Protocol for Deterministic Data Delivery in SpaceWire Networks

Valentin Olenev, Elena Podgornova, Irina Lavrovskaya
Saint-Petersburg State University of Aerospace Instrumentation
Saint Petersburg, Russia
{valentin.olenev, alena.podgornova, irina.lavrovskaya}@guap.ru

***Abstract*—The paper presents a new scheduling protocol for SpaceWire networks which provides deterministic data delivery in the network and performs arbitration of data coming from several transport protocols. Firstly, we give an overview of TDMA-based network protocols that have been developed for the ground-based and onboard networks. Then, we present a new scheduling protocol which is based on the STP-ISS scheduling mechanism and extended with additional features.**

I. INTRODUCTION

Deterministic behavior is an important paradigm for verification and validation of real-time systems such as those on manned space vehicles and robotic spacecraft. Deterministic systems have predictable behavior, which is necessary to perform analysis to ensure requirements are met. Research done in the last 35 years in the area of system safety has combined fault tolerance and determinism to enable a new generation of robust safety critical systems. Even for systems that are not safety critical, where guaranteed availability is not a requirement, determinism is an important concept. These types of systems may use more advanced architectures that are distributed, use software buses, or employ time-triggered software and deterministic networks. So both safety critical systems and non-safety critical systems rely on deterministic systems.

Providing deterministic characteristics of the data transfer for the spacecraft that uses the SpaceWire technology is an essential problem, especially for autonomous vehicles like satellites. Deterministic data delivery guarantees that data from one node of the onboard network would be delivered to the target node during the fixed time slot. The developer can schedule all the onboard traffic and prevent the potential deadlock and data delivery delays. Such task is solved by using the specific communication protocol that includes the scheduling service. However, the scheduling problem becomes more complex, when we consider a number of communication protocols simultaneously operating in every node of the network. In this case we need a single scheduling instance that would be able to control all the data, that is transmitted to the network from the particular node. Deterministic characteristics are obtained by using time-division multiplexing.

Time-division multiplexing (or scheduling) is used in network technologies to obtain guaranteed latency and throughput for the user data, and to avoid conflicts with simultaneous network resources usage. Time multiplexing requires periodic synchronization, which can be based on time stamps distribution over the network. Such kind of time stamps are provided by SpaceWire networks and are called time-codes. Time should be divided into a number of time intervals.

The main goal of this paper is to overview the currently existing communication protocols that use scheduling quality of service and propose a solution for scheduling of the traffic from several transport protocols operating on top of SpaceWire.

II. EXAMPLES OF TDMA-BASED DETERMINISTIC DATA DELIVERY PROTOCOLS

Time-division multiplexing problem is rather important issue for communication protocols where deterministic data delivery is required. Time multiplexing is actively used in 2G and 3G mobile networks, as well as in some wireless personal networks, such as Bluetooth, ZigBee, Ubiquiti.

SpaceWire on-board networks also require time-division multiplexing solutions as the technology that is used in spacecraft and avionics. We reviewed a number of ground-based and on-board network protocols and current section provides an overview of these protocols with scheduling quality of service [1], [2], [3]. This will help to understand, which mechanisms and algorithms are used to provide the deterministic data delivery for different tasks.

A. TTCAN

The Time-Triggered Controller Area Network protocol (TTCAN) [4], [5] is the extension of the standard CAN protocol. The main features of TTCAN are the synchronization of the communication schedules of all CAN nodes in a network, the possibility to synchronize the communication schedule to an external time base, and the global system time.

TTCAN is based on a time triggered and periodic communication which is clocked by a time master's reference message. This message can be easily recognized by its identifier. Synchronization of each CAN node is achieved by this reference messages. A period between two consecutive reference messages is called a basic cycle. The basic cycle consists of several time windows of different size. Basic cycles are not always identical in order to be able to transmit messages at different periodic frequencies. Scheduling tables determine when each node is allowed to send messages. It comprises several basic cycles and is repeated indefinitely until the vehicle network is turned off. The scheduling table has to be configured during a design phase and cannot be changed while runtime.

ISSN 2305-7254

B. Byteflight

Byteflight [6] is a bus system that satisfies the highest requirements concerning transmission rate, reliability and interference immunity. The basis for Byteflight data communication is formed by cyclical synchronization pulses (SYNC pulses). Any node can be configured as a SYNC master that generates synchronization pulses every time interval. Bus access is regulated according to the Flexible Time Division Multiple Access method (FTDMA). With this method, all nodes start 'slot counters' that are triggered by the synchronization pulse. Time window is divided into mini time slots with a fixed length. Every mini slot is assigned to a specific message. A message can exceed the length of the mini slot, consecutive mini slots will be postponed until the message is completely sent. When a slot counter reaches an identifier value for which a transmission request is present, the corresponding message is transmitted via the bus, and all the slot counters stop at the current value for the duration of the transmission. Once the transmission is complete, the slot counters begin to count upwards again.

The FTDMA procedure described above is thus a purely time-controlled bus-access process. Nevertheless, it allows the guaranteed or deterministic transmission of a specific number of high-priority messages in every communication cycle even when the bus capacity is fully used, and simultaneously permits flexible and statistic assignment of the bandwidth for the rest of the messages if bus load is low enough.

C. Flexray

FlexRay is a fast, deterministic and fault-tolerant bus system for automotive use [7], [8]. It is based on the experience of Daimler-Chrysler with the development of prototype applications and the developed by BMW Byteflight communication system. The core concept of the FlexRay protocol is a time-triggered approach to network communications.

In a TDMA system the communication cycle is broken down into several time segments: static segment, dynamic segment, symbol window and network idle time. The static segment is broken down into smaller sections called static slots. Every static slot is of the same duration. During transmission each slot is assigned to a specific message and only that message can transmit during that time slot. The Dynamic segment is an optional section of the communication cycle. It is broken down into smaller sections known as minislots. If a node wishes to communicate it must wait until its minislot comes around. If no transmission occurs after a given period the minislot counter is incremented and the node with the next message/frame identification number (ID) may begin transmission of data. The data will only be sent if there is enough time left in the dynamic segment. In this way the dynamic segment is priority driven with the message with the lowest ID having the highest priority, just like CAN. A symbol is used to indicate a need to wake up a cluster amongst other things. This depends on the symbol sent and the status of the controller at the time. Within the symbol window a single symbol may be sent. If there is more than one symbol to be sent then a higher level protocol must determine which symbol gets priority as the FlexRay protocol provides no arbitration for the symbol window. The network idle time is used to calculate clock adjustments and correct the node's view of the global time.

Clock synchronization consists of two main concurrent processes. The macrotick generation process (MTG) controls the cycle and macrotick counters as well as applies the rate and offset correction values. The clock synchronization process (CSP) performs initialization at cycle start, measurement and storage of deviation values, and calculation of the offset and the rate correction values.

D. TTP/C

The TTP/C protocol (Time Triggered Protocol, class C) [9] is an integral communication protocol for time-triggered architectures, designed to support the interconnection of electronic modules (nodes) of distributed fault tolerant real-time systems with stringent dependability requirements.

In TTP/C communication is organized in TDMA rounds. A TDMA round is divided into slots. Each node in the communication system has its sending slot and must send frames in every round. The frame size allocated to a node can vary from 2 to 240 bytes in length, each frame usually carrying several messages. The cluster cycle is a recurring sequence of TDMA rounds; in different rounds, different messages can be transmitted in the frames, but in each cluster cycle the complete set of state messages is repeated.

All nodes are aware of which node has access to the bus during a specified time slot, given the a priori scheduling allocation. By noting the time when messages are received from other nodes (TTP/C is a broadcast protocol, so all nodes receive all messages) with the known schedule, a node can calculate the difference between the clock of the sending node and its own clock.

E. TTEthernet

Time-Triggered Ethernet (TTEthernet) [10], [11] was developed by TTTech Computertechnik AG as an industrial protocol. TTEthernet is fully compliant with standard Ethernet and transparently integrates real-time traffic with standard-Ethernet traffic.

The clock synchronization of TTEthernet establishes a system-wide global-time base in a TTEthernet network. TTEthernet defines a dedicated synchronization message as the Protocol Control Frame (PCF) [10]. A PCF contains accumulated time information regarding its passing from a sender to a receiver. TTEthernet end systems or switches use the information of a PCF to calculate its time correction value.

In a TTEthernet system, the system-wide global-time base is performed on the cluster cycle specified in the common time-triggered (TT) communication schedule. After the local clock time of each TTEthernet device is already synchronized with the system-wide global-time base of a TTEthrenet system, the TTEthernet device transmits and receives synchronization messages and TT messages according to the TT communication schedule. The TT communication schedule is derived from scheduling TT traffic into the cluster cycle. The cluster cycle is an interval of time containing events of time-triggered message transmission. All transmission instants of

real-time applications in a real-time cluster are assigned in the cluster cycle using two terms: period and offset. The offset of a real-time application in the cluster cycle denotes a point in time which is relative to the cluster cycle reference.

F. TSN

Time-Sensitive Networking (TSN) [12] is a set of IEEE 802 Ethernet sub-standards that are defined by the IEEE TSN task group. These standards enable fully deterministic real-time communication over Ethernet. TSN achieves determinism over Ethernet by using a global sense of time and a schedule which is shared between network components.

Clock synchronization is a vital mechanism for achieving deterministic communication with bounded message latency in TSN. The Precision Time Protocol (PTP, IEEE 1588) uses physical layer timestamps to compute network delays and define synchronization events. For TSN systems a 1588 profile was developed (IEEE 802.1AS). It defines fewer options, but extends some physical layer options.

In the core of Time-Sensitive Networking is a time-triggered communication principle. In TSN this concept is known as the "time-aware shaper" (TAS), which deterministically schedules traffic in queues through switched networks. It is being standardized as IEEE 802.1Qbv. With the time-aware shaper concept it is possible to control the flow of queued traffic from a TSN enabled switch. Ethernet frames are identified and assigned to queues based on the priority field of the virtual local area network (VLAN) tag. Each queue is defined within a schedule, and the transmission of messages in these queues is then executed at the egress ports during the scheduled time windows. Other queues will typically be blocked from transmission during these time windows, therefore removing the chance of scheduled traffic being impeded by non-scheduled traffic.

G. Profinet IO IRT

Profinet IO IRT (Isochronous Real-Time) [13] is an Ethernet-based hard real-time communication protocol, which uses static schedules for time-critical data.

The time-triggered communication relies on the ability of individual nodes to send the respective messages exactly at the predetermined time instants. To be able to accomplish this, the nodes' clocks must be synchronized with such a precision that allows the jitter of the communication-cycle length to be as low as possible. Therefore, the Precision Transparent Clock Protocol (PTCP) [14] is used. PTCP synchronizes the clocks of the network nodes by broadcasting periodically the synchronization frames, which are forwarded from the elected synchronization master till covering the whole network. In a typical scenario, the synchronization frames are sent in every communication cycle.

Profinet IO IRT uses static schedules for time-critical data to fulfill the requirement on the timeliness of the data delivery. The individual-node schedules are downloaded into the nodes, each of which contains a switch. Thus, a special hardware (switch), unlike in the traditional Ethernet, is required to be able to accept the respective schedule. Part of the communication cycle (send clock) is reserved for the IRT communication in which the deterministic message frames are sent.

H. SpaceFibre

SpaceFibre [15] is a very high-speed serial link designed specifically for use onboard spacecraft. It aims to complement the capabilities of the widely used SpaceWire onboard networking standard: improving the data rate by a factor of 10 (2Gbit/s), reducing the cable mass and providing galvanic isolation. Multi-laning improves the data-rate further to well over 20 Gbits/s. SpaceFibre provides a coherent quality of service mechanism able to support best effort, bandwidth reserved, scheduled and priority based qualities of service. It substantially improves the fault detection, isolation and recovery (FDIR) capability compared to SpaceWire.

Scheduled quality of service provides a means of ensuring fully deterministic allocation of SpaceFibre network resources. Time is separated into a series of time-slots during which a virtual channel can be scheduled to send data. Time for use in the scheduled quality of service shall be taken from the local time register, which is regularly updated by the time-distribution broadcast channels. When a time-slot arrives in which a virtual channel is scheduled, it can send data based on its precedence. During all the other time-slots, when the virtual channel is not scheduled to send data, it is not permitted to send any data even when no other virtual channel has data to send. Several virtual channels can be scheduled to send data in the same time-slot. In this situation medium access controller sends data from the virtual channel with the highest precedence.

I. SpaceWire-D

SpaceWire-D protocol [16] is a standard prototype that provides deterministic data delivery over SpaceWire networks. It uses the RMAP protocol for transferring information over the SpaceWire network. The main feature of SpaceWire-D is a scheduling mechanism. SpaceWire-D divides the system operation time into a number of timeslots. A constant number of time-slots composes one epoch.

The receipt of a time-code by a node indicates the start of a time-slot. The time-slot number shall be the same as the time-value of the time-code that indicates the start of a time-slot. The end of a time-slot shall be normally indicated by the arrival of the next time-code.

SpaceWire-D uses a time-code watchdog timer that should be kept in each initiator node to check for the correct arrival of each time-code, early or late arrival of a time-code. In the event of an early or late time-code the initiator shall flag an error to the user application.

A schedule is defined by a schedule table for each initiator node. This table specifies in which time-slot the corresponding initiator is allowed to send RMAP commands. Each initiator node holds a copy of the schedule table. The initiator node may start the transmission of data to any target device during a time-slot in which it is scheduled to initiate RMAP transactions.

J. STP-ISS revision 2

STP-ISS [17], [18] is the transport layer protocol that describes informational and logic interaction between onboard devices, packets' formats and packet transmission rules for the SpaceWire network. STP-ISS transport protocol is now represented in two revisions. The first revision of STP-ISS is much simpler and compact, but the second one is more powerful. STP-ISS rev.2 provides connection-oriented and connectionless data transmission between the nodes of the network with priorities, guaranteed delivery, best effort and scheduling quality of service types. The transmitter side has three logical buffers for each type of packet (control commands, urgent messages and common messages). The receiver side has two logical buffers: for connectionless and connection-oriented traffic. In addition, there is a flow control mechanism implemented for the connection-oriented transmission with the guaranteed quality of service. STP-ISS protocol provides the duplicate control commands detection in receiver and gives ability for data resending in case of error detection in the received data. Thus, STP-ISS provides reliability, guaranteed services and scheduling.

III. SCHEDULING PROTOCOL

Modern space industry demands a protocol running over SpaceWire, which can provide deterministic data transmission characteristics [2]. The basic SpaceWire standard covers three bottom layers of the OSI model and does not provide transport services [19]. Nowadays, there is a number of transport protocols intended to operate over SpaceWire. They are: RMAP, CCSDS PTP, STP-ISS, STUP, JRDDP, SpaceWire-R, STP and SpaceWire-D, which uses RMAP. Each of them is intended to solve its particular tasks and, in some cases, there are two or more transport protocols operating in one network. Moreover, a single node can implement several transport protocols running over SpaceWire (for example, RMAP, CCSDS PTP, STP-ISS). Traffic from different transport protocols can interfere especially while getting access to the SpaceWire link in a node. It is rather difficult to avoid conflicts with simultaneous network resource usage, thus we cannot provide deterministic delivery of data in the network.

Consequently, there is not only an issue of schedule creation and synchronization between the nodes but also an issue of arbitration of different transport protocols data flows. For the SpaceWire networks only SpaceWire-D protocol deals with an issue of scheduling in SpaceWire networks. However, SpaceWire-D was not designed for scheduling traffic from several transport protocols as it gets data directly from the Application Layer. Moreover, SpaceWire-D utilizes RMAP to communicate over the network, which imposes restrictions and affects on its flexibility in use.

Therefore, it was decided to develop new scheduling protocol which solves all abovementioned problems. Fig. 1 shows the place of the scheduling protocol in the protocol stack and its comparison with the OSI reference model [20].

As a basis of this new protocol we took the STP-ISS scheduling mechanism as it best suites all stated tasks. STP-ISS scheduling mechanism is based on SpaceWire time-codes distribution provided by the SpaceWire standard. Let us now consider the existing STP-ISS scheduling mechanism and then enhance it with additional functionality for operation with several transport protocols simultaneously.

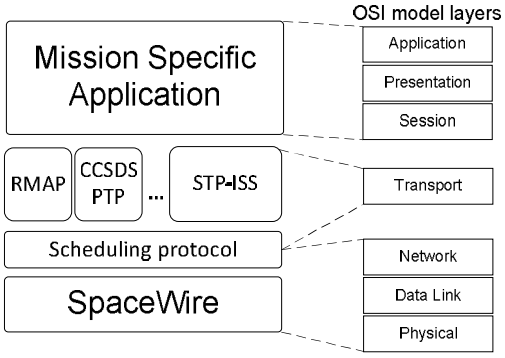

Fig. 1. Comparison of the scheduling protocol with OSI

A. SpaceWire time synchronization

The global time ticks of each node must be periodically resynchronized within the ensemble of nodes to establish a global time base [9]. The STP-ISS rev.2 scheduling mechanism is based on a SpaceWire time-codes broadcasting mechanism. These time-codes contain a six-bit value of system time. Each node and switch has its internal six-bit time counter. There is one node or switch in a network, which is set as a time-master. It is responsible for time distribution over a network. When the time master receives a tick from a host-system, it should increment its time counter and send new time value in a time-code. When a node or a switch receives a time-code, it should update its internal time counter with the received time value. This new value should be one more than the time-counter's previous time value. If the received time-code value is equal to internal counters value, then tick out signal should not be emitted. When a switch receives a time-code with time value, which is one more than the internal counter's value it increments the counter value and emits a tick signal. This tick signal propagates to all the output ports of the switch so that they emit the time-code. When switch receives a time-code with a time-code value that is equal to the internal counter value, then it is ignored. This helps to prevent circular time-codes propagation. This is the way the time-codes are used to synchronize all the network nodes with the time master's clock [19].

B. STP-ISS scheduling mechanism

According to this scheduling mechanism, there is a single schedule for the whole SpaceWire network. It gives an opportunity for the node to send data only during particular time-slots. The schedule and time-slot duration are set during the configuration phase and are stored in each end-node of the network. The time-slot timer (T_{TS}) counts duration of the current time-slot for a particular node. Synchronization according to a scheduling mechanism is performed once in an epoch. An epoch is a constant number of time-slots. For example, an epoch can consist of 10, 20, 64 or more time-slots, but it should contain at least 2 time-slots. The scheduling table describes one epoch.

The number of time-slots in one epoch should be defined during the configuration phase and should be set to the time-

slots counter C_{TS} value. The time-slot duration DTS should be set to the time-slot timer T_{TS}.

The epoch duration D_E is calculated the following way:

$$D_E = D_{TS} * C_{TS} \quad (1)$$

If the time-slot duration D_{TS} value changes, the epoch duration value D_E should be calculated and updated.

Each node is permitted to send packets at a particular time-slot in accordance with the schedule. At the end of its time-slot, the node should stop the data transmission. However, the transmission stops only after the current packet is transmitted to the network. If any other node has data for transmission, but it is not scheduled for transmission at the current time-slot, then this node should wait for its time-slot.

STP-ISS protocol defines the time-code relevancy window, that shows if the received time-code is relevant or not. This parameter defines a number of time-slots in the end of the epoch and in the beginning of the next epoch. During these time-slots a received time-code is considered as relevant (K is time-code relevancy window size). The time-code relevancy window is shown in Fig. 2. Time-code relevancy window is a configuration parameter and should be set during configuration phase. It could be defined individually for each node in accordance with the accuracy of local clocks.

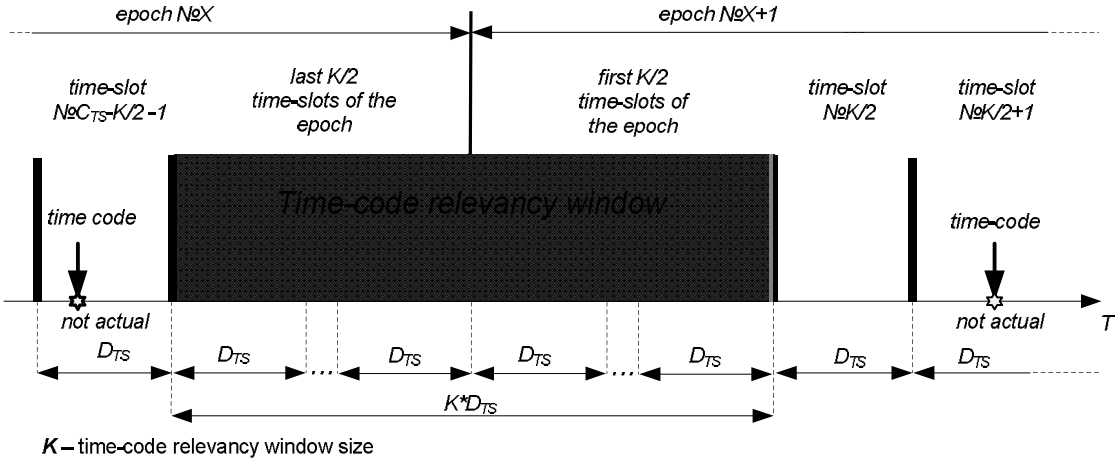

Fig. 2. Time-code relevancy window

The time-slot timer expiration in the last time-slot of an epoch and reception of a relevant time-code indicate the beginning of a new epoch, in which the time-slot counter C_{TS} will count time-slots starting from zero. When the node gets the time-code, it does not analyze the time-code number. The beginning of a new epoch is associated with the fact of the time-code reception.

There are two possible synchronization cases, which can occur:

- the next time-code is received during first *K/2* time-slots of the epoch;
- the next time-code is received during last *K/2* time-slots of the epoch.

Considering the node functionality, the abovementioned cases mean that the internal time-slot timer and the time master are not synchronized. This means that the node should start the synchronization process.

Fig. 3 shows the case, when a node started a new epoch and the expected time-code is received during first *K/2* time-slots of the new epoch. In this case, the node should terminate time-slot timer T_{TS} and calculate new value for the time-slot duration. The D_{TS_new} value is calculated according to the equation (2):

$$D_{TS\_new} = D_{TS} + \frac{\Delta t}{C_{TS}}, \quad (2)$$

where Δt is the current value counted since the beginning of the epoch.

Subsequently, the node updates the epoch duration value according to the equation (3).

$$D_E = D_{TS\_new} \cdot C_{TS} \quad (3)$$

The newly calculated value will be applied to the T_{TS} timer for the next time-slot.

Let us consider the second case when the time-code is received during the last *K/2* time-slots of the epoch (see Fig. 4). In this case, the node should terminate the current epoch and calculate new values for the time-slot timer. For this purpose, the node takes the current Δt value counted since the beginning of the epoch and calculates the new time-slot duration according to the equation (4):

$$D_{TS\_new} = \frac{\Delta t}{C_{TS}} \quad (4)$$

The next time-slot starts with the new T_{TS} timer value D_{TS_new}.

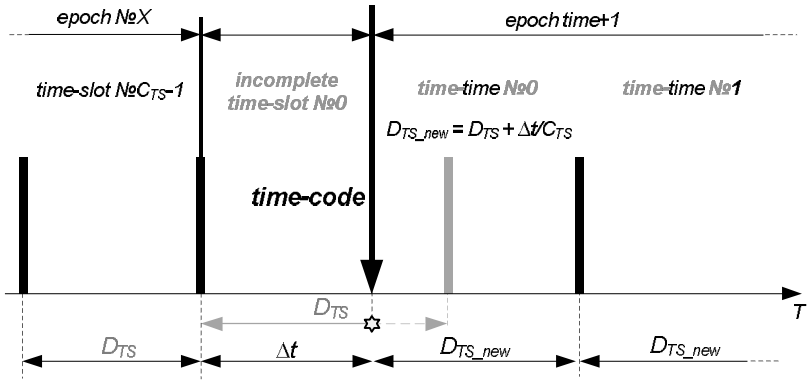

Fig. 3. Time-slot timer value correction (the time-code received during the first time-slot of the epoch

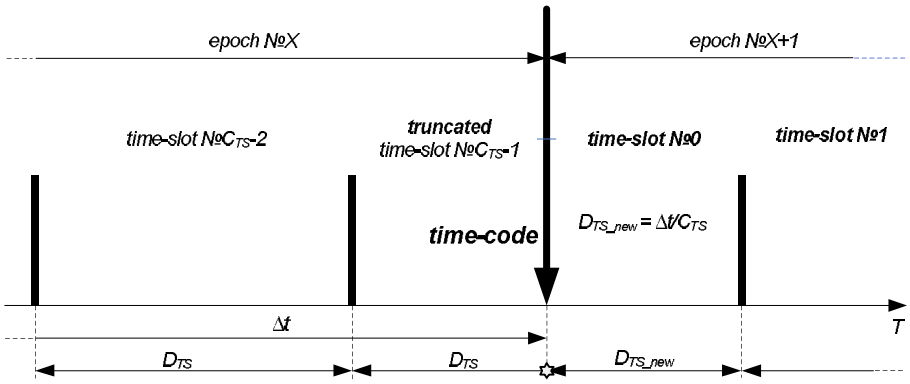

Fig. 4. Time-slot timer value correction (the time-code received the last time-slot of the epoch)

If the epoch timer expires simultaneously with the time-code reception, then there is no need to correct the epoch timer value. The moment of the epoch timer expiration and the time-code reception is determined depending on the implementation. These events are not strictly simultaneous in the hardware. So there is some gap between these events that could be considered as satisfactory or not. Also this gap can be useful to take into consideration the accuracy or jitter of the time-code reception.

The STP-ISS protocol should count a number of received irrelevant time-codes. Reception of three irrelevant time-codes means that the internal time-slot timer and the time master are significantly asynchronous. In this case, it is necessary to synchronize with the time master. Reception of the third irrelevant time-code should determine the beginning of a new epoch. The node should terminate the time-slot timer and wait for reception of the next time-code. In this new epoch the node should not send data until reception of the next time-code. After reception of a time-code the node should update the time-slot duration value and then continue data transmission according to the schedule. New time-slot duration value should be calculated according to the equation (5):

$$D_{TS\_new} = \frac{\Delta t}{C_{TS}}, \quad (5)$$

where Δt is the time value, that is counted starting from the moment of third irrelevant time-code reception and finishing with the next time-code reception.

C. New scheduling protocol

The described above STP-ISS scheduling mechanism, extended with additional features for operation with several transport protocols simultaneously, results in a new scheduling protocol for SpaceWire networks. This new scheduling protocol should be implemented not only in the end-nodes (as it is designed in STP-ISS) but also in switches which are directly connected to the nodes. Let us now consider in more details a new scheduling protocol implementation in nodes and switches.

1) Nodes: Generally, there is a schedule for the whole SpaceWire network defining in which time-slots a particular node can send data. An example of such scheduling table is given in Table I. This scheduling table should be stored in each node and switch and should be designed in such a way that an access to shared resources (e.g. output switch ports, links, etc.) is multiplexed in time. Time synchronization mechanism was taken from the STP-ISS without any modifications.

TABLE I. SCHEDULING TABLE FOR THE NETWORK

Node	Time-slots																				
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0				■	■		■			■			■			■			■		■
1	■		■			■		■			■			■			■			■	
2		■			■				■	■			■		■	■			■		
⋮																					
25		■	■			■			■						■	■				■	

According to the scheduling protocol a node can only send data in the specified time-slots. However, if there are several transport protocols operating in one node it is necessary to somehow share this time-slot between them. There are two main approaches for solving this issue in nodes:

- allocate full time-slot for a particular transport protocol;
- priority arbitration between concurrent transport protocols.

According to the first approach, an entire time-slot is assigned for transmission of data of a particular transport protocol. It gives an opportunity for the certain transport protocol on the certain node to send data only during a particular time-slot. In such case all other protocols, which operate in the node should wait for their time-slots. An example of the scheduling table for this approach is shown in the Table II.

TABLE II. SCHEDULING TABLE FOR NODES AND TRANSPORT PROTOCOLS

Node	Protocol	Time-slot																			
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	STP-ISS										■						■					
	RMAP					■								■						■		
	PTP				■			■														■
1	STP-ISS	■							■						■							
	RMAP											■						■				
	PTP			■			■														■	
2	STP-ISS		■							■						■						
	RMAP										■						■					
	PTP					■								■						■		
⋮																						
25	STP-ISS			■			■														■	
	RMAP		■							■						■						
	PTP										■						■					

The second concept assumes that during a time-slot all transport protocols operating in the node have an opportunity to send their data. Arbitration is performed by the scheduling protocol basing on priorities assigned to transport protocols. If several transport protocols have data for sending in the allowed time-slot, then the protocol with the higher priority will send data first. This approach gives an opportunity to effectively utilize the network capacity and, moreover, is easier in implementation as it does not need to perform additional scheduling between transport protocols.

The new scheduling protocol does not contain any data buffers: it passes transport protocol data directly to SpaceWire without keeping it inside the scheduler. In the allowed time-slot the scheduler arbiter analyses whether transport protocols have data to send or not. Depending on the implemented approach (scheduling or priorities) the arbiter makes a decision which transport protocol can transmit data. Then, the particular transport protocol can pass its data packet to the scheduling protocol for sending over the network.

2) Switches: SpaceWire-D protocol is intended to be implemented in the SpaceWire nodes only, but the new scheduling protocol would be implemented in SpaceWire switches also. In this scheduling protocol we propose to use the port guardian mechanism, which is described in [1]. The following mechanism should be implemented in switches, which are directly connected to the nodes. This mechanism is optional and it can be switched off for a particular switch.

Most of network technologies, described above, suppose port guardian mechanism in order to protect the network from nodes that try to transmit data at inappropriate time-slots. Often such a "watch dog" is implemented as a separate device or chip in order to increase fault tolerance. Port guardian guarantees that the node would not transmit data during wrong time-slots and eliminates «babbling idiot» problem.

SpaceWire switches store the scheduling table (see Table I), but it is simpler than for the nodes (Table II). Switch does not know which protocol sends the packet, but it should be able to block the packet transmission, if the node is not scheduled for data transmission in the current time slot. Switch's scheduling table identifies, which port is allowed to send data during a specific time-slot. Thus, according to this schedule, the switch can determine whether the packet should be discarded or it can be routed to the outgoing port. If the leading byte of the SpaceWire packet was received in the time-slot, in which the node is allowed to send data, then this packet can pass through the router. Otherwise, this packet should be discarded.

The Fig. 5 shows a SpaceWire network with network switches, marked as «Net guard», which store a scheduling table and permit data transmission. The switch, which is not connected to nodes is a standard SpaceWire switch, which is not able to analyze the scheduling.

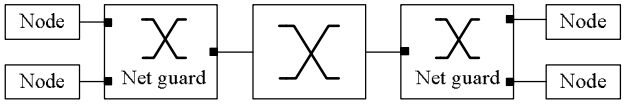

Fig. 5. Network with network guardians

IV. CONCLUSION

In the current paper we proposed the way for implementation of a protocol that will provide the determinism to SpaceWire networks, where multiple transport protocols are operating. For this purpose we overviewed communication protocols, which use Time-Division Multiplexing (scheduling) concept to provide the deterministic data delivery. We analyzed the scheduling mechanisms and decided to take the STP-ISS scheduling mechanism as the basis for a new protocol. Also we proposed some improvements and additions to this mechanism that increase the quality of deterministic data delivery. This mechanism can prevent network resources usage conflicts and increase the network bandwidth.

In addition, we proposed the network guarding mechanism that can improve fault-tolerance of a deterministic SpaceWire network by using the proposed scheduling protocol in SpaceWire switches. These Net guards eliminate «babbling idiot» problem and protect the network from the nodes, which are not synchronized with the time-master.

The proposed protocol is still in the development process. It will solve the serious problem that currently is one of the most important tasks for the space industry.

ACKNOWLEDGEMENT

The research leading to these results has received funding from the Ministry of Education and Science of the Russian Federation under the contract RFMEFI57814X0022.

REFERENCES

[1] I. Korobkov, E. Podgornova, D. Raszhivin, V. Olenev, I. Lavrovskaya "Scheduling Mechanisms for SpaceWire Networks", *Proceedings of 16th Conference of Open Innovations Association Finnish-Russian University Cooperation in Telecommunications (FRUCT) Program*; Russia, Yaroslavl, 2015. pp. 82-88.

[2] V. Olenev, I. Lavrovskaya, I. Korobkov, D. Dymov, "Analysis of the Transport Protocol Requirements for the SpaceWire On-board Networks of Spacecrafts", *Proceedings of 15th Seminar of Finnish-Russian University Cooperation in Telecommunications (FRUCT) Program*; Saint-Petersburg: Saint-Petersburg University of Aerospace Instrumentation, 2014. pp. 65-71.

[3] S. Balandin, A. Heiner, "Dynamic Localized Load Balancing", *SPIE Volume 5244: Performance and Control of Next-Generation Communications Networks (ITCom2003)*, USA, September 2003, pp. 164-175.

[4] Fuehrer, T., Mueller, B., Hartwich, F., and Hugel, R., "Time Triggered Communication on CAN (Time Triggered CAN-TTCAN)," SAE Technical Paper 2001-01-0073, 2001.

[5] F. Hartwich, T. Führer, R. Hugel, B. Müller, Robert Bosch GmbH, "Timing in the TTCAN Network"; in Proc. of the International CAN Conference; 2002.

[6] J. Berwanger, M. Peller, R. Griessbach, "Byteflight – A New Protocol for Safety Critical Applications" *in Proc. of the FISITA world Automotive Congress*, 2000.

[7] Flexray. FlexRay Communications System Protocol Specification Version 3.0.1. Specification, FlexRay Consortium, 2005.

[8] A. Hanzlik, "A Case Study of Clock Synchronization in FlexRay", Research Report 31/2006 Technische Universitat Wien, Institut fur Technische Informatik, 2006.

[9] H. Kopetz, "Real-Time Systems. Design Principles for Distributed Embedded Applications", Kluwer Academic Publishers, Boston, 1997.

[10] A. Ademaj, H. Kopetz, P. Grillinger, et al.,"Fault-Tolerant Time-Triggered Ethernet Configuration with Star Topology" *in Proc. 19th International Conference on Architecture of Computing Systems*, 2006.

[11] H. Kopetz, A. Ademaj, P. Grillinger, and K. Steinhammer, "The Time-Triggered Ethernet (TTE) Design," *in Proc. 8th IEEE International Symposium on Object-oriented Realtime distributed Computing (ISORC)*, Seattle, 2005.

[12] TTTech Computertechnik AG, IEEE TSN (Time-Sensitive Networking): A Deterministic Ethernet Standard, Web: https://www.tttech.com/download/technologies/deterministic-ethernet/time-sensitive-networking/file/1c49796102083798d7f3968fa4c3c2ae3b59a471/.

[13] Z. Hanzalek, P. Burget; P. Sucha, "Profinet IO IRT Message Scheduling With Temporal Constraints", *IEEE Transactions on Industrial Informatics*, 2010, pp. 369-380

[14] D. Fontanelli, D. Macii, S. Rinaldi, P. Ferrari, and A. Flammini, "Performance analysis of a clock state estimator for Profinet IO IRT synchronization,", *Instrumentation and Measurement Technology Conference (I2MTC)*, 2013 IEEE International, May 2013, pp. 1828-1833.

[15] S. Parkes, A. Ferrer-Florit, A. Gonzalez. and C. McClements - *SpaceFibre Draft H1*: Space Technology Centre, University of Dundee, 2013.

[16] S. Parkes and A. Ferrer-Florit, *SpaceWire-D – Deterministic Control and Data Delivery Over SpaceWire Networks*, Draft B. April 2010.

[17] Y. Sheynin, V. Olenev, I. Lavrovskaya, I. Korobkov, S. Kochura, S. Openko, D. Dymov "STP-ISS Transport Protocol Overview and Modeling", *in Proc. of 16th Conference of Open Innovations Association Finnish-Russian University Cooperation in Telecommunications (FRUCT) Program*; Oulu: University of Oulu, 2014. pp. 185-191.

[18] Y. Sheynin, V. Olenev, I. Lavrovskaya, I. Korobkov, D. Dymov "STP-ISS Transport Protocol for Spacecraft On-board Networks", *Proceedings of 6th International SpaceWire Conference 2014 Program*; Greece, Athens, 2014. pp. 26-31.

[19] ESA (European Space Agency). Standard ECSS-E-50-12C, Space engineering. SpaceWire – Links, nodes, routers and networks. European cooperation for space standardization. Noordwijk: ESA Publications Division ESTEC, 2008.

[20] A.S. Tanenbaum, *Computer Networks*, Fifth Edition; Prentice Hall, 2011.