aerospace

MDPI

Article

Simulation and Experimental Evaluation of a Flexible Time Triggered Ethernet Architecture Applied in Satellite Nano/Micro Launchers

Vincenzo Eramo [1,*], Francesco G. Lavacca [1], Francesco Valente [1] and Andrea Pisculli [2] and Stefano Caporossi [2]

[1] DIET, Sapienza University of Rome, 00184 Rome, Italy; francescogiacinto.lavacca@uniroma1.it (F.G.L.); francvalente@gmail.com (F.V.)

[2] SPACELAB, Via degli Esplosivi 1, 00134 Colleferro, Italy; Andrea.pisculli@ortec-it.it (A.P.); stefano.Caporossi@elv.it (S.C.)

* Correspondence: Vincenzo.Eramo@uniroma1.it; Tel.: +39-06-44585372

Received: 13 July 2018; Accepted: 7 August 2018; Published: 9 August 2018

Abstract: The success of small satellites has lead to the study of new technologies for the realization of Nano and Micro Launch Vehicle (NMLV) in order to make competitive launch costs. The paper has the objective to define and experimentally investigate the performance of a communication system for NMLV interconnecting the End Systems as On-Board Computer (OBC), telemetry apparatus, Navigation Unit...we propose a low cost Ethernet-based solution able to provide the devices with high interconnection bandwidth. To guarantee hard delays to the Guide, Navigation and Control applications we propose some architectural changes of the traditional Ethernet network with the introduction of a layer implemented in the End Systems and allow for the lack of any contention on the network links. We show how the proposed solution has comparable performance to the one of TTEthernet standard that is a very expensive solution. An experimental test-bed equipped with Ethernet switches and Hercules boards by Texas Instruments is also provided to prove the feasibility of the proposed solution.

Keywords: nano and micro launch vehicle; flexible time triggered Ethernet; TTEthernet; experimental test-bed

1. Introduction

The miniaturization of electronics, together with reliability and performance increase as well as reduction of cost, have allowed the use of Commercials-Off-The-Shelf (COTS) hardware/software in the space industry, fostering the use of a small satellite. An analysis of current and future launch vehicles reveals that we are currently in a phase of transition, where old launch vehicles get retired and new ones enter the market. However, the satellite launch vehicle business has been established to carry payloads of thousands of kilos into low Earth orbit and has not adjusted itself to the market of small satellites. As a result, there is only one launch vehicle for dedicated small satellite launches commercially available, but it carries a high price tag. In fact, the only commercially available Micro Launch Vehicle appropriate for dedicated Mini and Micro satellites is Pegasus with a payload capacity of about 250 kg. It carries a price tag of USD 56.3 million [1].

Several small low cost launch vehicles are under development. Since these small launch vehicles have similar complexity as huge launch vehicles, high development costs are intrinsic, leading to a high specific price (USD/kg payload) [1].

This paper is focused on the communication system of a launch vehicle interconnecting On-Board Computer (OBC), Telemetry apparatus, Navigation Unit. To get competitive launch prices, solutions have to be identified at lower costs than the ones for traditional launchers. The objective of the paper is to propose and experimentally investigate a low cost communication system for a Nano and Micro Launch Vehicle (NMLV) based on Ethernet technology. Actually the network used in small launchers is based on a 1553B [2] bus technology that has shown many limits in recent years. In Vettore Europeo di Generazione Avanzata (VEGA), small launcher jointly designed by Italian Space Agency and European Space Agency [3], the 1 Mb/s bandwidth of the 1553B bus is now saturated and this prevents it from transporting additional telemetry messages.

One of the main factors of attractions to Ethernet is due to its high bandwidth, much higher than in other communication networks. Another one is that the prices of Commercial-Of-The-Shelf (COTS) standard Ethernet components are relatively low [4–6]. Since standard Ethernet was not originally designed to be capable of providing temporal guarantees for real-time communication, there have been many protocol designs adapting standard Ethernet to be capable of providing such temporal guarantees. Some examples are: EtherCAT, Ethernet/IP, PROFINET [7], Avionic Full Duplex Ethernet (AFDX) [8] and TTEthernet [9–12]. AFDX and TTEthernet are considered as mature solutions but their high cost suggests to study alternative solutions. The proposed communication system will be based on the following architectural choices:

- use of Common-Off-The-Shelf (COTS) Ethernet technology as much as possible with the addition of the minimum number of functional components for the support of real-time traffic;
- use of traditional Ethernet switches, full duplex transmission and simple network topology in order to reduce the number of both nodes and ports;
- implementation of a real-time layer bypassing the Transmission Control Protocol/User Datagram Protocol/Internet Protocol (TCP/UDP/IP) stack to avoid potential delays introduced by this protocols;
- definition of a layer for the scheduling of messages so as to guarantee the Quality of Service (QoS) requested by the Guide, Navigation and Control and telemetry applications.

We consider a solution in which there are only modifications on top of medium access control (MAC) layer, in order to move the intelligence towards the end systems and to maintain the simplicity of the switching architecture. The solution is based on the Flexible Time Triggered Ethernet (FTTE) [13,14] that fulfills a set of specific requirements, such as predictability, support for periodic traffic with different periods, support for sporadic traffic and bounded latency.

The main research objectives are:

- to define a low cost FTTE-based architecture for the communication system of a NMLV; we describe in detail all of the aspects as functionalities, message and frame structures;
- we extend the solution proposed in [13] to the case of networks equipped with Ethernet switches and we define an ad-hoc message scheduling algorithm that allows for the parallel delivery of messages with different sources and destinations;
- to compare the proposed solution with the TTEthernet one in terms of effectiveness in using the network bandwidth;
- to provide an experimental investigation of the proposed solution by means of the realization of an experimental test-bed equipped with Ethernet switches and market boards in which the proposed protocols are implemented.

The general communication system of a NMLV is described in Section 3. The FTTE-based communication system is illustrated in Section 4 while Section 5 is devoted to introducing the message scheduling algorithm. The main numerical results comparing the proposed solution with the TTEthernet one are reported in Section 6. The experimental test-bed and the performed tests are described in Section 7. Finally, conclusions and future research items are illustrated in Section 8.

2. Related Work

The introduction of real-time services in industrial environments has led to the development of a variety of protocols providing deterministic behavior such as Token Bus, Token Ring and a number of fieldbuses or control networks such as Profibus, SERCOS, CAN [15], MIL-STD 1553 [16]. Most of those solutions were developed some decades ago and are now considered as too limited in bandwidth with respect to other technologies as Ethernet. There is hence a need to upgrade the existing systems toward higher speeds and performances. This updating may be prohibitive due to the high cost of new developments in networking technologies of small market.

Ethernet is now available at higher and higher speeds and low costs because of its high diffusion. It has a history that spans over 50 years. It was initially a joint effort of Digital Equipment, Intel and Xerox to establish a flexible way to interconnect computers. The primary use case at the time was the connection of workstations to servers. Flexibility was crucial from the beginning, so the communication was organized using the best-effort principle, which is based on client-server communication. Best-effort means that clients can access data equally, and that no clients are granted priority access to data. If too many clients are trying to access a single path at the same time, the communication delay is not predictable.

For this reason, changes to the Ethernet standard have been proposed to provide real-time guarantees such as maximum transfer delay, jitter in the transmissions and available bandwidth. For instance, 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. The TSN standards [17] enable deterministic real-time communication over Ethernet by using time synchronization and a schedule which is shared between network components. By defining queues based on time, Time-Sensitive Networking ensures a bounded maximum latency for scheduled traffic through switched networks. The first market products implementing TSN solutions have recently appeared.

Another promising solution able to guarantee hard delays is the TTEthernet solution [12] standardized by the Society of Automotive Engineers (SAE) in the SAE6802 standard. Time-Triggered Ethernet is a scalable networking technology that uses time scheduling to deliver deterministic real-time communication over Ethernet. It has been specifically designed for safe and highly available real-time applications, cyber-physical systems and unified networking. It is fully compatible with IEEE 802.3 Ethernet and integrates transparently with Ethernet network components. Time-Triggered Ethernet simplifies the design of fault-tolerant and high availability solutions for aerospace, automotive and industrial applications. Safety and redundancy are maintained at the network level without any need for application involvement.

TSN and TTEthernet solutions allows for the support of real time services in Ethernet networks, but they have led to substantial changes of the Ethernet technologies with developments of high cost product. As a matter of example, the cost of a TTEthernet switch can also reach the cost of $10,000. The objective of this paper is to study solutions based on the use of COTS Ethernet devices in which traditional Ethernet switches can be used of negligible cost and not higher than $100. We inherit the Flexible Time Triggered Ethernet bus solution proposed in [13] and extend it to the case in which the network is switched and equipped with switches. The hard delay is guaranteed thanks to the definition of a message scheduling algorithm that avoids resource contentions in the switches.

The interest towards COTS technology applied in aerospace field on the part of stakeholders is very high because of the low costs [18]. However, investigations about performance, robustness and reliability are still going. For instance, National Aeronautics and Space Administration (NASA) is evaluating [18] the reliability of Commercial-Off-The-Shelf (COTS) Electronics for Extreme Cold Environments.

3. Nano/Micro Launcher Communication System

The requirement to be capable of air, sea and land launch, strongly suggests to split the avionic architecture of a Nano/Micro Launch Vehicle (NMLV) into several modules, each one autonomous and capable of performing its part of the mission without the intervention of the other stages. A NMLV is characterized by autonomous-flight-stages in which each stage will be equipped with its own avionic system in order to perform its part of the launch mission (with the only exclusion of navigation sensors, present only on the orbital stage but whose data will be available to all the stages).

The communication system in an NMLV allows for the interconnection of the terminals reported in Figure 1. As a matter of example we consider a three stages launch vehicle. We can notice how each stage is equipped with an On Board Computer (OBC). We report in Table 1 the list and the meaning of the acronyms used for the communication system.

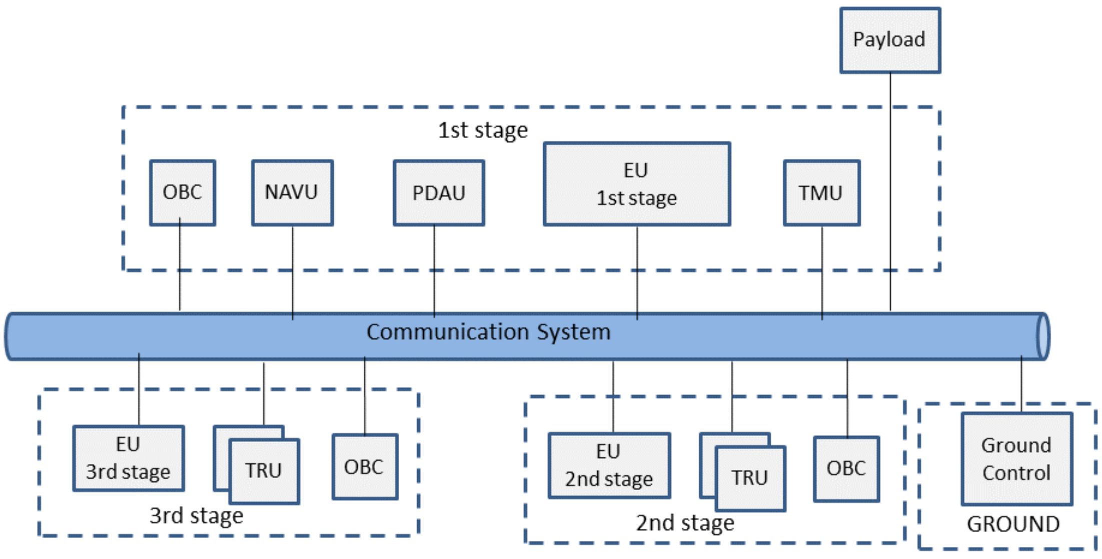


Figure 1. Terminals and communications system of a small launcher.

Table 1. List and meaning of acronyms.

Acronym	Meaning
GNC	Guide, Navigation and Control
TMS	Telemetry Subsystem
NAVU	Navigation Unit
OBC	On Board Computer
PDAU	Power Distribution and Actuation Unit
TVC	Thrust Vector Control
EU	Electronic Control Unit
TMU	Telemetry Master Unit
GC	Ground Control
TRU	Telemetry Remote Unit
PL	Payload

The communication system implements the functions of the following two logical Sub-Systems (SS):

- The Guidance, Navigation and Control (GNC) Subsystem, which groups sensors, actuators and data handling needed to perform a mission; it also includes the battery power sources;
- The Telemetry Subsystem (TMS), which groups sensors, data handling and Radio Frequency (RF) telemetry transmission chain.

In particular, the GNC subsystem is comprised of the on board electrical and electronic pieces of equipment providing the launcher flight control. The GNC S/S and TLM S/S are both connected to the communication system. Their components are listed hereafter:

- Navigation Unit (NAVU) with its flight software;
- On Board Computer (OBC) with its flight software that will perform the GNC functions needed during the flight and the management of other functions of the launcher on the basis of a "mission timeline";
- Power Distribution and Actuation Unit (PDAU) that performs several actuations and operations needed by flight management (main engine firings, stage separations, etc.);
- Thrust Vector Control (TVC) subsystems, one for each stage, comprised of one Electronic control Unit (EU) and one mechanical actuator;
- Telemetry Master Unit (TMU) that receives telemetry messages from all of the terminals and send them to the ground station;
- Ground Control (GC) that follows the preliminary and the first phases of the mission;
- Telemetry Remote Units (TRUs) collect the sensor data from each stage and send the acquired information to the TMU; currently in the 1553B bus of VEGA [3], these units are connected to another communications system different from the one transporting GNC messages.

The terminals are involved in transferring three different type of messages:

- GNC messages: they are needed for the guide, navigation and control operations;
- telemetry messages: they are sent to Ground using the TMU equipment; the TMU will manage two types of data: (i) CVI (Control Visuel Immediate) transmitted to Ground, immediately visible on screen by mission control and used by Mission Control to decide the launch vehicle destruction in case of anomalies; (ii) CVD (Control Visuel Differe) transmitted to ground as soon as possible and stored on Ground for post-processing purposes.
- sporadic messages: they belong to the sporadic sequences that are a set of predefined messages separated by predefined time intervals; a sequence is executed once, or more than once along the mission, to perform sporadic functional orders (e.g., stage separation, engine ignition, etc.).

The interest towards a novel communication architecture is due to the bandwidth limit of the communication systems for small launchers based on 1553B bus. This capacity is not higher than 1 Mb/s and it has been saturated. As a matter of example, the communication system would not be able to support new telemetry devices or new services. A TTEthernet-based solution [11,12] allows for the achievement of the goal with the introduction of devices (End Systems and Switches) able to allocate static bandwidth resources with the consequence of guaranteeing hard delay. The problem of TTEthernet is its high cost. For this reason we propose and evaluate a FTTE-based [13,14] communication architecture designed ad hoc for a launcher network and using COTS technology.

4. Flexible Time Triggered Ethernet Architecture for Nano/Micro Launcher Networks

The proposed FTTE-based architecture for Nano/Micro Launcher Networks is reported in Figure 2. The architecture is composed by three main planes:

- the data plane that delivers data messages and manages the message sending/receiving on redundant channels;
- the synchronization plane that performs the synchronizations of local clocks of the end systems;
- the management plane that configures the network, by updating/changing the System Requirement Database (SRDB) of a master node that stores the time instants in which the messages are scheduled to be sent.

Finally, we highlight that the switches implement the Medium Access Control (MAC) and physical layers only and support the synchronization protocol IEEE1588. For this reason, they are compliant with traditional COTS Ethernet switches. Both terminals and switch interfaces are configured as trunk mode ports of IEEE 802.1Q Virtual Local Area Network (LAN) [19]. In particular, each communication between source and destination ESs of the NMLV communication system is supported by a Virtual Local Area Network (VLAN) characterized by a VLAN Identifier [19].

Next we describe the Data Link Layer of the proposed NMLV architecture in Section 4.1 while an example of network operation mode is illustrated in Section 4.2.

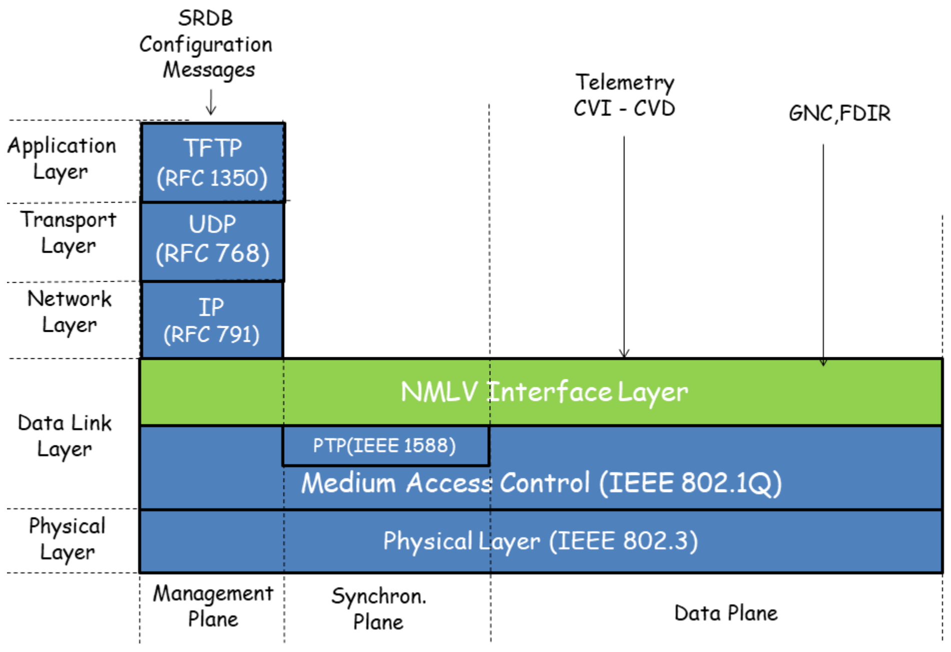


Figure 2. General Architecture of the NMLV Communication System.

4.1. Data Link Layer

In the general architecture of the NMLV Communication System, we define two sublayers: the traditional Medium Access Control (MAC) layer based on the IEEE 802.1Q [19] and the NMLV Interface Layer. The IEEE 1588 Synchronization layer is also introduced in the architecture to allow the synchronization of the network nodes so as to guarantee the support of the real-time layer service. The integration of the offered services needs the definition of a sophisticated scheduling of the MAC frames in end systems. We describe the IEEE 802.1Q and the NMLV Interface layers in Sections 4.1.1 and 4.1.2 respectively. The scheduling functionality is briefly illustrated in Section 4.1.3.

4.1.1. IEEE 802.1Q Layer

The architecture of the NMLV communication system is based on full-duplex operation that does not require the support of the collision handling functions. In the NMLV Communication System, the MAC layer is based on IEEE 802.1Q that is the standard for tagging frames on a trunk (aggregation of flows with the same path and characterized by the same QoS parameters) and support up to 4096 VLANs. The IEEE 802.1Q frame structure is reported in Figure 3. Beyond the Ethernet traditional fields, the IEEE 802.1Q frame includes the VLAN tag of 4 bytes, characterized by the following fields:

- Tag Protocol Identifier (TPID) is a 16-bit field set to a value of 0x8100 in order to identify the frame as an IEEE 802.1Q-tagged frame;
- Priority Code Point (PCP) is a 3-bit field which refers to the IEEE 802.1p class of service priority. This field indicates the frame priority level which can be used to support different QoS, in terms of delivery delay;

- Canonical Format Indicator (CFI) is a 1-bit field that indicates if the MAC address is in non canonical (1) or canonical (0) format;
- VLAN Identifier (VID) is a 12-bit field that uniquely identifies the VLAN to which the frame belongs.

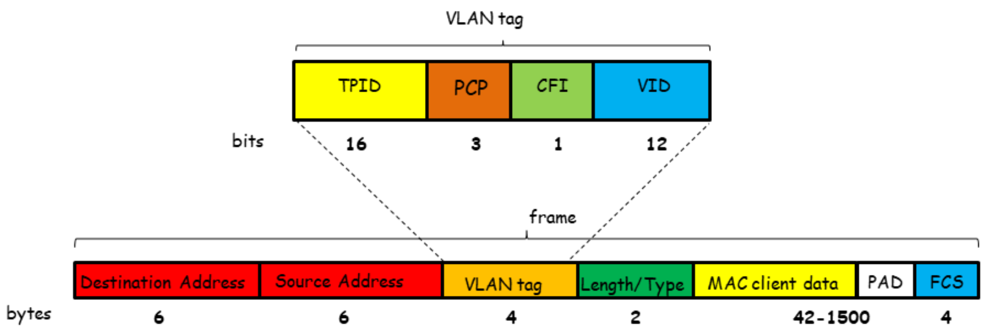


Figure 3. IEEE 802.1Q Frame.

4.1.2. NMLV Interface Layer

The introduction of the NMLV Interface layer leads to real-time behavior guarantees for the communication system. It inherits the functionalities of the FTTE standard described in [13]. The main differences could be summarized as follows:

- we remove the polling requests mechanism of the asynchronous window in the NMLV; we assume that all of the real-time traffic is periodic and a priori scheduled so as to emulate the circuit behavior of the traditional communication system for launch vehicles as 1553B bus [2];
- the NMLV layer supports the spatial redundancy [12] and end systems has to send the message on two output interfaces and to number the messages so that the receiving End System (ES) is able to recognize two copies of a same message; for this reason, a Sequence Number (SN) field is inserted in the NMLV Protocol Data Unit (PDU);
- the NMLV layer defines the scheduling instants for the messages so as to exploit the filtering operation of the switches and allowing for the parallel sending of messages that have no common links on their path connecting the source and destination ESs; that leads to an increase in the bandwidth use.

The NMLV layer has one layer service only referred to as Real-Time Layer Service (RT-LS). It allows for frame delivery with deterministic delay and limited jitter. It is offered to periodic messages flows. The frames are sent/received to/by the end systems in off-line scheduled time intervals; the successfully delivering of frames is possible if the involved networks elements (end systems and switches) are correctly synchronized by the synchronization protocol. The frame delivery instants are determined by a scheduling algorithm. The frames in which the messages requiring RT-LS are inserted is based on the IEEE 802.1Q standard frame and the NMLV PDUs are represented in Figure 4. Two types of NMLV PDUs are defined. They are referred to as Triggered and Real-Time Messages and are reported in Figure 4a,b respectively. For each field of the messages, we also report its length (bit or bytes). In both message types, the first field (MSG_ID) identifies the message type. The FTTE protocol organizes the message transmission in periods of equal duration and referred to as Elementary Cycles (EC). At the beginning of every EC, the Triggered message is sent by a Master ES. It notifies the ESs with the instants in which they have to start their transmission. Then the next fields of the MSG_ID in the Triggered Message contain the number of data messages that should be transmitted in that EC and, for each of these messages, the respective ES identifier (ID) and sending time (ST). The Real-Time message frame contains flags (FLAGS) and data fields (Real-Time Data) respectively.

Finally, a Sequence Number (SN) is added at the end of each type of message to handle the spatial redundancy [11,12].

4.1.3. Scheduling Functionality

The synchronized local clocks and the pre-configured schedule specify the temporal position of the Real-Time frames on the timeline. The sending times are a priori scheduled and all information are contained in the System Requirement Database of the Master End System (ME) of each stage. In every EC, the ME has to prepare the Triggered Message for that EC that contains the transmission instants for the enabled communications in that EC and disseminates it to the Slave End Systems. An heuristic for the determination of the message scheduling instants that avoids link contentions is proposed in Section 5. The proposed heuristic is bandwidth effective because it takes into account the filtering capacity of the Ethernet switches and allows for the parallel delivery of messages with different source and destination ESs.

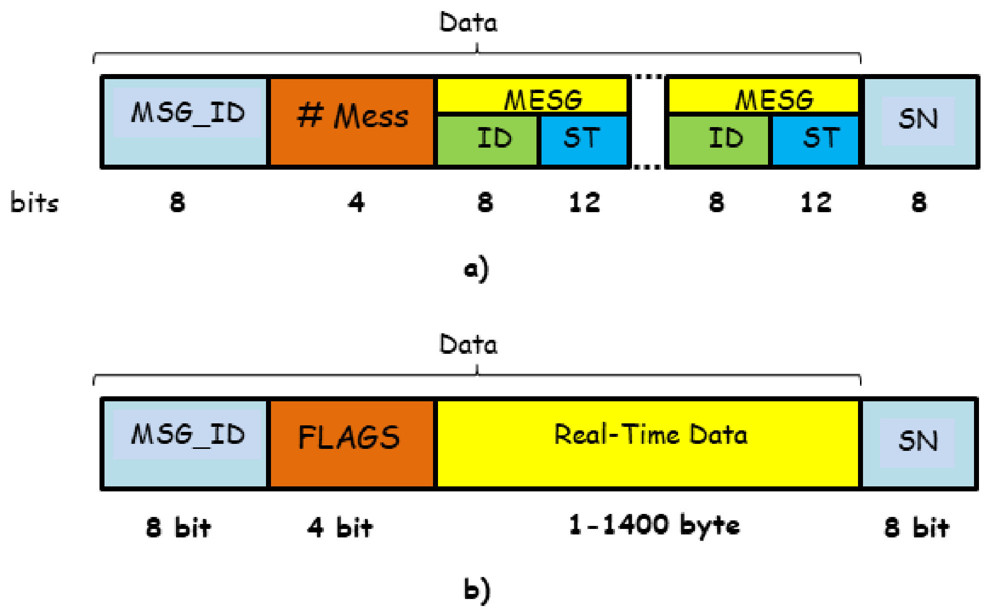
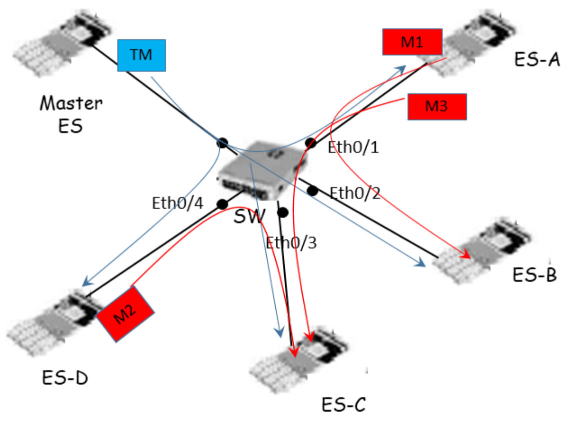


Figure 4. NMLV Protocol Data Unit. Triggered Message (**a**); Real Time Message (**b**).

4.2. Network Operation Mode

We illustrate the network operation mode in the case of the simple network of Figure 5 equipped with five ESs (Master ES, ES-A, ES-B, ES-C, ES-D) and one switch (SW). We assume that three messages M1, M2 and M3 have to be transferred, between the ES tuples ES-A-ES-B, ES-D-ES-C, ES-A-ES-C. We also assume that all of the messages have the same length and the transferring total time, including processing, transmission and propagation times, is equal to 10 µs. The Master End System emits the Triggered Message (TM) to all of the ESs; the scheduling times of M1, M2 and M3 are reported in it.



	Mesg ID	Sender	Receiver
M1	1	ES-A	ES-B
M2	2	ES-D	ES-C
M3	3	ES-A	ES-C

	Master ES	ES-A	ES-B	ES-C	ES-D
Switch Port	Eth0/0	Eth0/1	Eth0/2	Eth0/3	Eth0/4

Figure 5. Network example with five End Systems and one switch; the Triggered message is sent to all of the End Systems; three messages have to be transported.

Each ES has a table reporting the VID of the VLANs in which its messages have to be transferred. These VIDs are the ones associated to the VLAN interconnecting the source ES and the destination ESs of the message. As a matter of example, we report the tables of ES-A and ES-D in Figure 6a,b. Finally, the switch is configured so as to support the VLANs needed to interconnect the ESs. We report in Figure 6c the switch table with the values of VIDs concerning the VLANs on which the messages M1, M2 and M3 have to be transferred.

ES-A's Table

Msg ID	VID
1	10
3	30

a)

ES-D's Table

Msg ID	VID
2	20

b)

Switch Table

VID	Ports
10	Eth0/1 Eth0/2
20	Eth0/3 Eth0/4
30	Eth0/1 Eth0/3

c)

Figure 6. ES-A (**a**) and ES-D (**b**) Tables reporting the VID for the transport of the messages M1, M2, M3. VLAN configuration in the switch (**c**).

The network operation mode can be summarized as follows:

- The master ES applies the NMLV scheduling algorithm and determines the sending instants of the messages M1, M2 and M3; the application of the algorithm may lead to the scenario reported in Figure 7 where the message scheduling instants that avoid contention on the network links may be $t_{M1} = 11$ µs, $t_{M2} = 11$ µs and $t_{M3} = 22$ µs;

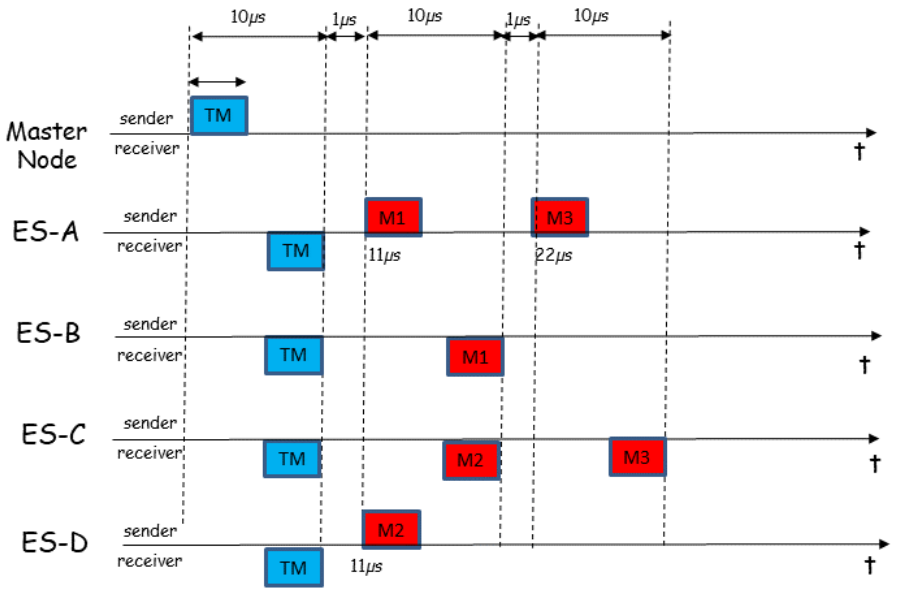


Figure 7. Scheduling instants evaluated by the NMLV scheduling algorithm.

- The master ES sends the TM, enveloped in an 802.1Q frame, to all ESs in the network; the TM is shown in Figure 8; it provides information on the sending time of the three messages M1, M2 and M3;

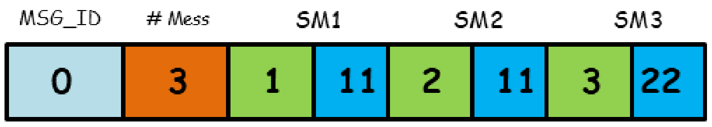


Figure 8. Field values of the Triggered Message.

- The ES-A sends in the instants $t_{M1} = 11$ μs and $t_{M3} = 22$ μs the messages M1 and M3 respectively, while the ES-B sends in the instant $t_{M2} = 11$ μs the message M2; the messages M1, M2 and M2 are carried in 802.1Q frames with VID equal to 10, 20 and 30 respectively.

5. Heuristic for Message Scheduling

The objective of the heuristic is to schedule the instants in which the ESs have to transmit the messages so as to avoid bandwidth contentions in any network link. We assume that the time axis is organized in ECs numbered with an integer starting from the value 0. Because most of sensor and actuation messages of a launch vehicle are periodically sent, we assume periodical messages. Next we introduce the following input parameters of the heuristic:

- T_{EC}: EC duration time;
- P: number of message types, each one characterized by a certain periodicity;
- T_j $(j = 1, \cdots, P)$: period of the i-th type messages expressed in terms of number of ECs; we assume that the values T_j $(j = 1, \cdots, N_p)$ are ordered in increasing order $(T_1 < \cdots, T_j < \cdots, P)$;
- Q: Multiple Common Minimum of the values T_j $(j = 1, \cdots, P)$;
- $\Lambda_{i,j}$ $(i = 0, \cdots, Q-1; j = 1, \cdots, P)$: set of messages of period M_j and with offset i; that is to say that the messages of the set $\Lambda_{i,j}$ have to be sent in the ECs $i + kM_j$ $(k = 0, 1, \cdots)$;
- Π_m: network path in which the message m has to be routed; Π_m is composed by a set of network links;
- S_m: ES emitting the message m.

The outputs of the heuristic are the scheduling instants T_m of the messages $m \in \Lambda_{i,j}$ $(i = 0, \cdots, Q-1; j = 1, \cdots, P)$, evaluated starting from the beginning of any EC in which m has to be sent. The determination of T_m will lead to send the messages in the instants $iT_{EC} + T_m + kT_jT_{EC}$ $(k = 0, 1, \cdots)$.

The proposed heuristic is applied in each EC and its goal is the one to first select the messages than can be sent earlier by avoiding any link contention. It is based on the following variables:

- t_S: transmission end instant of the last message sent by the ES S; this variable is updated every time in which the heuristic decides for the scheduling of a message emitted by the ES S;
- Γ_L: set of messages that have already been scheduled and use the network link L;
- Ω_r: higher time instant in which the message r has been completely received by the destination ESs.

The main steps of the heuristic are described in Algorithm 1.

Algorithm 1 HEURISTIC FOR MESSAGE SCHEDULING

```
1:  Input: set $\Lambda_{i,j}$ $(i = 0, \cdots, Q-1; j = 1, \cdots, P)$
2:  $i = 0; j = 1$
3:  while $j \leq P$ do
4:      while $i \leq Q-1$ do
5:          while $\Lambda_{i,j} \neq \emptyset$ do
6:              Initialize $\Lambda_{aux} = \Lambda_{i,j}$
7:              while $\Lambda_{aux} \neq \emptyset$ do
8:                  Choose $n \in \Lambda_{aux}$
9:                  Set $\Lambda_{aux} = \Lambda_{aux} - \{n\}$
10:                 $T_n = \max(t_{S_n}, \max_{L \in P_n, r \in \Gamma_L} \Omega_r)$;
11:             end while
12:             $m = \arg\min_{n \in \Lambda_{i,j}} T_n$;
13:             Set $T_m = \min_{n \in \Lambda_{i,j}} T_n$
14:             Set $\Lambda_{i,j} = \Lambda_{i,j} - \{m\}$
15:         end while
16:         $i = i + 1$
17:     end while
18:     $j = j + 1$
19: end while
20: Output: $T_m$ $m \in \Lambda_{i,j}$ $(i = 0, \cdots, Q-1; j = 1, \cdots, P)$
```

Λ_{aux} is an auxiliary variable. The heuristic chooses to schedule the messages in increasing order of periodicity (Line 2) and next in order increasing of EC index (Line 4). For all of the messages n in the set $\Lambda_{i,j}$ not scheduled yet, the heuristic evaluates the first instant T_n in which the message n can be sent (Line 10) so as to avoid bandwidth contentions in any network link L of the path P_n in which the message n has to be routed. To guarantee contention lack, the message has to be sent after the transmission end instant of all of the messages already scheduled on the path P_n. Among the messages n the heuristic schedules, the one with smaller T_n (Line 12). The operation is repeated for the remaining messages until the set $\Lambda_{i,j}$ becomes empty (Line 5).

From the steps performed in the Algorithm 1 we can notice that: (i) the first while cycle performs P times (Line 3); (ii) the second while cycle performs Q times (Line 4); (iii) the scheduling instant of one message at a time is evaluated for the set $\Lambda_{i,j}$ (Line 5); (iv) the evaluation of each of these scheduling instants involve, for each message not scheduled yet, the evaluation of the earliest instant in which the message can be sent without causing contention on any output link of the message (Lines 7 and 12); (v) the evaluation of this earliest time is evaluated by performing as many maximum operations as the length of the message path (Line 10). According to these remarks, if we denote with N_Λ the maximum of the cardinality of the sets $\Lambda_{i,j}$ $(i = 0, \cdots, Q-1; j = 1, \cdots, P)$ and with ξ the length of the longest expressed as number of hops, we can conclude that the computational complexity of the proposed heuristic is $O(PQN_\Lambda^3\xi)$.

6. Numerical Results

We report some results on the bandwidth efficiency that the proposed solution for realizing the NMLV communication system allows us to achieve. In particular, we compare three solutions:

- a FTTE-based solution with the use of store and forward traditional Ethernet switches and when the scheduling heuristic of Section 5 is applied;
- a FTTE-based solution with the use of cut-through Ethernet switches and when the scheduling heuristic of Section 5 is applied; the cut-through solution allows for the minimization of the switch forward delay because the frame can be forwarded not just when the frame header is read;
- a TTEthernet-based solution [12] with the application of a scheduling heuristic similar to the one illustrated in Section 5 by taking into account that the TTEthernet switch allows the introduction of offset times [12] to solve the bandwidth contentions in the network links; obviously this solution is bandwidth effectiveness but its disadvantage is the higher cost considering that a TTEthernet switch can cost up to 40 times the cost of a traditional Ethernet switch.

We consider the network topology of Figure 9. It is composed by three switches denoted as SW1, SW2 and SW3 respectively and three stages with one switch in each of them. The Navifation Unit (NAVU) is attached to the switch of the first stage as well as an On-Board Computer (OBC) (OBC1 for the first stage), the Telemetry Master Unit (TMU) and an Actuation Unit (ACTU) (ACTU1 for the first stage). In the other remaining stages, one OBC (OBC2 and OBC3 for the second and third stages respectively) and one ACTU (ACTU2, ACTU3 for the second and third stages respectively) are attached. The messages sent to the TMU only, can be routed only using the telemetry links through the switches SW1, SW2, SW3. The links interconnecting these switches are duplicated so as to route on different network paths the GNC and FDIR messages and the ones directed to the TMU only. The messages directed to the TMU do not need to be delivered with hard delay.

We consider a real traffic scenario and requirements. In particular, we have considered a set of GNC and telemetry messages extrapolated by the VEGA message set [3]. We report in Table 2 the characteristics of the GNC and telemetry messages respectively. For each message we report the source and destination terminals, the message length (byte), the period and the offset expressed in terms of number of ECs. We assume the EC duration time of 4.9996 ms, which is equal to one of the minor frames in VEGA [3]. The link bandwidth is 100 Mbps.

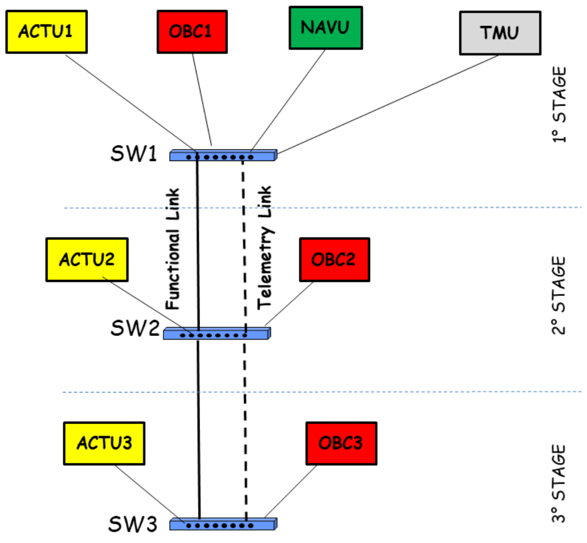


Figure 9. Network Topology.

Table 2. Launcher Messages. Source (S) and Destinations (D) of the messages are reported as well as their length (bytes), period and offset expressed in terms of number of ECs.

Message	S	D	Length	Period	Offset	Message	S	D	Length	Period	Offset
M_1	OBC1	All	40	1	0	M_{19}	OBC3	TMU	400	8	2
M_2	OBC1	ACTU1 TMU	40	1	0	M_{20}	OBC3	TMU	400	8	6
M_3	OBC2	ACTU2 TMU	40	1	0	M_{21}	NAVU	TMU	128	8	2
M_4	OBC3	ACTU3 TMU	40	1	0	M_{22}	ACTU1	TMU	128	8	4
M_5	OBC1	ACTU1 TMU	40	8	3	M_{23}	ACTU2	TMU	128	8	4
M_6	OBC2	ACTU2 TMU	40	8	3	M_{24}	ACTU3	TMU	128	8	4
M_7	OBC3	ACTU3 TMU	40	8	3	M_{25}	OBC1	ACTU1 TMU	40	1	0
M_8	OBC2	OBC1 TMU OBC3	64	4	1	M_{26}	OBC1	ACTU1 TMU	40	1	0
M_9	ACTU1	OBC1 TMU	40	8	5	M_{27}	OBC1	ACTU1 TMU	40	1	0
M_{10}	ACTU2	OBC2 TMU	40	8	5	M_{28}	OBC2	ACTU2 TMU	40	1	0
M_{11}	ACTU3	OBC3 TMU	40	8	5	M_{29}	OBC2	ACTU2 TMU	40	1	0
M_{12}	OBC1	ACTU1 NAVU TMU	40	1	0	M_{30}	OBC2	OBC1 OBC3 TMU	400	1	0
M_{13}	OBC2	ACTU2 NAVU TMU	40	1	0	M_{31}	OBC2	ACTU2 TMU	40	1	0
M_{14}	OBC3	ACTU3 NAVU TMU	40	1	0	M_{32}	OBC3	ACTU3 TMU	40	1	0
M_{15}	OBC1	TMU	400	8	0	M_{33}	OBC3	ACTU3 TMU	40	1	0
M_{16}	OBC1	TMU	400	8	4	M_{34}	OBC3	OBC1 TMU	400	1	0
M_{17}	OBC2	TMU	400	8	1	M_{35}	OBC3	ACTU3 TMU	40	1	0
M_{18}	OBC2	TMU	400	8	5						

We have evaluated three flight phases referred to as Phase-1, Phase-2 and Phase-3. In particular, Phase-1 is the one in which all of the three stages are attached, Phase 2 is the one in which the Stage-3 is detached while Phase-3 is the one in which both Stage-2 and Stage-3 are detached. Let us introduce for each link the Bandwidth Allocation Factor (BAF). It characterizes the consumed effective bandwidth including both the one needed to carry the messages and the one not used to avoid contentions on any link. The performance comparison between TTEthernet and FTTE is reported in Figures 10–12 for Phase-1, Phase-2 and Phase-3 respectively. We show the BAF as a function of the network links denoted with the network elements (ES and switches) interconnecting them. For the tuple of switches SW1-SW2 and SW3-SW3, we report four values of BAF relative to the link directionality and the link type (Functional and Telemetry).

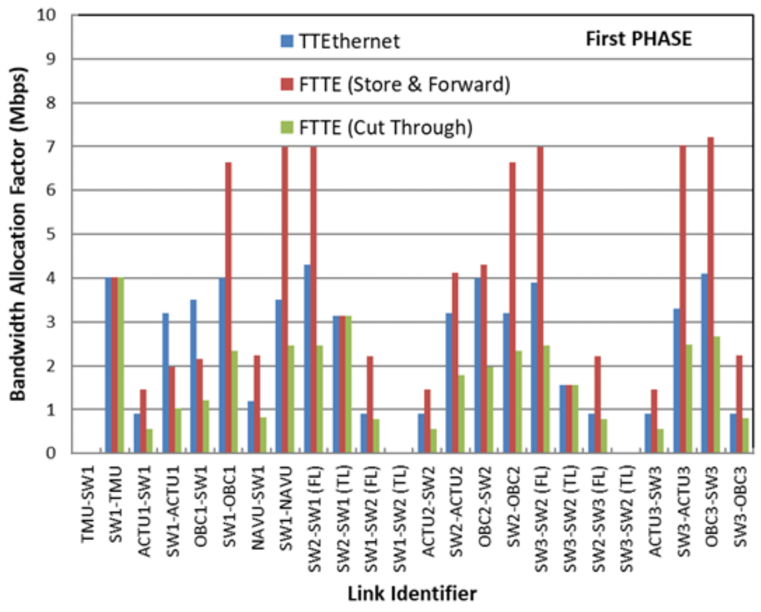


Figure 10. BAF as a function of the network links for the Phase-1.

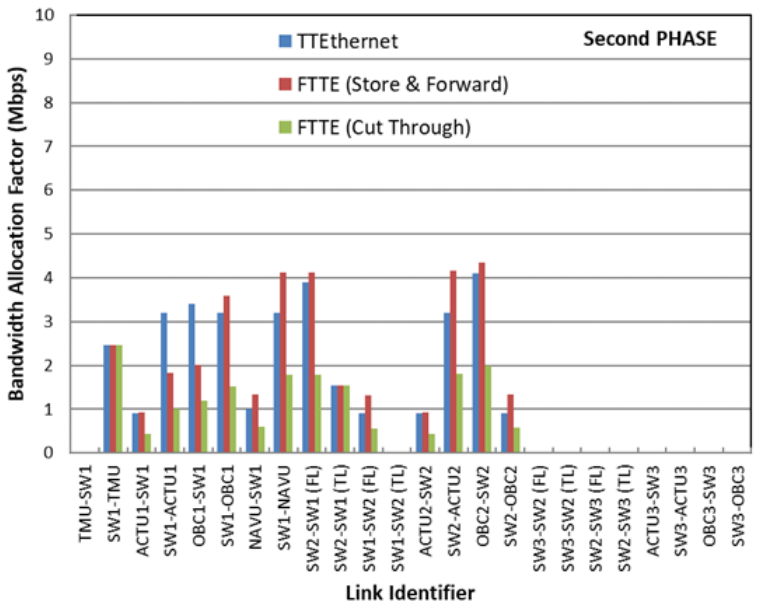


Figure 11. BAF as a function of the network links for Phase-2.

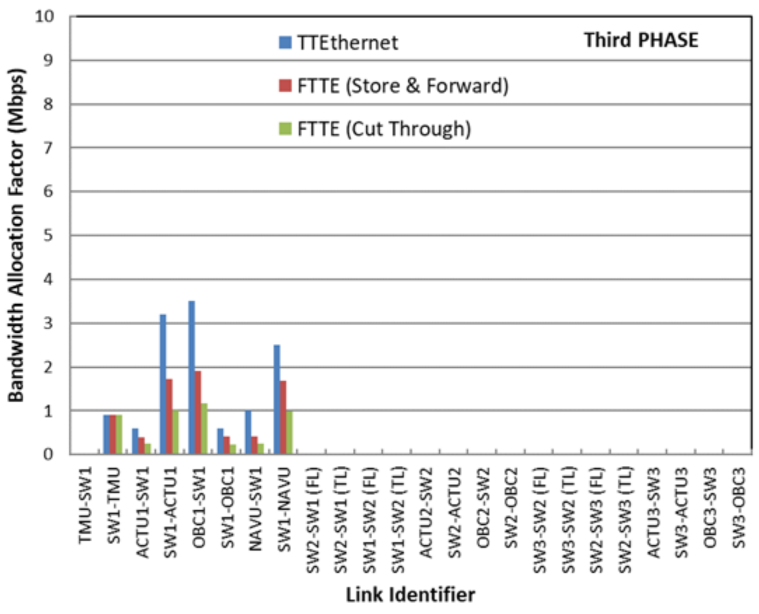


Figure 12. BAF as a function of the network links for Phase-3.

As expected, from Figures 10–12, we can notice how the FTTE solution with Store and Forward switches is the least bandwidth effective one in Phase-1 and Phase-2 because differently from TTEthernet solution, the message scheduling is only implemented in the ESs. However, it increases by only a few Mbps the BAF with respect to TTEthernet solution but with a big advantage of costs given the possibility to exploit COTS devices and in particular traditional Ethernet switches. Furthermore, we notice how the FTTE (Store and Forward) performance significantly improves in Phase-3 where the detachment of the first two stages allows for smaller network delays that consequently leads to better performance of the scheduling algorithm. In Phase-3, even better performance than the TTEthernet can be obtained. The worse performance of TTEthernet is due to the Timely Blocking problem that leads to the bandwidth inefficiency caused by empty intervals introduced by the switches to guarantee hard delay [20–22]. For instance, BAF of 1.91 Mbps and 3.50 Mbps are obtained in the OBC1-SW1 link for FTTE (Store and Forward) and TTEthernet solutions respectively.

Finally, we can observe how the FTTE solution with Cut Through switches allows for a remarkable reduction in BAF. For some links this solution allows for better BAF values with respect to the TTEthernet solution. For instance, in the case of Phase-2, for the NAVU-SW1 link, we have BAF values of 0.25 Mbps and 1 Mbps for the FTTE (Cut & Through) and TTEthernet solutions respectively.

7. Experimental Test-Bed

Next we describe the Demonstrator Hardware/Software in Section 7.1. The Test Configurations and some experimental results on delay and jitter are illustrated in Section 7.2.

7.1. Demonstrator Hardware/Software

The demonstrator allows for the experimental evaluation of a single flight stage and consists of the following parts:

- Three Hercules TMS570 MCU Development Kits. They emulate an OBC, a NAVU and an ACTU respectively; the development board includes an on-board XDS100v2 JTAG emulator, access to connectivity peripherals and access to all other peripheral pins; the kits also include a DC power supply, cables, software environment, demo software and code examples;
- Ethernet Switch. It is based on COTS technology and implements VLAN;

- Network Monitoring PC. It is a PC connected to the same Ethernet network as the units under test by means of network tap and in order to spy on the traffic using Wireshark. The test support software will be mainly composed by GNC and telemetry traffic generator and software for the and frame delivery time and loss measurement in order to verify the correctness of the network operation mode. Some test configurations will be defined in Section 7.2;
- Development PC. It is a PC containing the Software Development Environment (SDE); it is used to code, build, run and debug software on the target platforms.

The FTTE communication library is realized in C programming language. The library allows managing the deterministic communication on Ethernet and will be shared by all the applications. Typical services exposed by this library allows sending/receiving message, configuring the network, etc.

The development environment is developed in C programming language. It is composed by three components:

- FreeRTOS Operating System [23]. The Operating System shall be FreeRTOS executed on the top of Hercules board. FreeRTOS is a real-time operating system kernel for embedded devices that has been ported to about 35 microcontrollers. FreeRTOS kernel itself consists of only three C files. To make the code readable, easy to port, and maintainable, it is written mostly in C, but there are a few assembly functions included. FreeRTOS provides methods for multiple threads or tasks, mutexes, semaphores and software timers. A tick-less mode is provided for low power applications. Thread priorities are supported.
- HALCoGen [24]. It is an application that represents the Hardware Abstraction Layer Code Generator of the Hercules board. HALCoGen provides a graphical user interface that allows the user configuring peripherals, interrupts, clocks, and other microcontroller parameters. Once the device is configured, the user can generate peripheral initialization and driver code, which can be imported into Code Composer Studio. It is important to underline that HALCoGen includes support for FreeRTOS. Indeed, the configuration generated with Halcogen allows completely setting the Hercules board, also taking on the role that is usually played by the Firmware.
- Code Composer Studio [25]. It is an integrated development environment (IDE) that supports development of applications on Texas Instruments microcontrollers and embedded processors. It is based on Eclipse and comprises a suite of tools used to develop and debug embedded applications. It includes an optimizing C/C++ compiler, source code editor, project build environment, debugger, profiler, and many other features. Code Composer Studio communicates with the board for programming and debug using USB XDS100v2 JTAG.

7.2. Tests Description and Results

The sent/received messages are structured as described in Sections 4.1.1 and 4.1.2 and are reported in Table 3. For each of them we report: the message name, the VLAN ID field of Figure 3, the MSG_ID field of Figure 4, the sources and destination ESs, the period and the Offset expressed in terms of number of ECs whose duration is the one of the Minor Frame in VEGA launcher Table 3 that is 4.999 ms. Furthermore, Table 3 also reports the message scheduling time evaluated at the beginning of the EC.

Table 3. Messages generated in the Experimental Test-Bed.

Message Name	VLAN ID	MSG_ID	Source ES	Destination ES	Period (EC)	Offset (EC)	Scheduling Time (ms)
NMLV_TM	10	1	OBC	ACTB,NAVB	1	0	0
ACTU_C_EV	50	2	OBC	ACTU	1	0	1
ACTU_C_TVC	50	5	OBC	ACTU	8	2	3
NAVU_R_data	170	8	NAVU	OBC	4	1	1

The function of each message is the following:

- NMLV_TM is the Time Triggered Message that coordinates the messages transmission so as to avoid any bandwidth contention; its length depends on the number of messages transmitted in the EC in which NMLV_TM is sent;
- ACTU_C_EV is the message sent from the OBC to the ACTU to switch on/off the Electro-Valves (EV); we assume that the Real_Time field length (Figure 4) of the message equals 8 bit with possibility to switch on/off up to eight EVs;
- ACTU_C_TVC is the message sent from the OBC to the ACTU to pilot the Electro-Mechanical-Actuator (EMA) of the TVCs; we assume that the Real_Time field length (Figure 4) of the message equals 9 bytes: one flag byte and a couple of 4 bytes, each one reporting a float that express in mm the displacement of an EMA;
- NAVU_R_data is the message sent from the NAVU to the OBC to provide the estimation of altitude, position and velocity of the NMLV by the navigation function; we assume that the Real_Time field length (Figure 4) of the message equals 72 bytes.

We perform three tests:

- Test-A: the network tap of Figure 13 is inserted between the Processor Board and the Ethernet switch and the following messages are collected: NMLV_TM, ACTU_C_EV, ACTU_C_TVC (emitted by the OBC) and NAVU_R_data (received by the OBC);
- Test-B: the network tap of Figure 13 is inserted between the Navigation Board and the Ethernet switch and the following messages are collected: NMLV_TM (received by the NAVU) and NAVU_R_data (emitted by the NAVU);
- Test-C: the network tap of Figure 13 is inserted between the Actuation Board and the Ethernet switch and the following messages are collected: NMLV_TM, ACTU_C_EV, ACTU_C_TVC (emitted by the OBC).

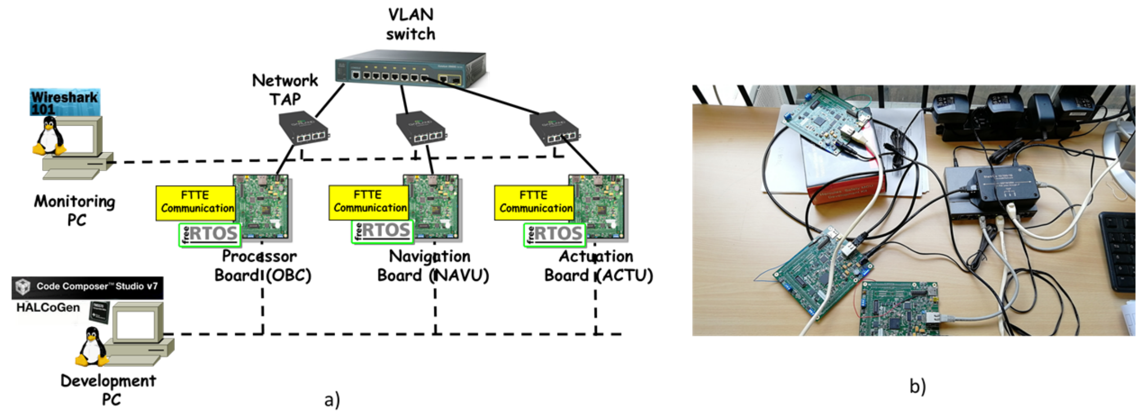


Figure 13. Scheme (**a**) and Picture (**b**) of the Experimental Test-bed.

The objective is the measure of the mean and the standard deviation of the inter-arrival times of the messages; the measured values are reported in Table 4.

We can observe how the implementation of the FTTE protocol allows for a correct delivery of the messages with the correct periodicity and a negligible standard deviation.

Table 4. Mean and Standard Deviation of the message inter-arrival times for Test-A, Test-B and Test-C.

Message	Test-A Mean (ms)	Test-A Standard Deviation (ms)	Test-B Mean (ms)	Test-B Standard Deviation (ms)	Test-C Mean (ms)	Test-C Standard Deviation (ms)
NMLV_TM	4.9996	0.0337	4.9996	0.0262	4.9996	0.0227
ACTU_C_EV	4.9987	0.0644	—	—	4.9987	0.0537
ACTU_C_TVC	39.9903	0.1075	—	—	39.9918	0.4143
NAVU_R_data	19.9989	0.4747	19.9996	0.1558	—	—

8. Conclusions

We have proposed an Ethernet-based communication system to be implemented in Nano and Micro Launch Vehicle. Its performance has been evaluated by simulation and with the realization of an experimental test-bed. The achieved results show how the proposed solution is bandwidth effective and allows for a performance comparable to very expensive solutions, such as TTEthernet. In particular, we have shown how the FTTE solution is more bandwidth effective in the flight phases in which some stages of the launcher have detached and the network is smaller so as to improve the performance of the scheduling algorithm which the FTTE protocol is based on. The results of the demonstrator, equipped with Ethernet switches and Hercules boards by Texas Instruments, have proved the feasibility of the proposed solution.

We hope that the positive results reported in this manuscript on the Flexible Time Triggered Ethernet can contribute to sensitizing the beginning of a standardization activity that is currently absent.

Author Contributions: V.E. has organized and written the paper. F.G.L. and F.V. have implemented and carried out the performance comparison of the TTEthernet and FTTEthernet solutions. F.G.L. and A.P. have implemented the experimental test-bed. S.C. have provided useful comments for the results section.

Funding: This research was funded by European Space Agency grant number ITT ESA AO/1-8275/16/NL/FE.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wekerle, T.; Filho, J.B.P.; da Costa, L.E.V.L.; Trabasso, L.G. Status and Trends of Smallsats and their Launch Vehicles: An Up-to-date Review. *J. Aerosp. Technol. Manag.* **2017**, *3*, 269–286. [CrossRef]
2. Wu, K.; Jiang, J.; Hu, M. Intelligent fault-tolerant 1553B bus system based on adaptive learning. In Proceedings of the 2011 IEEE third International Conference on Communication Software and Networks (ICCSN), Xi'an, China, 27–29 May 2011; pp. 378–381.
3. Accettura, A.; Balduccini, M.; Carducci, F.; De Lillis, A.; D'Aversa, E. VEGA: The History of an European Success. In Proceedings of the 2012 IEEE First AESS European Conference on Satellite Telecommunications (ESTEL), Rome, Italy, 2–5 October 2012.
4. Sandic, M.; Teslic, N.; Velikic, Y. Bandwidth utilization in deterministic networks. In Proceedings of the 2016 Zooming Innovation in Consumer Electronics International Conference (ZINC), Novi Sad, Serbia, 1–2 July 2016.
5. Nguyen, N.T.; Leu, M.C.; Liu, X.F. RTEthernet: Real-time communication for manufacturing cyberphysical systems. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3433. [CrossRef]
6. IEEE802.3-2005, IEEE Standard for Ethernet. 2005. Available online: https://standards.ieee.org/about/get/802/802.3.html (accessed on 10 July 2018).
7. Decotignie, J.D. The many faces of Industrial Ethernet. *IEEE Ind. Electron. Maga.* **2009**, *3*, 8–19. [CrossRef]
8. ARINC 644 Part 7, Aircraft Data Network Part 7 Avionics Full Duplex Switched Ethernet (AFDX) Network. 2007. Available online: http://standards.globalspec.com/std/204622/arinc-664-p7 (accessed on 10 July 2018).
9. SAE AS6802, Time-Triggered Ethernet. 2011. Available online: http://standards.sae.org/as6802/ (accessed on 10 July 2018).
10. Steiner, W.; Bauer, G.; Hall, B.; Paulitsch, B. *Triggered Ethernet: TTEthernet*; CRC Press: Boca Raton, FL, USA, 2011.

11. Eramo, V.; Lavacca, F.G.; Listanti, M.; Caporossi, S. Performance Evaluation of TTEthernet-based Architectures for the VEGA Launcher. In Proceedings of the 2018 IEEE Aerospace Conference, Big Sky, MT, USA, 3–10 March 2018.
12. Eramo, V.; Lavacca, F.G.; Listanti, M.; Caporossi, S. TTEthernet: A Networking Technology for the support of Real-Time Services in Launcher Networks. *IEEE Aerosp. Electron. Syst. Mag.* **2018**, doi:10.1109/MAES.2018.170161. [CrossRef]
13. Pedreiras, P.; Gai, P.; Almeida, L.; Almeida, G. FTT-Ethernet: A flexible real-time communication protocol that supports dynamic QoS management on Ethernet-based system. *IEEE Trans. Ind. Inf.* **2005**, *1*, 162–172. [CrossRef]
14. Derasevic, S.; Barranco, M.; Proenza, J. Designing fault-diagnosis and reintegration to prevent node redundancy attrition in highly reliable control systems based on FTT-Ethernet. In Proceedings of the 2016 IEEE World Conference on Factory Communication Systems (WFCS), Aveiro, Portugal, 3–6 May 2016; pp. 1–14.
15. Decotignie, J.D. Ethernet-Based Real-Time and Industrial Communications. *Proc. IEEE* **2005**, *93*, 1102–1117. [CrossRef]
16. Haverty, M. MIL-STD 1553—A standard for data communications. *Commun. Broadcast.* **1986**, *10*, 29–33.
17. IEEE Std 802.1AS-2011—IEEE Standard for Local and Metropolitan Area Networks—Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks. Available online: https://standards.ieee.org/findstds/standard/802.1AS-2011.html (accessed on 10 July 2018).
18. Song, M.; Yang-Scharlotta, J.Y.; Ashtijou, M.; Mojarradi, M. Evaluation of Commercial-Off-The-Shelf (COTS) Electronics for Extreme Cold Environments. In Proceedings of the 2018 IEEE Aerospace Conference, Big Sky, MT, USA, 3–10 March 2018.
19. IEEE8021.Q, IEEE Standard for Local and Metropolitan Area Networks Bridges and Bridged Networks. 2014. Available online: https://ieeexplore.ieee.org/browse/standards/get-program/page/series?id=68 (accessed on 10 July 2018).
20. Suethanuwong, E.; Hirunmutrapom, S.I. Searching of Best-effort Messages in TTEthemet Switches during the Timely Blocking Intervals. In Proceedings of the 2016 IEEE RIVF International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future, Hanoi, Vietnam, 7–9 November 2016.
21. Suethanuwong, E. Message Fragmentation of Event-Triggered Traffic in TTEthernet Systems Using the Timely Block Method. In Proceedings of the 2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT), New Delhi, India, 11–13 March 2016.
22. Zhou, X.; He, F.; Wang, W. Using network calculus on worst-case latency analysis for TTEthernet in preemption transmission mode. In Proceedings of the 2016 10th International Conference on Signal Processing and Communication Systems (ICSPCS), Gold Coast, Australia, 19–21 December 2016.
23. FreeRTOS Kernel, Software. Available online: https://www.freertos.org/ (accessed on 10 July 2018).
24. HALCoGen: Hardware Abstraction Layer Code Generator for Hercules MCUs. Available online: http://www.ti.com/tool/HALCOGEN (accessed on 10 July 2018).
25. Code Composer Studio (CCS) Integrated Development Environment (IDE). Available online: http://www.ti.com/tool/CCSTUDIO (accessed on 10 July 2018).