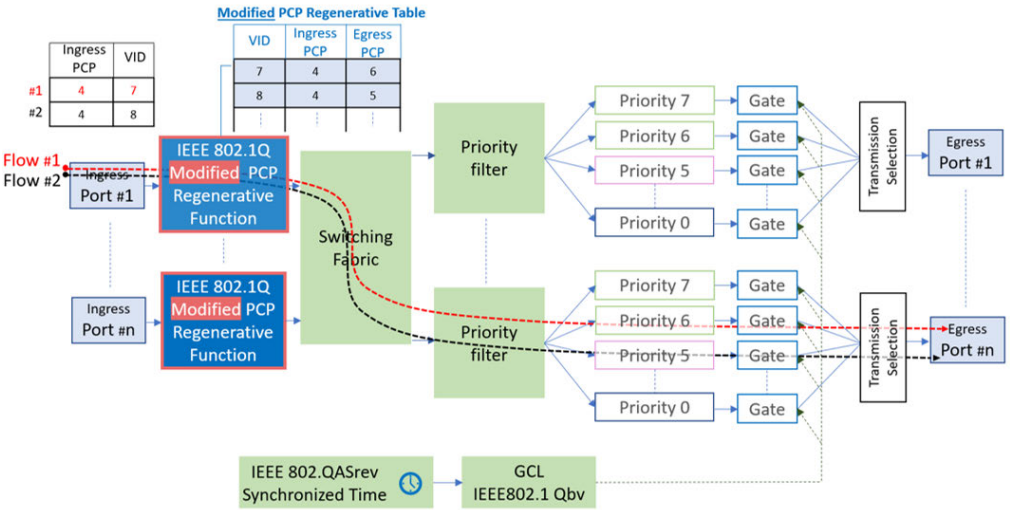


FIGURE 10. TSN-Lite-v2 switch. The two flows, denoted as Flow #1 and Flow #2, traverse the same receiving port. It is possible to assign different regenerated PCPs to the two flows, even though they pass through the same receiving port.

Therefore, we propose TSN-Lite-v2, an extension of TSN-Lite-v1 with the modification as shown in Fig. 10, whose scheduling constraints will be outlined in the optimization problem section. As depicted in Fig. 10, there are two flows, namely Flow #1 and Flow #2, passing through the same reception port of the switch. Both of these flows can be assigned regenerated PCP values. Despite sharing the same reception port, it is possible to assign distinct regenerated PCP values to each flow. That can be be realized by introducing an additional column in the PCP regenerative table identifying the flow. For example a column VID may be inserted. The switch is shown in Fig. 10 where the VIDs 7 and 8 are reported for the flows #1 and #2 respectively. In such a way the frames of the flows #1 and #2 can be addressed towards two different output queues with PCP equal to 6 and 5 respectively.

TABLE 1. Distinguishing variances in TSN.

Solution	Scheduled Traffic	Synchronization	Assignment of Priorities
TSN-Target	IEEE 802.1Qbv	IEEE 802.1ASrev	IEEE 802.1Qci
TSN-Lite-v1	IEEE 802.1Qbv	IEEE 802.1ASrev	IEEE 802.1Q PCP Regenerative Function
TSN-Lite-v2	IEEE 802.1Qbv	IEEE 802.1ASrev	IEEE 802.1Q PCP Modified Regenerative Function

VI. TSN-LITE OPTIMIZATION PROBLEM FOR MESSAGE SCHEDULING

In this section, we delve into the network and traffic modeling, as presented in Subsection VI-A. Additionally, we provide an illustrative explanation of the optimization problem in Subsection VI-B for the TSN-Lite-v1 and TSN-Lite-v2.

A. NETWORK AND TRAFFIC MODELING

The network is a multi-hop layer 2 switched Ethernet with full-duplex physical links and the network model is defined as a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{L}$ is the set of links between nodes. A full-duplex physical link between nodes $n_a \in \mathcal{V}$ and $n_b \in \mathcal{N}$ results in two directional logical links, each denoted by an ordered tuple, namely $[n_a, n_b] \in \mathcal{L}$. Each physical link $[n_a, n_b]$ is described by the tuple $<[n_a, n_b].s, [n_a, n_b].d, [n_a, n_b].mt>$ where $[n_a, n_b].s$ is the speed of the link, $[n_a, n_b].d$ is the propagation delay on the link, and $[n_a, n_b].mt$ is the macrotick of the link. Slots are used to divide time.

The origin node sends messages to the destination node. A stream (flow) is defined as a periodic data transmission from one sender (talker) to one or multiple receivers (listeners). We define TT message flows, sometimes known as streams, and define $\mathcal{S}$ as the set of all currently active streams in the network. A stream $s_i \in \mathcal{S}$, which travels from a sender node n_a to a receiver node n_b, is $s_i = [[n_a, n_1], [n_1, n_2], \ldots, [n_n, n_b]]$, where $n_1, n_2, \ldots, n_n$ indicates the stream's path through the network. A stream is characterized by the tuple $<s_i.L, s_i.T, s_i.\delta>$, where $s_i.L$ is the stream data size, $s_i.T$ is the stream period, and $s_i.\delta$ indicates the initial offset of stream i. A stream $s_i \in$ S on a link $[n_a, n_b] \in \mathcal{L}$ is denoted by $s_i^{[n_a,n_b]}$. Flows are assumed to follow a deterministic periodic behavior and the message scheduling needs to be performed in a Hyper-Cycle (HC) whose duration is given by the least common multiple of the time periods $s_i.T$ ($\forall i \in \mathcal{S}$). Let us denote with T_{HC} the duration of the time period of the HC. The initial offset $s_i.\delta$ indicates the initial absolute temporal offset of stream i from the beginning of the HC.

We denote the set of frames $f_{i,k}^{[n_a,n_b]}$ of a stream $s_i^{[n_a,n_b]}$ by $\mathcal{F}_i^{[n_a,n_b]}$. A frame $f_{i,k}^{[n_a,n_b]} \in \mathcal{F}_i^{[n_a,n_b]}$ is defined by $<f_{i,k}^{[n_a,n_b]}.\phi, f_{i,k}^{[n_a,n_b]}.T, f_{i,k}^{[n_a,n_b]}.L, f_{i,k}^{[n_a,n_b]}.\delta>$ scaled to the link macrotick, where $f_{i,k}^{[n_a,n_b]}.\phi$ is the scheduled transmission instant of the frame on link $[n_a, n_b]$, $f_{i,k}^{[n_a,n_b]}.T$ is the period, $f_{i,k}^{[n_a,n_b]}.L$ is the transmission duration of the frame on the link, and $f_{i,k}^{[n_a,n_b]}.\delta$ is the initial offset of the frame on the link. Frames of two different flows, s_i and s_j, might have different periods. Consequently, they are repeated a different number of times within their hypercycle. We denote the hypercycle of the two streams by $hc_i^j = lcm(s_i.T, s_j.T)$, in which $lcm(s_i.T, s_j.T)$ is the least common multiple of the two periods. We indicate with α and β the set of translations of the frames of flow s_i and s_j within the hypercycle, formulated as $\alpha \in [s_i.\delta,\ s_i.\delta + \frac{hc_i^j}{s_i.T} - 1]$, and $\beta \in [s_j.\delta,\ s_j.\delta + \frac{hc_i^j}{s_j.T} - 1]$. We introduce an additional variable specifying the assigned queue for the instance of a stream on a particular device, $s_i^{[n_a,n_b]}.p$. Keep in mind that the queue variable corresponds to the stream priority within the device egress port. Let $r_{i,j}^{[n_a,n_b]}$ be a binary indicator with $r_{i,j}^{[n_a,n_b]} = 1$ if stream s_i and stream s_j are assigned to the same priority $s_i^{[n_a,n_b]}.p = s_j^{[n_a,n_b]}.p$, and $r_{i,j}^{[n_a,n_b]} = 0$ otherwise, for $[n_a, n_b] \in \mathcal{L}$. Table 2 provides a list of the network model set and constants. IEEE 802.1Qbv specifies a time-based shaper feature for scheduling, that enables time-triggered communication at the egress ports. In essence, a time-aware shaper is a gate that, in accordance with the specification of a periodic schedule, enables or disables the transmission of frames for a queue. The frame scheduled transmissions $f_{i,k}^{[n_a,n_b]}.\phi$ represent the open and close events for the timed-gate of the assigned queue of the frame/stream.

TABLE 2. Attributes of the network model.

Notation	Description
$\mathcal{N}$	Set of network nodes
$\mathcal{L}$	Set of network links between nodes $\mathcal{N}$
$\mathcal{G} = (\mathcal{N}, \mathcal{L})$	Directed graph of network topology
n_a	Network node in $\mathcal{N}$, either switch or end system
$[n_a, n_b]$	Data link in $\mathcal{L}$, from n_a to n_b
$[n_a, n_b].s$	Speed of link $[n_a, n_b] \in \mathcal{L}$
$[n_a, n_b].d$	Propagation delay on link $[n_a, n_b] \in \mathcal{L}$
$[n_a, n_b].mt$	Macrotick of link $[n_a, n_b] \in \mathcal{L}$
$\mathcal{S}$	Set of streams
s_i	Single stream in $\mathcal{S}$
$s_i.L$	Data size of stream $s_i \in \mathcal{S}$
$s_i.T$	Period of stream $s_i \in \mathcal{S}$
$s_i.\delta$	Initial transmission offset of stream $s_i \in \mathcal{S}$
$s_i^{[n_a,n_b]}$	Instance of stream $s_i \in \mathcal{S}$ on link $[n_a, n_b] \in \mathcal{L}$
HC	Hypercycle
T_{HC}	Duration of the time period of HC
$\mathcal{F}_i^{[n_a,n_b]}$	Set of frames of stream $s_i^{[n_a,n_b]} \in \mathcal{S}$ on link $[n_a, n_b] \in \mathcal{L}$
$f_i^{[n_a,n_b]}$	Single frame in $\mathcal{F}_i^{[n_a,n_b]}$
$f_i^{[n_a,n_b]}.T$	Period of frame i in $\mathcal{F}_i^{[n_a,n_b]}$ on link $[n_a, n_b] \in \mathcal{L}$
$f_i^{[n_a,n_b]}.L$	Transmission duration of frame i in $\mathcal{F}_i^{[n_a,n_b]}$ on link $[n_a, n_b] \in \mathcal{L}$
$f_i^{[n_a,n_b]}.\delta$	Initial transmission offset of frame i in $\mathcal{F}_i^{[n_a,n_b]}$ on link $[n_a, n_b] \in \mathcal{L}$
hc_i^j	Hypercycle of the stream s_i and s_j, with $s_i, s_j \in \mathcal{S}$
α	Translations of the frames of stream $s_i \in \mathcal{S}$ within the hyper cycle hc_i^j
β	Translations of the frames of stream $s_j \in \mathcal{S}$ within the hyper cycle hc_i^j
$r_{i,j}^{[n_a,n_b]}$	Binary indicator with $r_{i,j}^{[n_a,n_b]} = 1$ if s_i and $s_j \in \mathcal{S}$ are assigned to the same priority $s_i^{[n_a,n_b]}.p = s_j^{[n_a,n_b]}.p$ for $[n_a, n_b] \in \mathcal{L}$

TABLE 3. Optimization problem variables.

Notation	Description
$s_i^{[n_a,n_b]}.p$	Assigned queue for the instance of stream $s_i \in \mathcal{S}$ on link $[n_a, n_b] \in \mathcal{L}$
$f_{i,k}^{[n_a,n_b]}.\phi$	Transmission instant of a frame k in $\mathcal{F}_i^{[n_a,n_b]}$ on link $[n_a, n_b] \in \mathcal{L}$

The two sets of the optimization problem variables are reported in Table 3.

B. STATEMENT OF IEEE 802.1QBV TSN OPTIMIZATION PROBLEM

In order to obtain the scheduling decisions a Mixed-Integer Linear Programming (MILP) model for the scheduling optimization problem is considered and described in this section. The message optimization schedule is derived from [23] and [24]. However, it necessitates a refinement through the inclusion of an additional constraint specific to TSN-Lite-v1 and TSN-Lite-v2, a constraint not accounted for in the conventional problem formulation. This additional constraint is elucidated in the final section of this section. Furthermore, the problem presented is intricately tailored to address the unique requirements of the aerospace

application, specifically pertaining to a spacecraft launcher. Consequently, the various constraints have been meticulously transcribed to incorporate the characteristic parameters of this application. The scheduling in the case of Time Sensitive Networking is defined on the level of streams and not of individual frames like in TTEthernet or FTTE. The TAS scheduling relies on time and operates within a defined cycle, during which multiple frames of traffic are transmitted. If we want to achieve the same level of determinism as TTEthernet, we need to derive, in addition to the TTEthernet constraints, specific 802.1Qbv constraints. Considering behaviour on a per-frame basis is aimed at achieving the highest level of determinism. This is related to a necessary bandwidth overhead. The frame offsets represent the open and close events for the timed-gate of the assigned queue of the frame/stream. Frame offset and queue index directly translate into gate open and gate close events. The offset represents the gate open event for the specific queue index, while the gate close event is determined by the duration of the frame. Specifically, a gate can be in one of two states–opened or closed, respectively–allowing or disallowing the transmission of a frame from a certain queue. A specific scheduling technique is used to determine which frame of the queue will be dispatched within each queue with an open gate. Additionally, it is necessary to undertake a precise scheduling of the time instants at which to change the gate states. The constraints can be viewed as a set of inequality equations, where the variables are the message scheduling times that specify the proper temporal behavior for the communication streams.

Frame Constraint: given the periodic repeating pattern of critical streams, any frame belonging to a critical stream must be scheduled between the initial offset time and its period. Therefore, the stream time must encompass the whole transmission window. We have the condition in place to enforce this.

$$\begin{aligned}
&\forall\, s_i \in \mathcal{S}, \quad \forall\, [n_a, n_b] \in \mathcal{L}, \quad \forall\, f_{i,k}^{[n_a,n_b]} \in \mathcal{F}_i^{[n_a,n_b]} \\
&: f_{i,k}^{[n_a,n_b]}.\phi \geq f_{i,k}^{[n_a,n_b]}.\delta \\
&\wedge f_{i,k}^{[n_a,n_b]}.\phi \leq f_{i,k}^{[n_a,n_b]}.T + f_{i,k}^{[n_a,n_b]}.\delta - f_{i,k}^{[n_a,n_b]}.L \qquad (1)
\end{aligned}$$

Link Constraint: we enforce that no two frames that are routed through the same egress port of a device may overlap in the time domain because there can only be one frame at a time on a physical link.

$$\begin{aligned}
&\forall\, [n_a, n_b] \in \mathcal{L}, \quad \forall\, f_{i,k}^{[n_a,n_b]} \in \mathcal{F}_i^{[n_a,n_b]}, \\
&\forall\, f_{j,l}^{[n_a,n_b]} \in \mathcal{F}_j^{[n_a,n_b]} \quad with\ (i \neq j), \\
&\forall\, \alpha \in [s_i.\delta,\ s_i.\delta + \frac{hc_i^j}{s_i.T} - 1], \\
&\forall\, \beta \in [s_j.\delta,\ s_j.\delta + \frac{hc_i^j}{s_j.T} - 1] : \\
&(f_{i,k}^{[n_a,n_b]}.\phi + \alpha f_{i,k}^{[n_a,n_b]}.T \ \geq\ f_{j,l}^{[n_a,n_b]}.\phi \\
&\quad + \beta f_{j,l}^{[n_a,n_b]}.T + f_{j,l}^{[n_a,n_b]}.L) \\
&\vee \\
&(f_{j,l}^{[n_a,n_b]}.\phi + \beta f_{j,l}^{[n_a,n_b]}.T \ \geq\ f_{i,k}^{[n_a,n_b]}.\phi \\
&\quad + \alpha f_{i,k}^{[n_a,n_b]}.T + f_{i,k}^{[n_a,n_b]}.L) \qquad (2)
\end{aligned}$$

Frames of different streams with different periods, i.e. s_i and s_j, are repeated a different number of times within their hypercycle hc_i^j. The frames could potentially overlap in the time domain for any choice of α and β (the set of translations of frames of stream s_i and s_j). The link constraint (2) avoid these overlaps, expressing that the frame $f_{i,k}^{[n_a,n_b]}$ must finish before $f_{j,l}^{[n_a,n_b]}$ starts for each choice of α and β, or vice-versa.

Stream Transmission Constrain: the propagation of stream frames throughout the routed path must adhere to sequential order to guarantee minimum latency. This constraint ensures that a frame is forwarded only after it has been received by a specific device, even though it is not necessary for the schedule to be right. This constraint helps a scheduler to find solutions that reduce end-to-end latency. The network precision, indicated by δ_{sync}, is a crucial component in this case, and represents the maximum possible discrepancy between any two synchronized devices local clocks.

$$\begin{aligned}
&\forall\, s_i \in \mathcal{S}, \quad \forall\, [n_a, n_x], [n_x, n_b] \in \mathcal{L}, \\
&\forall\, f_{i,k}^{[n_a,n_x]} \in \mathcal{F}_i^{[n_a,n_x]}, \quad \forall\, f_{i,k}^{[n_x,n_b]} \in \mathcal{F}_i^{[n_x,n_b]} : \\
&(f_{i,k}^{[n_x,n_b]}.\phi * [n_x, n_b].mt - [n_a, n_x].d) - \delta_{sync} \\
&\geq ((f_{i,k}^{[n_a,n_x]}.\phi + f_{i,k}^{[n_a,n_x]}.L) * [n_a, n_x].mt) \qquad (3)
\end{aligned}$$

A switch cannot forward a frame until the full frame has been buffered in the switch. Consequently, the stream transmission constraint (3) imposes that a frame can only be scheduled on a subsequent link $[n_x, n_b]$ after the complete reception on the previous link $[n_a, n_x]$, considering the propagation delay of the respective link $[n_a, n_x].d$.

Finally, we need to consider an additional constraint. When a frame is scheduled to be transmitted on a link in a given time period, the corresponding gate will be open in that interval. Let us say that anything goes wrong, causing the frame to not be completely received or to not appear as the first frame in the queue as intended. The link then transmits the incorrect frame or doesn't transmit when it should. As a result, non-determinism is introduced, which compromises timeliness. Two streams that arrive from different devices can arrive at the same time in the same switch. The arrival order of frames during run-time can change, resulting in various queue states, due to a number of variables, such as a frame loss. As a result, it is possible that the individual frames order in the planned queue during run-time will not be predictable. As previously mentioned, the schedule in TSN does not determine the sequence of frames in the queue but the opening and closing of the timed gates on the queues of the egress port. We include conditions that ensure a predictable sequencing of frames in the queues in order to prevent this delay and jitter. This can be

enforced by preventing streams arriving at conflicting times from being queued together.

Determinism Constraint: It ensures that streams are run in the proper order; for example, if a frame from one stream has entered a queue, no frames from another stream may join the queue until the entire prior stream has been dispatched. In addition, there are only frames of one stream in the queue at a time, i.e., frames from another stream may only enter the queue if the already queued frames of the initial stream have been serviced. For this reason we add the following determinism constraint:

$$\begin{aligned}
&\forall\, [n_a, n_b] \in \mathcal{L}, \quad \forall\, s_i^{[n_a,n_b]}, s_j^{[n_a,n_b]} \in \mathcal{S}, \quad with\ (i \neq j),\\
&\quad \forall\, f_{i,k}^{[n_a,n_b]} \in \mathcal{F}_i^{[n_a,n_b]}, \quad \forall\, f_{j,l}^{[n_a,n_b]} \in \mathcal{F}_j^{[n_a,n_b]},\\
&\quad \forall\, \alpha \in [s_i.\delta,\ s_i.\delta + \frac{hc_i^j}{s_i.T} - 1],\\
&\quad \forall\, \beta \in [s_j.\delta,\ s_j.\delta + \frac{hc_i^j}{s_j.T} - 1] :\\
&\quad r_{i,j}^{[n_a,n_b]}(f_{j,l}^{[n_a,n_b]}.\phi * [n_a, n_b].mt + \beta * s_j.T + \delta_{sync})\\
&\quad \leq r_{i,j}^{[n_a,n_b]}(f_{i,k}^{[n_x,n_a]}.\phi * [n_x, n_a].mt + \alpha * s_i.T + [n_x, n_a].d)\\
&\quad \vee\, r_{i,j}^{[n_a,n_b]}(f_{i,k}^{[n_a,n_b]}.\phi * [n_a, n_b].mt + \beta * s_i.T + \delta_{sync})\\
&\quad \leq r_{i,j}^{[n_a,n_b]}(f_{j,l}^{[n_y,n_a]}.\phi * [n_y, n_a].mt + \alpha * s_j.T + [n_y, n_a].d)
\end{aligned} \tag{4}$$

To enforce determinism, (4) schedules carefully queue-sharing streams. It is necessary to schedule the streams of such flows in a manner where the frames of only one stream are present in the queue at a time. Therefore, (4) expresses the case where $f_{i,k}^{[n_a,n_b]}$ is scheduled to leave the queue in n_a before $f_{j,l}^{[n_a,n_b]}$ enters, and vice versa. Similar to the link congestion constraint, a queue can only contain frames from one stream at a time. In addition, the isolation of the streams is ensured in the event that the streams are put in different queues. Then, this constraint is always verified if $s_i^{[n_a,n_b]}.p \neq s_j^{[n_a,n_b]}.p$, which corresponds to having the binary indicator $r_{i,j}^{[n_a,n_b]}$ equal to 0. Consequently, the proposed modification to IEEE 802.1Q in TSN-Lite-v2, that enables the regeneration of a distinct priority for every differential flow at each receiving port, allows optimal resource management, analogous to the resource management observed in TSN employing IEEE 802.1Qci. In the case of TSN-Lite-v1, resource management will not be optimal as the following additional constraint needs to be taken into account. Therefore, the "PCP regeneration constraint" is exclusively applied to TSN-Lite-v1.

PCP Regeneration Constraint: if multiple flows pass through a receiving port, the same regenerated PCP must be assigned to them.

$$\begin{aligned}
&\forall\, [n_a, n_b] \in \mathcal{L}, \quad \forall\, s_i^{[n_a,n_b]}, s_j^{[n_a,n_b]} \in \mathcal{S}, \quad with\ (i \neq j),\\
&\quad s_i^{[n_a,n_b]}.p = s_j^{[n_a,n_b]}.p
\end{aligned} \tag{5}$$

Finally, as already pointed out, the frame offsets represent the open and close events for the timed-gate of the assigned queue of the frame/stream. A schedule and the collection of GCLs for all egress ports are equivalent. It is possible to create a set of GCLs from a schedule and vice versa. We use schedules rather than tables of GCLs for the rest of this paper. The interaction between various links and egress ports is visualized much more clearly in schedules.

The optimization problem aims to minimize the scheduling time for all frames. To achieve this, the objective function can be defined as the makespan MS of each EC, representing the total time required to transmit all frames within EC j. Let us denote with T_{EC} the duration of an EC, let $N_{HC} = \frac{T_{HC}}{T_{EC}}$ be the number of ECs in an HC that are numbered from 0 to N_{HC}-1:

$$\begin{aligned}
&min \sum_{j=0}^{N_{HC}-1} (MS)_j\\
&with \quad MS = \sum_{[n_a,n_b] \in \mathcal{L}} \ \sum_{k \in \mathcal{F}_i^{[n_a,n_b]}} \ \sum_{i \in \mathcal{S}_i^{[n_a,n_b]}} f_{i,k}^{[n_a,n_b]}.\phi
\end{aligned} \tag{6}$$

We now briefly discuss the implications of a number of special configurations which include non-scheduled traffic and different configurations of the scheduled and priority queues.

Configuration 1: network having one single queue per egress port operated as a scheduled queue. In this case, condition $s_i^{[n_a,n_b]}.p \neq s_j^{[n_a,n_b]}.p$ is always false and, hence, high-criticality flows are scheduled to be completely sequential in the time domain. This case amounts to serializing incoming traffic that converges on the same egress port reproducing a Time-Triggered Protocol. By using the additional Qbv constraint to achieve the same degree of determinism as TTEthernet, it is possible to obtain adequate time guarantees for critical traffic.

Configuration 2: network with all egress ports as many queues as there are incoming scheduled streams. Each stream could be assigned to its own dedicated queue. Hence, the problem is similar to scheduling TTEthernet, since $s_i^{[n_a,n_b]}.p \neq s_j^{[n_a,n_b]}.p$ is always true. In this case, each queue will behave as a "large" buffer, sufficient to accommodate all frames of each stream instance, guaranteeing isolation between flows through the schedule. The optimization scheduling problem presented was solved using the CPLEX solver, which is a high-level mathematical optimization software developed by IBM. CPLEX is capable of solving a wide range of optimization problems, including MILP problems, such as the one addressed in this context. The code is available on GitHub (https://github.com/tizianafiori/TSN-Lite-Paper-IEEE-Access).

VII. NUMERICAL RESULTS

In a real network and traffic scenario, we evaluated the performance of TSN-Lite-v2, and compared it to the performance of TSN-Lite-v1, TTEthernet and FTTE considering

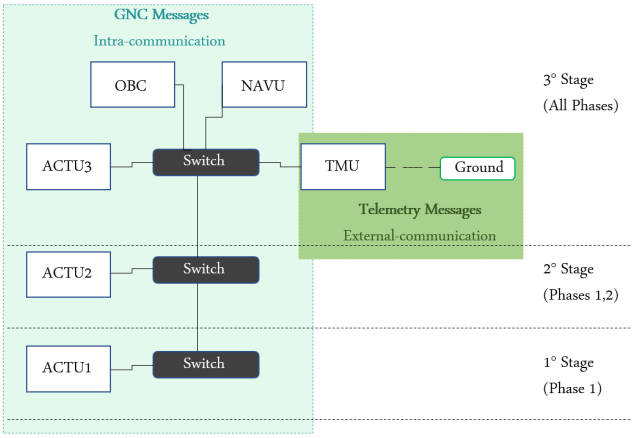


FIGURE 11. Network topology.

TABLE 4. Message flow parameters.

Stream_ID	Phase	Sender	Receiver	Periodicity [EC]	Offset[EC]
1	All	OBC	All	1	0
2	All	OBC	ACTU3	1	0
3	1,2	OBC	ACTU2	1	0
4	1	OBC	ACTU1	1	0
5	3	OBC	ACTU3	8	3
6	2	OBC	ACTU2	8	3
7	1	OBC	ACTU1	8	3
8	All	NAVU	OBC	4	1
9	All	ACTU3	OBC	8	5
10	1,2	ACTU2	OBC	8	5
11	1	ACTU1	OBC	8	5
12	All	OBC	ACTU3	1	0
13	1,2	OBC	ACTU2	1	0
14	1	OBC	ACTU1	1	0
15	All	OBC	ACTU3	1	0
16	All	OBC	ACTU3	1	0
17	All	OBC	ACTU3	1	0
18	1,2	OBC	ACTU2	1	0
19	1,2	OBC	ACTU2	1	0
20	1,2	OBC	ACTU2	1	0
21	1	OBC	ACTU1	1	0
22	1	OBC	ACTU1	1	0
23	1	OBC	ACTU1	1	0

a scheduling message optimization problem reported in [17] and [23] respectively. Subsection VII-A contains a description of the case study. In Subsection VII-B, the performances are compared.

A. CASE STUDY

Let us describe the topology and the message set used.

The topology is reported in Fig. 11. A space launch vehicle with three stages (1° Stage, 2° Stage, 3° Stage) that gradually separate after lift-off is used to evaluate the performances of the three alternatives. As a result, three flight phases with different network architectures may be recognized. Not all stages are attached to the launcher for the duration of the flight since the launch vehicle separates stages while in flight. Let us focus on three distinct flight phases in particular:

- Flight Phase 1 (FP1), when all three stages are part of the launch vehicle, implying the maximum number of messages involved.
- Flight Phase 2 (FP2), when only two stages (2° Stage and 3° Stage) are still part of the launch vehicle.
- Flight Phase 3 (FP3), when launcher only consists of 3° Stage.

The topology consists of three stages with one switch in each of them. A NAVU is attached to the switch of the upper stage as well as an OBC, the TMU, and an ACTU (ACTU3 for the upper stage). In the other remaining stages, one ACTU (ACTU2 and ACTU1 for the second and third stages respectively) is attached. The messages sent to the TMU only, can be routed only using the telemetry links through the switches.

The message set used is drawn from the traffic exchanged within the VEGA launch vehicle [28]. The MIL-STD-1553B-compliant original messages are grouped in the message set used for simulation in order to reduce overhead from the Ethernet network technology proposed in this study. The EC lasts for 5 *ms*. The periodicity of the message streams are equal to 5 *ms*, 20 *ms*, and 40 *ms*, resulting in a 40 *ms* HC duration. Thus, one HC has eight ECs. Information of the message streams are reported in Table 4, providing the following data: i) an identifier (Stream_ID), ii) the phases of activity for message flows; iii) the sender and the receiver; iv) the periodicity and the offset stated in terms of number of ECs. 23 periodic message flows that repeat every HC have been considered. The MAC SDU length in Figs 4 and 5 is chosen to be equal to 40 bytes. The flows in Table 4 represent a real-world network traffic, in the context of the VEGA launcher (the European Space Agency launch vehicle). The parameters have been thoughtfully selected to align with the phase, periodicity, and initial offset observed in the actual scenario. In this application, the aim is to transmit periodic message flows within the ECs according to their specified offset and periodicity. The critical requirement is to ensure that messages are transmitted within the specified 5 *ms* time-frame of each relevant EC. For example, let us consider sensor data from NAVU that needs to be sent to the OBC for processing. Suppose this data must be transmitted in EC #3. It is crucial that the entire transmission occurs within the 5 *ms* time-frame of EC #3 because the OBC will process the sensor data in the subsequent elementary cycle. Minimizing the makespan is equivalent to reducing latency for each EC. To evaluate the impact of latency, the average delay for all transmitted message flows is assessed for each EC, ensuring that each message flow effectively adheres to the 5 *ms* boundary of the relevant ECs, as detailed in section VII-B4. Additionally, to assess the overall performance in terms of its adaptability and effectiveness, it is necessary to vary the parameters and number of the message flows. We evaluated the performance of both TSN-Lite-v1 and TSN-Lite-v2 within scenarios featuring diverse message flows. These scenarios considered both double and triple the number of message flows as listed in Table 4. The periodicity was uniformly varied, encompassing values of 1, 2, 4, and 8, (mirroring the expected values for the real VEGA network messages), along with offset values ranging from 0 to 7. Additionally, we varied the length of flow messages, ranging

from 88 bytes (the minimum Ethernet frame length) to double or triple this length for duplicated and triplicated messages, respectively. This comprehensive testing allowed for a thorough assessment of performance. The results are presented in VII-B2.

B. RESULTS

The performance index is the makespan, defined as the length of time between the beginning of the EC and the end of the scheduled frames transmission. First, we provide a makespan performance comparison of TSN-Lite-v2 with TTEthernet and FTTE, followed by a performance comparison of TSN-Lite-v2 with TSN-Lite-v1 and TSN-Target, considering variations of the parameters and number of messages. Additionally, the makespan performance of TSN-Lite-v1 and TSN-Lite-v2 are evaluated while varying the network precision to assess the effects of clock misalignment in synchronized devices. Finally, the average delay is evaluated for TSN-Lite-v2, TSN-Lite-v1, and TTEthernet.

1) STATE-OF-THE-ART RESULTS COMPARISON

For the comparison with the state-of-the-art, the makespan has been evaluated for each EC, considering the message set of Tab. 4, in the following scenarios:

- TTEthernet: applying the optimization problem described in [31], with the same case-study setup, with the use of TTEthernet switches which can schedule the forwarding points in time per frame.
- TSN-Lite-v2: applying the optimization problem described in VI-B with eight available queues per link and with the use of TSN switches which can schedule the forwarding points in time per stream.
- FTTE: applying the optimization problem described in [31], with the same case-study setup. The frame scheduling instants are determined in the end systems only. Store & Forward (SF) and Cut Through (CT) forwarding solutions of traditional switches are considered; where, in the SF solution, retransmission of a message on a port is only permitted once the switch has completely received the message; instead, in the CT solution, the message is retransmitted as soon as the destination address is detected.

The makespan is provided as a function of the EC index for the Flight Phases 1, 2 and 3 in Fig. 12, leading to the following remarks:

- The highest makespan values are achieved in Flight Phase 1 for all three scenarios when all stages are attached, needing the scheduling of the highest number of messages.
- In all flight phases, the makespan value in the TTEthernet and TSN-Lite-v2 cases is lower that the one in the FTTE case. The largest percentage increase in Store and Forward FTTE is 166.7%, 125% and 83.3% in FP1, FP2 and FP3, respectively. This result is not unexpected due to the use of TTEthernet and TSN switches, which are non-traditional and that enable the rescheduling of messages in the switches with the introduction of an offset time. Of course, higher TTEthernet/TSN network performance comes at a higher cost due to a more complex switching technology.
- CT switching technology allows for a reduction of the performance gap between FTTE and TSN/TTEthernet, especially in the Flight Phase 3 in which few messages are scheduled and the delay is reduced because the network is simply composed by the switch of the 3rd stage only. In this scenario the maximum percentage makespan reduction in TSN/TTEthernet with respect to FTTEthernet is 33%, 25% and 17% in FP1, FP2 and FP3, respectively.
- The use of IEEE802.1Qbv time aware shaping allows TSN-Lite-v2 to achieve the same degree of determinism as TTEthernet in FP2 and FP3; a light degradation of 6 μs only occurs in FP1.

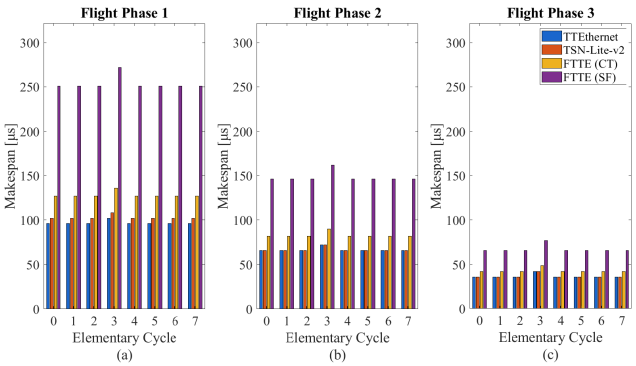


FIGURE 12. Comparison of TTEthernet, TSN-Lite-v2, and FTTE in terms of the makespan; the results are reported for the Flight Phases 1 (a), 2 (b), and 3 (c).

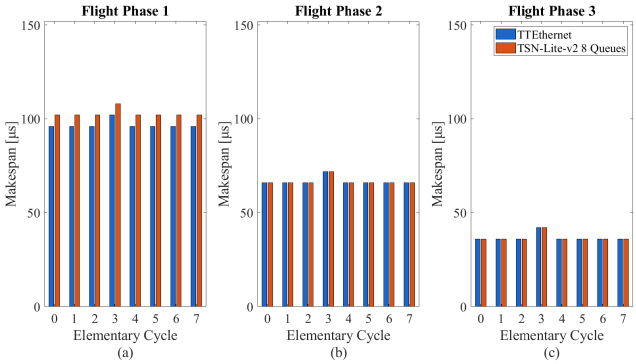


FIGURE 13. Makespan values comparison for TTEthernet and TSN-Lite-v2 as a function of the EC index in the Flight Phases 1 (a), 2 (b), and 3 (c). TSN-Lite-v2 case with eight queues available per egress port.

In addition, we carry out a comprehensive comparison between TSN-Lite-v2 and TTEthernet, taking into account scenarios with one and eight queues per egress port for TSN-Lite-v2.

TTEthernet and TSN-Lite-v2 are compared in Figs 13 and 14 for eight and one queues available per

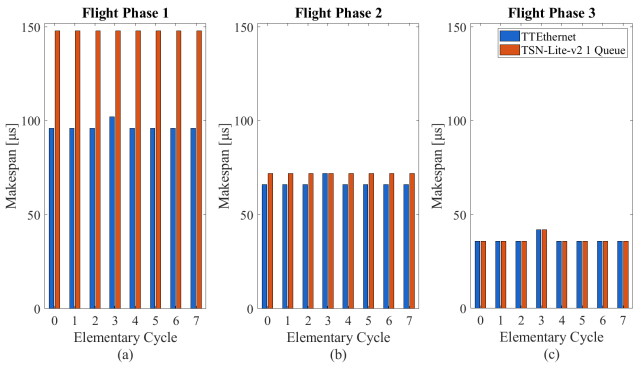


FIGURE 14. Makespan values comparison for TTEthernet and TSN-Lite-v2 as a function of the EC index in the Flight Phases 1 (a), 2 (b) and 3 (c). TSN-Lite-v2 case with one queue available per egress port.

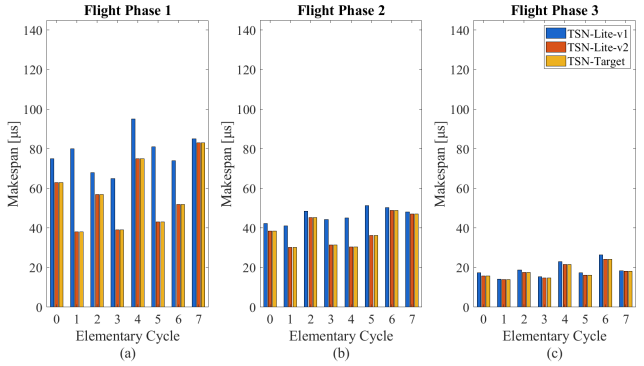


FIGURE 15. Average makespan values obtained by applying the respective optimization problem for TSN-Lite-Target TSN-Lite-v1 and TSN-Lite-v2, considering double traffic and uniform variations in periodicity, offset, and message flow length in the Flight Phases 1 (a), 2 (b), and 3 (c).

egress port respectively. The results for TSN-Lite-v2 with eight available queues were already included in Fig. 12. However, it was decided to present them separately in Fig. 13 for the sake of clarity in presentation, without adding the FTTE. providing the following remarks:

- As already pointed out, the makespan values in TSN-Lite-v2 with eight queues are equal to TTEthrnet in Flight Phases 2 and 3. TTEthernet achieves better performances only in Flight Phase 1 with a maximum difference of only 6 μs.
- The TSN-Lite-v2 makespan values in the worst possible case of one only queue available are equivalent to the TTEthernet case in Flight Phase 3, and there is a maximum percentage makespan difference of 4% and 20% in Flight Phase 2 and 1, respectively. Even in the case in which TT traffic uses only one queue in TSN-Lite-v2, it is possible to obtain adequate time guarantees for TT traffic.

In the context of a launcher communication system, the configuration with a single queue dedicated to critical traffic is an extreme and highly unlikely scenario. Nevertheless, this extreme case has been considered, demonstrating that even under the worst-case conditions, TSN-Lite-v2 provides performance that meets the requirements of the aerospace application under consideration.

2) TSN-BASED RESULTS COMPARISON WITH VARIED MESSAGE FLOWS

The makespan was evaluated by first considering double and then triple the number of messages listed in Table 4. During these assessments, we concurrently varied the periodicity, offset, and message length. For each EC, the average makespan was calculated for TSN-Lite-v1, TSN-Lite-v2, and TSN-Target. This allows for the assessment of performance variations in response to changes in network parameters and the number of message flows, thereby evaluating the flexibility and adaptability in terms of generalization. In this analysis we considered:

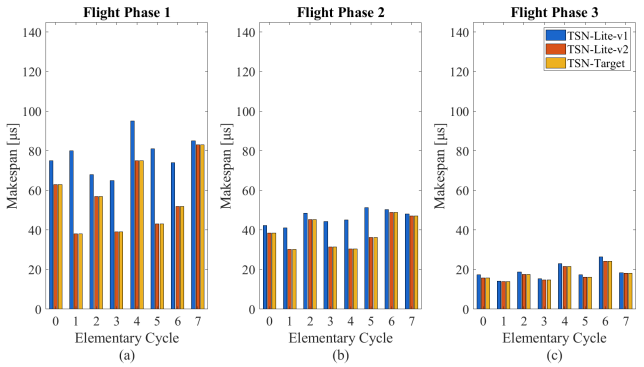


FIGURE 16. Average makespan values obtained by applying the respective optimization problem for TSN-Lite-Target TSN-Lite-v1 and TSN-Lite-v2, considering triple traffic and uniform variations in periodicity, offset, and message flow length in the Flight Phases 1 (a), 2 (b), and 3 (c).

- TSN-Target: incorporating the standards IEEE802.1Qci and IEEE802.1Qbv standard for filter frames and time aware shaping respectively [23].
- TSN-Lite-v1: incorporating the standard IEEE802.1Qbv only for time aware shaping and a static priority code mechanism for each flow.
- TSN-Lite-v2: incorporating the standard IEEE802.1Qbv for time aware shaping and a dynamic priority code mechanism for each flow described in Section V.

The comparison is depicted in Fig. 15 for the double traffic case and in Fig. 16 for the triple traffic case. We report the average of the makespan for each EC for the three flight phases. Let us noting that:

- TSN-Lite-v2 achieves the same makespan performance as TSN-Target for all flight phases, while offering the advantage of being simpler in terms of configuration and implementation, to avoid the use of the standard IEEE802.1Qci.
- TSN-Lite-v2 allows for better performance than TSN-Lite-v1, particularly in Flight Phase 1 where the message count is higher. The use of dynamic priority values in TSN-Lite-v2 enables the allocation of flows to distinct

priority queues for each link, maximizing the efficiency of network resource allocation. This capability is not available with a static priority assignment in TSN-Lite-v1, which is why it yields inferior performance. Specifically, in the case of double traffic, the maximum percentage difference is 52.50%, 26.54%, and 8.67%, corresponding to FP1, FP2, and FP3. Similarly, in the case of triple traffic, the maximum percentage difference is 58.06%, 24.40%, and 4.97%, corresponding to the three flight phases.

- By increasing the number of frame messages and varying the parameters of periodicity, offset, and frame message length, we observe enhanced flexibility and bandwidth efficiency of TSN-Lite-v2.

3) NETWORK PRECISION VARIATION RESULTS

Lastly, we assessed the performance of TSN-Lite-v1 and TSN-Lite-v2 by varying network clock precision. For both, we initially evaluated the maximum makespan achieved in the ECs under perfect synchronization conditions, using the flow set from Tab. 4. We then proceeded to evaluate the makespan in the same scenario while varying network precision from 1 ns to 10 μs, enabling us to compare the results with those obtained under perfect synchronization conditions.

Fig. 17 illustrates, for each chosen network precision value (δ_{sync}), the disparity between the maximum makespan obtained with clock misalignment and the maximum makespan achieved under perfect synchronization. This assessment of makespan difference was conducted across all three flight phases and for TSN-Lite-v1 and TSN-Lite-v2. The results depicted in Fig. 17 are presented on a logarithmic scale for both axes. From these results, we can make the following observations:

- As expected, an increase in the maximum misalignment of device clocks, and consequently network precision, is associated with a corresponding rise in total reserved bandwidth. This increase in makespan becomes more pronounced as the misalignment grows. For example, with a network precision on the order of nanoseconds, the makespan differs by only a few nanoseconds from that of perfect synchronization. However, when considering a network precision of 10 μs, which is of a similar order of magnitude as the total makespan obtained in FP2 and FP3, the difference in makespan becomes more significant. It is important to note that typical network precision values are on the order of nanoseconds.
- TSN-Lite-v2 exhibits improved performance as clock misalignment increases. In the worst-case scenario, with δ_{sync} equal to 10 μs, TSN-Lite-v1 implies a percentage degradation of 67%, 104%, and 94% compared to TSN-Lite-v2 in FP1, FP2, and FP3, respectively.

TSN-Lite-v2 allows for superior performance, however, depending on the traffic and network characteristics, TSN-Lite-v1 may exhibit performance that is adequate to fulfill the application requirements. In this scenario, choosing the static PCP assignment approach, with its simpler configuration, is a feasible option. Furthermore, in cases where the topology and message set make the scheduling optimization problem infeasible for TSN-Lite-v2, the second option is TSN-Lite-v1.

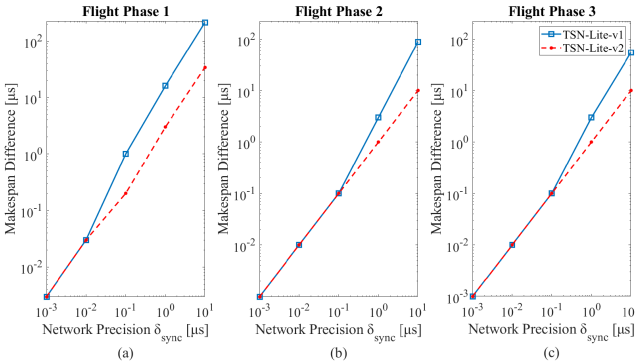


FIGURE 17. Makespan difference values obtained by subtracting the makespan achieved with synchronization misalignment from that achieved with perfect synchronization for TSN-Lite-v2 and TSN-Lite-v1, in the Flight Phases 1 (a), 2 (b), and 3 (c).

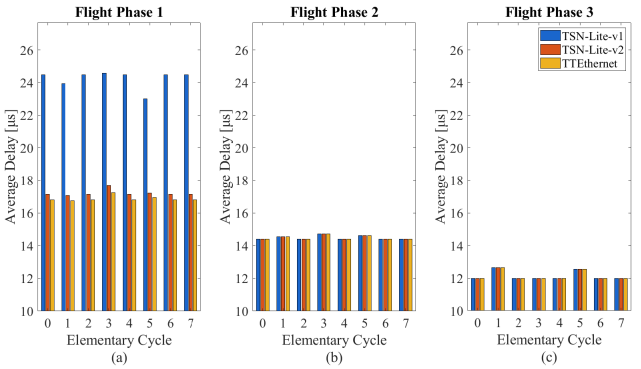


FIGURE 18. Comparison of average delay values for TSN-Lite-v1, TSN-Lite-v2, and TTEthernet, in the Flight Phases 1 (a), 2 (b), and 3 (c).

4) DELAY RESULTS

The average delay has been evaluated for each EC, considering the message set of Table 4, in the following scenarios:

- TSN-Lite-v1: incorporating the standard IEEE802.1Qbv only for time aware shaping and a static priority code mechanism for each flow, as described in V-A.
- TSN-Lite-v2: incorporating the standard IEEE802.1Qbv for time aware shaping and a dynamic priority code mechanism for each flow, as described in V-B.
- TTEthernet: with the use of TTEthernet switches which can schedule the forwarding points in time per frame, as described in [31].

In Fig. 18, the results are presented. The average delay value was evaluated by considering the mean of all values obtained for the messages to be transferred within an EC. We can observe the following key findings:

- TSN-Lite-v2 enables to achieve an average delay value for each EC equal to TTEthernet in FP2 and FP3. During

FP1, there is only a maximum marginal delay difference of 2.5%.

- TSN-Lite-v2 consistently delivers significantly improved average delay values compared to TSN-Lite-v1 in FP1, where a higher number of message flows are present. This performance advantage becomes more pronounced as the message load increases. In particular, the maximum percentage difference is 42.6%.
- In all scenarios, the average delay meets the requirements of the specific aerospace application, ensuring that all messages within a certain EC are transmitted within the 5 *ms* limit. However, as the number of messages increases, TSN-Lite-v2 outperforms TSN-Lite-v1, offering superior performance.

VIII. CONCLUSION

FTTE is undoubtedly characterized by lower costs than the TTEthernet and TSN. This is the consequence of the use of traditional Ethernet switches that are now available at higher speeds and low costs. However, no market products are today available implementing the FTTE protocols. On the other hand, TTEthernet is mature, and its implementation is a reality. Neverthless, it is characterized by a lack of adaptability when it comes to adding new features to its end systems and switches and, more crucially, it is linked to extremely high costs because characterized by mono-vendor market. Finally, the definition of the TSN standard by a highly prestigious body (IEEE) and its application in not only the aerospace field but also automation and vehicle sectors has led to a multi-vendor market that certainly will allow for significantly lower costs than those of TTEthernet. Therefore, TSN-Lite-v2 has been defined and based on a dynamic priority code. TSN-Lite-v2 offers straightforward configuration and implementation, yet it delivers outstanding performance while maintaining the same bandwidth efficiency as the TSN complemented by the IEEE802.1Qci standard. Finally, the performance comparison with the state-of-the-art demonstrates that TSN-Lite-v2 achieves equivalent performance to the well-established TTEthernet in the aerospace application under consideration.

REFERENCES

[1] M. Haverty, "MIL-STD 1553—A standard for data communications," *Commun. Broadcast.*, vol. 10, pp. 29–33, Jan. 1986.

[2] O. Stan, A. Cohen, Y. Elovici, and A. Shabtai, "Intrusion detection system for the MIL-STD-1553 communication bus," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 56, no. 4, pp. 3010–3027, Aug. 2020.

[3] R. Polonowski and R. Clavier, "Ariane launchers digital engineering: Stakes and challenges," in *Proc. 8th Medit. Conf. Embedded Comput. (MECO)*, Jun. 2019, pp. 1–5, doi: 10.1109/MECO.2019.8760090.

[4] TTTech. (2023). *TTTech Aerospace on Board for NASA's Artemis*. [Online]. Available: https://www.tttech.com/tttech-on-board-nasa-artemis-i

[5] M. Tugnoli, M. Sarret, M. Aliberti, M. Tugnoli, M. Sarret, and M. Aliberti, "Overview on micro launchers," *Eur. Access Space, Bus. Policy Perspect. Micro Launchers*, vol. 1, pp. 5–28, Jan. 2019.

[6] V. Eramo, F. G. Lavacca, M. Listanti, and S. Caporossi, "Performance evaluation of TTEthernet-based architectures for the VEGA launcher," in *Proc. IEEE Aerosp. Conf.*, Mar. 2018, pp. 1–6.

[7] V. Eramo, F. Valente, F. G. Lavacca, T. Fiori, V. Papandrea, M. Albano, and S. Ciabuschi, "Flexible time triggered Ethernet: A cost efficient COTS-based technology for the development of launcher networks," in *Proc. IEEE 9th Int. Workshop Metrology Aerosp. (MetroAeroSpace)*, Jun. 2022, pp. 97–102.

[8] V. Eramo, F. Valente, F. G. Lavacca, T. Fiori, V. Papandrea, M. Albano, S. Ciabuschi, and E. Cavallini, "Extension of the FTT-Ethernet architecture for the support of telemetry messages in launcher networks," in *Proc. AEIT Int. Annu. Conf. (AEIT)*, Oct. 2022, pp. 1–6.

[9] L. Lo Bello and W. Steiner, "A perspective on IEEE time-sensitive networking for industrial communication and automation systems," *Proc. IEEE*, vol. 107, no. 6, pp. 1094–1120, Jun. 2019.

[10] H. Wang, Z. Zhao, and J. Wei, "Adaptive scheduling algorithm based on time aware shaper," in *Proc. 5th Int. Conf. Adv. Electron. Mater., Comput. Softw. Eng. (AEMCSE)*, Apr. 2022, pp. 555–562.

[11] T. Fedullo, A. Morato, F. Tramarin, L. Rovati, and S. Vitturi, "A comprehensive review on time sensitive networks with a special focus on its applicability to industrial smart and distributed measurement systems," *Sensors*, vol. 22, no. 4, p. 1638, Feb. 2022.

[12] TTTech. *Time-Triggered Ethernet*. Accessed: Jul. 3, 2023. [Online]. Available: https://www.tttech.com/technologies/time-triggered-ethernet/

[13] R. Li, B. Fu, G. Xie, F. Peng, and R. Li, "Efficient holistic timing analysis with low pessimism for rate-constrained traffic in TTEthernet," *J. Syst. Archit.*, vol. 134, Jan. 2023, Art. no. 102785.

[14] R. Zhao, G. Qin, J. Yan, and J. Qin, "Schedule optimization for TTEthernet-based time-triggered automotive systems," *Int. J. Automot. Technol.*, vol. 21, no. 6, pp. 1483–1494, Dec. 2020.

[15] H. Geppert, F. Dürr, S. Bhowmik, and K. Rothermel, "Just a second—Scheduling thousands of time-triggered streams in large-scale networks," 2023, *arXiv:2306.07710*.

[16] V. Eramo, F. Lavacca, F. Valente, A. Pisculli, and S. Caporossi, "Simulation and experimental evaluation of a flexible time triggered Ethernet architecture applied in satellite nano/micro launchers," *Aerospace*, vol. 5, no. 3, p. 84, Aug. 2018.

[17] V. Eramo, T. Fiori, F. G. Lavacca, F. Valente, A. Baiocchi, S. Ciabuschi, M. Albano, and E. Cavallini, "A max plus algebra based scheduling algorithm for supporting time triggered services in Ethernet networks," *Comput. Commun.*, vol. 198, pp. 85–97, Jan. 2023.

[18] O. Hotescu and A. Finzi, "Scheduling rate constrained traffic in end systems of time-aware networks," in *Proc. 26th IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2021, pp. 1–8.

[19] Y. Seol, D. Hyeon, J. Min, M. Kim, and J. Paek, "Timely survey of time-sensitive networking: Past and future directions," *IEEE Access*, vol. 9, pp. 142506–142527, 2021.

[20] L. Lo Bello, G. Patti, and L. Leonardi, "A perspective on Ethernet in automotive communications—Current status and future trends," *Appl. Sci.*, vol. 13, no. 3, p. 1278, Jan. 2023.

[21] T. Stüber, L. Osswald, S. Lindner, and M. Menth, "A survey of scheduling algorithms for the time-aware shaper in time-sensitive networking (TSN)," *IEEE Access*, vol. 11, pp. 61192–61233, 2023.

[22] A. Nasrallah, A. S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. Elbakoury, "Performance comparison of IEEE 802.1 TSN time aware shaper (TAS) and asynchronous traffic shaper (ATS)," *IEEE Access*, vol. 7, pp. 44165–44181, 2019.

[23] S. S. Craciunas and R. S. Oliver, "An overview of scheduling mechanisms for time-sensitive networks," in *Proc. Real-time Summer School LÉcole dÉté Temps Réel (ETR)*, 2017, pp. 1551–3203.

[24] S. Craciunas, R. Serna Oliver, M. Chmelík, and W. Steiner, "Scheduling real-time communication in IEEE 802.1QBV time sensitive networks," in *Proc. 24th Int. Conf. Real-Time Netw. Syst.*, 2016, pp. 183–192.

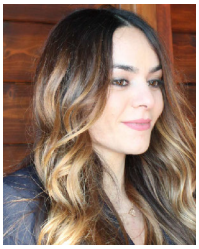
[25] A. A. Syed, S. Ayaz, T. Leinmüller, and M. Chandra, "MIP-based joint scheduling and routing with load balancing for TSN based in-vehicle networks," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, Dec. 2020, pp. 1–7.

[26] D. Hellmanns, L. Haug, M. Hildebrand, F. Dürr, S. Kehrer, and R. Hummen, "How to optimize joint routing and scheduling models for TSN using integer linear programming," in *Proc. 29th Int. Conf. Real-Time Netw. Syst.*, Apr. 2021, pp. 100–111.

[27] M. Weitbrecht, "ILP-based schedule planning for dynamic IEEE time-sensitive networks," M.S. thesis, Inst. Parallel Distrib. Syst., Univ. Stuttgart, Stuttgart, Germany, 2022.

[28] V. Eramo, F. G. Lavacca, M. Listanti, and S. Caporossi, "Definition and performance evaluation of an advanced avionic TTEthernet architecture for the support of launcher networks," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 33, no. 9, pp. 30–43, Sep. 2018.

[29] J. C. Eidson, *Measurement, Control, and Communication Using IEEE 1588*. Cham, Switzerland: Springer, 2006.
[30] IEEE802.1Q-2022. (2022). *IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks*. [Online]. Available: https://standards.ieee.org/about/get/802
[31] V. Eramo, T. Fiori, F. G. Lavacca, F. Valente, M. Albano, S. Ciabuschi, and E. Cavallini, "Performance comparisons of flexible time triggered Ethernet and TTEthernet technologies for space launcher networks," in *Proc. IEEE 10th Int. Workshop Metrol. Aerosp. (MetroAeroSpace)*, Jun. 2023, pp. 61–66.



TIZIANA FIORI (Graduate Student Member, IEEE) received the B.S. degree in aerospace engineering and the M.S. degree in space and astronautical engineering from the Sapienza University of Rome, Italy, in 2018 and 2021, respectively, where she is currently pursuing the Ph.D. degree in information and communication technologies (ICT) with the Department of Information Engineering, Electronics and Telecommunications. In 2021, she received a Junior Fellowship for the research activity of "Study of real-time ethernet technologies for TLC networks of space launchers," and a Temporary Fellowship Researcher for the project NIBBIO "Development of an experimental real time ethernet test bed," in collaboration with the Italian Space Agency (ASI) from the Sapienza University of Rome. Her research interests include real-time ethernet technologies, space launcher networks, and message scheduling.

FRANCESCO GIACINTO LAVACCA received the Laurea (M.Sc.) degree (cum laude) in electronic engineering and the Ph.D. degree in information technology from the Sapienza University of Rome, Italy, in 2013 and 2017, respectively. In 2018, he joined the TLC Group, Fondazione Ugo Bordoni, as a Researcher. He is currently a Researcher with the DIET Department, Sapienza, where he started his postdoctoral research activities. In 2016, he was a Visiting Researcher with the College of Computing, Georgia Institute of Technology, Atlanta, GA, USA. Furthermore, he was involved in the framework of national and international projects, like Advanced Avionic Architecture (AAA) and Nano Micro Launch Vehicle (NMLV) with Italian Space Agency (ASI) and European Space Agency (ESA), respectively. His current research interests include all-optical networks and switching architectures, 5G networks planning and design, network function virtualization, and time-triggered and deterministic ethernet.

FRANCESCO VALENTE (Graduate Student Member, IEEE) received the bachelor's degree (cum laude) in aerospace engineering and the master's degree (cum laude) in space and astronautical engineering from the Sapienza University of Rome, in 2018 and 2020, respectively, where he is currently pursuing the Ph.D. degree in information and communication technologies with the Department of Information Engineering, Electronics, and Telecommunications. His research has involved collaboration on projects with the Italian Space Agency (ASI) and the aerospace industry. His current interests include orbital edge computing, artificial intelligence applied to network function virtualization, and time-sensitive networking, including deterministic ethernet.

VINCENZO ERAMO (Member, IEEE) received the Laurea degree in electronics engineering and the Dottorato di Ricerca (Ph.D.) degree in information and communications engineering from the University of Roma La Sapienza, in 1995 and 2001, respectively. From June 1996 to December 1996, he was a Researcher with Scuola Superiore Reiss Romoli. In 1997, he became a Researcher with Fondazione Ugo Bordoni, Telecommunication Network Planning Group. From November 2002 to October 2005, he was an Assistant Professor. From November 2006 to June 2010, he was an Aggregate Professor with the INFOCOM Department, Sapienza University of Rome. He is currently an Associate Professor with the Department of Engineering of Information, Electronics, and Telecommunications. His research activities involve national and international projects, notably as the Scientific Coordinator with the Sapienza University of Rome in two Networks of Excellence, E-PhotoONe+, and BONE, funded by the European Commissions (FP6 and FP7), from 2006 to 2007 and from 2008 to 2011, respectively. These projects focused on studying optical networks. He served as an Associate Editor for multiple IEEE journals, including IEEE TRANSACTIONS ON COMPUTER (2011–2015), IEEE COMMUNICATION LETTERS, since 2014, IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING, since 2016, and IEEE OPEN JOURNAL OF THE COMMUNICATIONS SOCIETY, since 2019. Additionally, he is an Associate Editor of *Optical Fiber Technology* (Elsevier), since 2020, and *ACM Computing Surveys*, since 2021. He was the Winner of the Exemplary Editor Award 2016 and 2017 of IEEE COMMUNICATIONS LETTERS and the Exemplary Editor Awards 2021 of IEEE OPEN JOURNAL OF THE COMMUNICATIONS SOCIETY.

Open Access funding provided by 'Università degli Studi di Roma "La Sapienza" 2' within the CRUI CARE Agreement