Development of the Transport Layer Scheduling Mechanism for the Onboard SpaceWire Networks

Valentin Olenev, Elena Podgornova, Irina Lavrovskaya, Ilya Korobkov, Nadezhda Matveeva
Saint-Petersburg State University of Aerospace Instrumentation
Saint Petersburg, Russia
{valentin.olenev, alena.podgornova, irina.lavrovskaya, ilya.korobkov, nadezhda.matveeva}@guap.ru

***Abstract*—Scheduling becomes an important mechanism for the onboard networks. It gives an ability to control the traffic and network capacity. This paper gives an overview of the development of the new transport layer scheduling mechanism for the SpaceWire onboard networks and presents the scheduling mechanisms modeling results. In addition, the paper compares the developed scheduling mechanism with the one that was adapted from the SpaceWire-D transport protocol. The results of this research will be used for the development of a new transport protocol operating over SpaceWire.**

I. Introduction

Modern onboard networks need to schedule the traffic to avoid conflicts with simultaneous network resource usage and to reduce the network capacity. There are different scheduling mechanisms for packet-switching networks and IP networks. In general, they can be divided into two classes: reactive (or feedback control schemes) and proactive (or resource reservation algorithms) [1], [2]. Most of them are implemented by means of store-and-forward switches. Unfortunately, these mechanisms are not suitable for SpaceWire networks. According to SpaceWire official standard [3], switches support wormhole routing without full packet buffering. Analysis of the transport protocols presented in [4] showed that there is no transport layer protocol officially released for the SpaceWire network that provides scheduling quality of service.

The SpaceWire-D [5] protocol prototype has such a scheduling mechanism. Nevertheless, this protocol is not available for the public. Therefore, in this paper we took the scheduling mechanism from the SpaceWire-D, simulated it and made some improvements. As a result, we evaluated a new scheduling mechanism which is more efficient than SpaceWire-D mechanism. The modeling and protocols comparison was made by means of SystemC network model [6], [7].

II. SpaceWire time synchronization

In order to implement the scheduling mechanism a network should be able to synchronize the schedule among all the nodes. SpaceWire provides a mechanism for broadcasting time values via special control codes – time-codes [3]. These time-codes contain a six-bit value of system time. Each node and router has its internal six-bit time counter. There is one node or router in a network, which is set as a time master. It is responsible for time distributing 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 router 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 router 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 router so that they emit the time-code. When router receives a time-code with a time-code value that is equal to the internal counter value, then this time-code 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 [3].

III. Scheduling mechanism based on the SpaceWire-D

We took the scheduling mechanism from the SpaceWire-D and adapted it for our needs. This adapted mechanism is supposed to be introduced to the new transport protocol.

According to this adapted scheduling mechanism, there is a 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 node. The time-slot timer (T_{TC}) counts duration of the current time-slot for a particular node.

The adapted scheduling mechanism implies time synchronisation for each time-slot. A new time-slot begins if a time-slot timer expired and a new time-code from a SpaceWire network is received. If the time-code is lost then the node ends the current time-slot on time-slot timer

ISSN 2305-7254

expiration. When the next time-code is received, the node should continue its functioning in conformance with the schedule. In this case functioning continues from the time-slot with the same number that has been received in the time-code.

There are two possible synchronization cases, which can occur:

- the next time-code is received after the time-slot timer expiration;
- next time-code is received before the time-slot timer expiration.

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. 1 illustrates the situation when the next time-code is received after the time-slot timer expiration. In order to perform synchronization, the node calculates a new value of time-slot timer and sets it to the T_{TC}. For this purpose, the node saves the current T_{TC} value into Δt and restarts the time-slot timer T_{TC}. When T_{TC} timer expires timer value should be set to $T_{TC} + \Delta t$. As a result, the next time-slot timer starts with a new value.

Fig. 2 shows the situation, when a new time-code is received before expiration of T_{TC} timer. In this case, the node terminates the current time-slot and saves the current T_{TC} value into Δt. The next time-slot starts with the T_{TC} timer value equal to Δt.

If the time-slot timer expires simultaneously with the time-code reception, then there is no need to correct the time-slot timer value. Ideally, the difference between the moments of time-code reception and T_{TC} timer expiration should be 0 ns. This means that the time-slot timer of a node works synchronously with the SpaceWire time master.

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.

IV. SystemC model of the scheduling mechanism

We implemented a SystemC model of the SpaceWire network in order to check the correctness and efficiency of scheduling mechanisms. It gave us important results for the analysis and comparison of these mechanisms.

The SystemC model consists of a number of modules. Each module has its own functionality. The model is implemented in SystemC language.

The SystemC model consists of two main modules:

- Node model:
  - Trafgen module;
  - QoS module;
- Switch model.

The architectural diagram of the simplified SpaceWire network model is shown in Fig. 3. The structure of the model is given in comparison with OSI model [8].

A. Node model description

The node model consists of two modules – trafgen and QoS. The traffic generator (Trafgen) is a model of an application layer. It is responsible for generation and sending different packet sequences to the transport layer for testing and modeling purposes. After generation of a packet, the Trafgen transfers it to the QoS module of the same node.

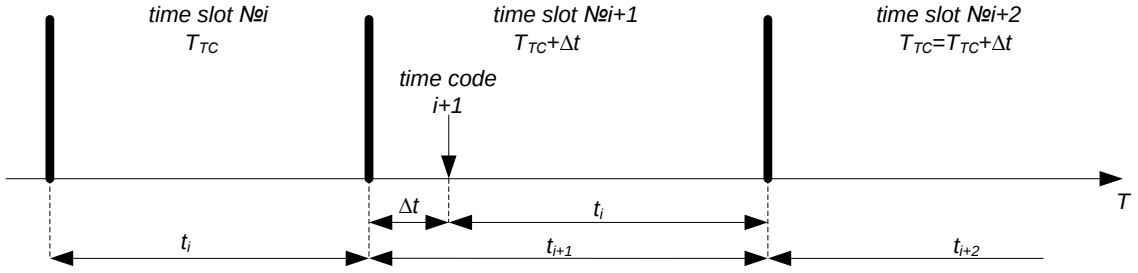

Fig. 1. Next time-code is received after the time-slot timer expires

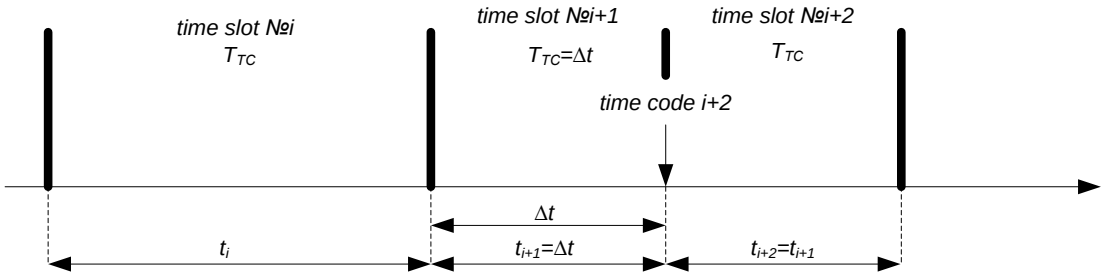

Fig. 2. Next time-code is received before the time-slot timer expires

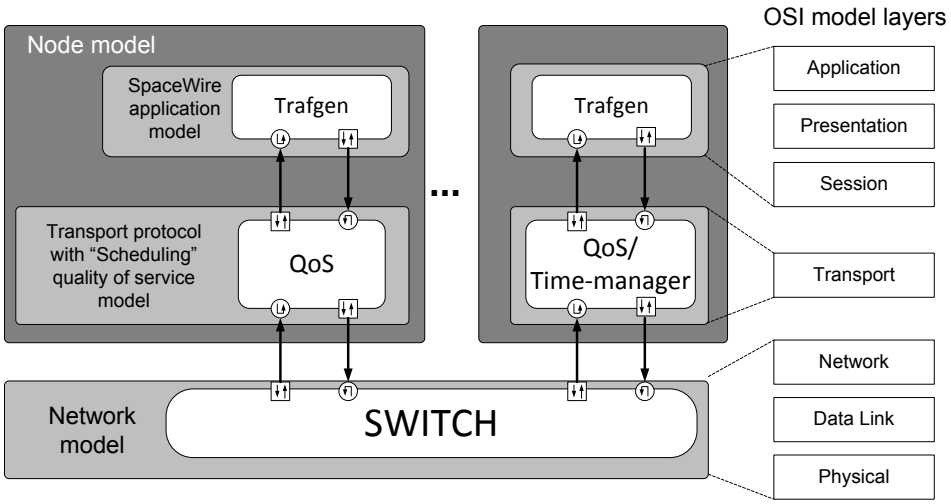

Fig. 3. The simplified SpaceWire network model

B. Scheduling mechanism model description

The QoS module is responsible for implementation of the scheduling mechanism. This module performs the following actions:

- packets reception from the Trafgen;
- data transmission control in accordance with the defined speed, priority and schedule;
- data transfer to the switch block;
- timer synchronization according to the scheduling mechanism;
- writing results to the log file.

According to the scheduling mechanism, one of the nodes is a SpaceWire time master. At the beginning of each time-slot this node sends time-codes with time-slot numbers to the SpaceWire network.

C. Switch model description

The switch model is a simple implementation of the SpaceWire switch that is in order for the packet switching.

This model is responsible for:

- data reception from QoS module;
- wormhole routing;
- sending data to QoS model of the remote node in accordance with the defined speed and priority;
- updating of the saved time-slot number.

V. Adapted Scheduling Mechanism Simulation Results

An example of network topology that we used for the scheduling simulation is shown in Fig. 4. Nodes of the SpaceWire network have numbers from 0 to 23; switches have numbers from 0 to 3. Nodes and switches are connected via SpaceWire channels. Node 0 is a SpaceWire time master (TM).

Simulation of the model lasted for 64 milliseconds with data transmission speed 200 Mbit/s. Time-slot duration in this model is equal to 500 microseconds. Packet size is 1 Kbyte, including 1 byte for destination address, 1022 bytes for data, 1 byte for end of packet (EOP) symbol.

We used the following schedule table and data transmission directions for simulation:

- nodes 0, 1, 9, 10, 12, 19 and 20 send data to 11;
- nodes 4, 5, 6, 13, 14, 21 and 22 send data to 23;
- nodes 2, 3, 7, 8, 15, 16 and 18 send data to 17.

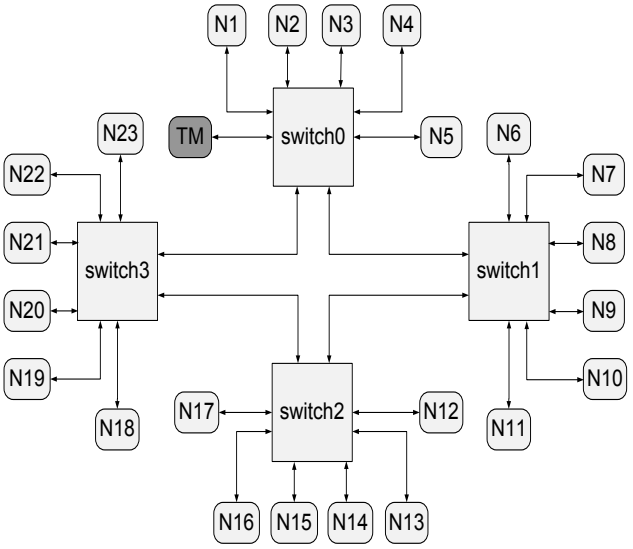

Fig. 4. SpaceWire network topology

Schedule for data transmission is shown in Table I. Time-slots marked with the black color define the time-slot number allocated for the particular node.

Simulation results are summarized in Table II and in a diagram in Fig. 5.

TABLE I. SCHEDULE TABLE

Slot	Nodes																							
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
0										■						■						■		
1					■								■						■					
2				■			■														■			
3	■							■						■										
4										■						■							■	
5			■			■														■				
6		■							■						■									
7										■						■						■		
8					■								■						■					
9				■			■														■			
⋮																								
61			■			■														■				
62		■							■						■									
63										■						■						■		

Fig. 5 clearly shows that in case of simulation without scheduling mechanism, the number of packets sent by nodes is non-uniformly distributed. The adapted scheduling mechanism provides data transmission with more uniform distribution.

Simulation results prove that the scheduling mechanism gives ability to uniformly distribute the network traffic among all nodes of the network.

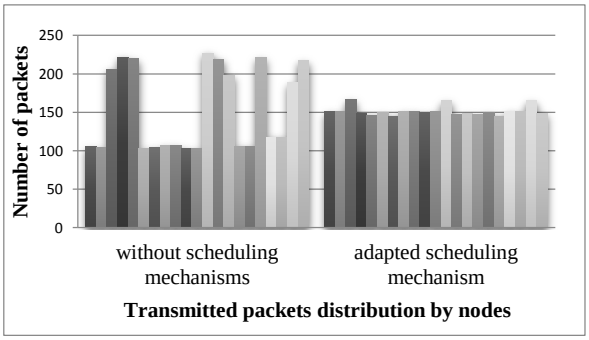

Fig. 5. Comparison of data transmission results with scheduling mechanism and without it

TABLE II. ADAPTED MECHANISM SIMULATION RESULTS

Receiver	Sender	Number of received packets	
		without scheduling mechanisms	adapted scheduling mechanism
node 11	node 0	105	151
	node 1	104	151
	node 9	206	166
	node 10	222	149
	node 12	220	146
	node 19	103	150
	node 20	104	145
node 17	node 2	107	151
	node 3	107	151
	node 7	103	150
	node 8	103	151
	node 15	227	165
	node 16	219	147
	node 18	198	148
node 23	node 4	106	147
	node 5	106	150
	node 6	221	145
	node 13	118	151
	node 14	118	151
	node 21	188	165
	node 22	217	149
Total number of received packets:		3192 packets	3179 packets

However, the adapted from SpaceWire-D scheduling mechanism has two disadvantages. The first one is that the node synchronizes and updates time-slot timer too frequently. There are a number of reasons, why it is not efficient to have such frequent synchronization:

- it requires additional computational resources from a node;
- it requires additional channel capacity;

- it could slow down data transmission (and even block it) because time-codes have higher priority than data;
- some satellite equipment has high-precision clocks that does not need frequent synchronization.

The second disadvantage of the adapted scheduling mechanism is that it depends on a time-code number. According to the SpaceWire-D specification, the total number of time-slots could be 64 only. This causes many problems for the network schedule creation.

Thus, we evaluated a new version of the scheduling mechanism that does not depend on a time-code number and synchronizes less often than once at a time-slot.

VI. New version of scheduling mechanism

Synchronization according to a new 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. Therefore, the schedule table is divided into epochs. Each epoch consists of the same number of time-slots.

The number of time-slots of one epoch should be defined during the configuration phase and should be set to the time-slots counter C_{TC} value. The time-slot duration should be set to the time-slot timer T_{TC}. The epoch duration T_E is calculated the following way:

$$T_E = T_{TC} * C_{TC} \quad (1)$$

If the time-slot timer T_{TC} value changes, the epoch timer value T_E should be calculated and updated.

Similar to the adapted scheduling mechanism, the time-slot timer counts the duration of the current time-slot. The timer expiration means that the current time-slot ends.

Epoch timer expiration and reception of a time-code indicate the beginning of a new epoch, in which the time-slot counter C_{TC} 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.

Let us again consider two possible synchronization cases, which we have already considered for the adapted scheduling mechanism.

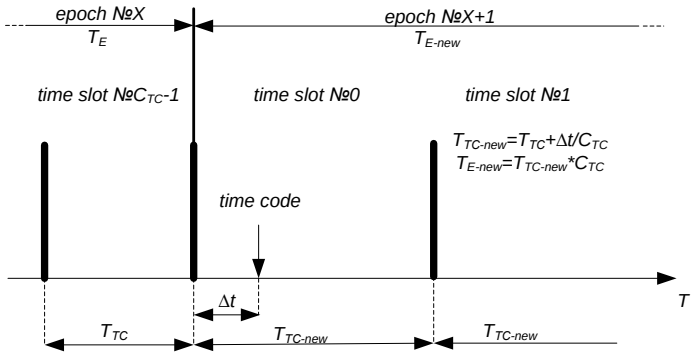

Fig. 6. Time-slot and epoch timers values correction (the time-code received after the epoch timer expired)

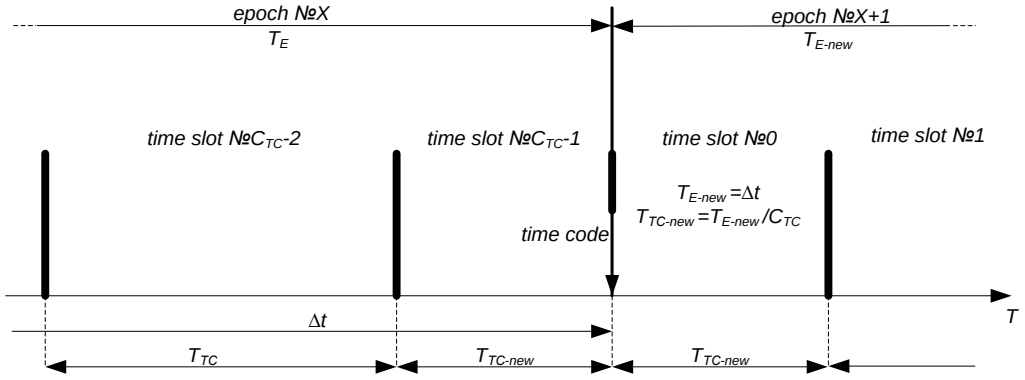

Fig. 7. Time-slot and epoch timers values correction (the time-code received before the epoch timer expired)

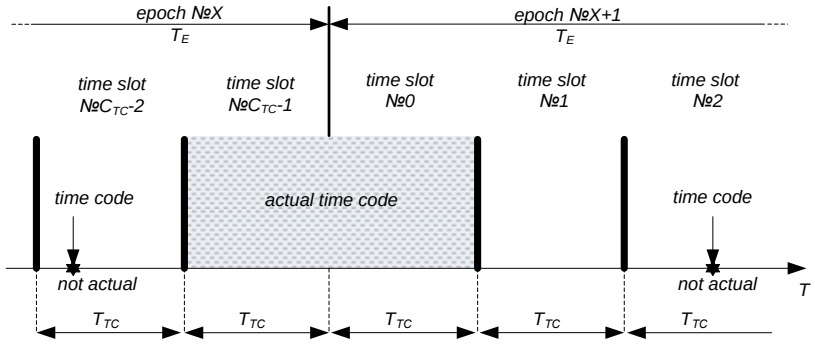

Fig. 8. Time-slot and epoch timers values correcting (received not actual time-code)

Fig. 6 shows the case, when an epoch timer expired and a node started a new epoch. The expected time-code is received during the new epoch's first time-slot. In this case, the node should calculate new values for the time-slot timer and epoch timer and set them to the T_{TC} and T_E correspondently. The T_{TC} value is calculated according to the equation (2):

$$T_{TC} = T_{TC} + \frac{\Delta t}{C_{TC}} \quad (2)$$

where Δt is the current value counted by the time-slot timer.

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

$$T_E = T_{TC} \cdot C_{TC} \quad (3)$$

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

Let us consider the second case. The time-code is received during the last time-slot of the previous epoch, before the epoch timer expiration (see Fig. 7). In this case, the node should terminate the current time-slot and calculate new values for the time-slot and epoch timers. For this purpose, the node takes the current Δt value counted by epoch timer and sets it to T_E . Then the node calculates the new time-slot timer value according to the equation (4).

$$T_{TC} = \frac{\Delta t}{C_{TC}} \quad (4)$$

The next time-slot and epoch start with the new T_{TC} and T_E timers values.

If time-code is received before the last time-slot of the epoch, or after the first time-slot of the epoch, then this time-code is considered as irrelevant and synchronization should not be performed. This situation is showed in Fig. 8.

If the epoch timer expires simultaneously with the time-code reception, then there is no need to correct the epoch timer value.

VII. New scheduling mechanism simulation results

Simulation has been done using the same SystemC model as for the adapted scheduling mechanism. The logic of the model operation was changed according to the previously described mechanism.

Simulation gave results, which are provided in Table III and Fig. 9.

According to the results of simulation, the new scheduling mechanism also provides data transmission with uniform distribution. However, in comparison with adapted scheduling mechanism, the mean-square deviation of packets number decreases from 6.04 to 5.51. The number of transmitted time-codes decreases from 4224 to 34 (for 64 ms modeling), which made possible to transfer more data packets. Therefore, the new scheduling mechanism gives ability to send more packets than scheduling mechanism based on SpaceWire-D during the same simulation period.

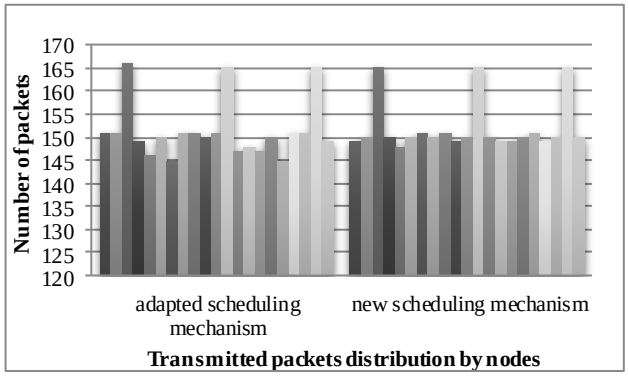

Fig. 9. Comparison of the simulation results for adapted scheduling mechanism modeling and new scheduling mechanism modeling

TABLE III. NEW MECHANISM SIMULATION RESULTS

Receiver	Sender	Number of received packets	
		adapted scheduling mechanism	new scheduling mechanism
node 11	node 0	151	148
	node 1	151	150
	node 9	166	165
	node 10	149	150
	node 12	146	148
	node 19	150	150
	node 20	145	152
node 17	node 2	151	150
	node 3	151	152
	node 7	150	148
	node 8	151	150
	node 15	165	165
	node 16	147	150
	node 18	148	148
node 23	node 4	147	148
	node 5	150	150
	node 6	145	152
	node 13	151	148
	node 14	151	150
	node 21	165	165
	node 22	149	150
Total number of received packets:		3179 packets	3189 packets

VIII. CONCLUSION

This paper described the process of development of the new transport layer scheduling mechanism for the SpaceWire networks. This mechanism was compared with the adapted mechanism from the SpaceWire-D protocol prototype. Comparison was done on the basis of simulation results obtained from the SystemC network model.

The new scheduling mechanism has the following advantages:

1) it prevents conflicts of network resources usage;
2) it decreases the network capacity;
3) it allows to increase network bandwidth;
4) it provides uniform data transmission;
5) it gives more flexibility in schedule creation.

The developed scheduling mechanism is planned to be included into the new transport protocol for the SpaceWire networks.

ACKNOWLEDGEMENT

The research leading to these results has received funding from the Ministry of Education and Science of the Russian Federation according to the base part of the state assignment in 2014.

REFERENCES

[1] H. Zhang *"Service disciplines for guaranteed performance service in packet-switching networks"*, Proceedings of the IEEE, Vol. 83, No.10, October 1995. pp. 1374-1396.

[2] A. La Piana, I. Ferraro, C. Beoni, *"Queuing, scheduling and traffic shaping over IP networks"*, Vehicular Technology Conference, 2004. VTC 2004-Spring. 2004 IEEE 59th, Vol. 4, May 2004, pp. 2322 – 2326.

[3] 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.

[4] 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.

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

[6] D. Black, J. Donovan, B. Bunton, A. Keist, *SystemC: From the Ground Up*, Springer, 2010.

[7] Esperan, "Introduction to the SystemC Tutorial". Web: http://homepages.cae.wisc.edu/~ece734/SystemC/Esperan_SystemC_tutorial.pdf

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