Time Synchronization in SpaceFibre Networks

Elena Suvorova
Saint-Petersburg State University of Aerospace Instrumentation
Saint Petersburg, Russia
suvorova@aanet.ru

***Abstract*—Real-time requirements are actual for most aerospace systems. Time synchronization in all network devices needs for implementation of real-time mechanisms. The SpaceFibre standard is developed for aerospace networks. But current version of this standard does not include any time synchronization mechanism.**

In this paper, we consider time synchronization mechanisms supported in standards currently used for aerospace systems and propose several mechanisms for time synchronization in SpaceFibre networks. We evaluate achievable synchronization accuracy for proposed mechanisms. For some of them achievable synchronization accuracy is better than that for considered standards.

We propose implementation of proposed mechanisms based on dynamically reconfigurable local time controller unit. This implementation made it possible to explore the achievable characteristics of all proposed mechanisms. It is planned to use it in further research due to the possibility of reconfiguration.

In the paper we show that in different networks, it may be advisable to use different synchronization mechanisms depending on user requirements. Our implementation with the dynamic reconfiguration provides the possibility of using various mechanisms, including when implemented with ASIC technology.

I. Introduction

The SpaceFibre standard is developed for aerospace local networks. Most aerospace systems have real-time requirements. The severity of these requirements may vary, depending on the purpose of the networks. For some systems, an accuracy of several ms is acceptable, for other systems an accuracy of several μs is required, in some cases a higher accuracy may be required [18], [19], [20].

To implement real-time mechanisms, in particular, guaranteed data delivery time, time synchronization is required in all network devices (routers and terminal nodes). The achievable characteristics of real-time mechanisms will depend on the accuracy of time synchronization in the network.

Synchronization of time counting in devices is necessary for several reasons. First, the time counter operates using the clock generated by the PLL. The clock waveform generated by the PLL is not ideal. This affects the accuracy of the timing.

Secondly, at the beginning of the system operation, the devices, as a rule, are not turned on strictly at the same time. In addition, different PLLs may take different times after power-on to start generating the clock signal. Several devices may turn on after some time after start of system. For example, this can happen if cold redundancy is used in the network or some devices are not used during several periods of system operation.

Due to these factors, time synchronization in devices is necessary throughout the entire operation of the network.

The current version of the SpaceFibre [1] standard does not specify time synchronization mechanisms. In this paper, we propose time synchronization mechanisms for the SpaceFibre network. For the proposed mechanisms, the achievable synchronization accuracy is evaluated.

The paper is organized as follows. In section 2 we briefly describe existing approaches to time synchronization in local networks that support real-time services. Section 3 is about Proposed time synchronization mechanisms for the SpaceFibre standard. IN section 4 we describe proposed implementation of the mechanisms. In section 5 we estimate the achievable accuracy of time synchronization using the proposed mechanisms. Section 6 concludes the paper.

II. Existing approaches to time synchronization in local networks that support real-time services

In this section, we consider standards and protocols that were originally focused on use in real-time systems, including hard real-time, and which are now actively used in aerospace networks.

A. The network time protocol

The first protocol for time synchronization - network time protocol (NTP) [2] was developed in 1985. It was focused primarily on the Ethernet based networks. When using this protocol, it is assumed that there is at least one source of exact time (astronomical) connected to one or several network subscribers - primary time servers.

The current time is represented by a 64-bit number, suggesting the time elapsed since 0000 UT on January 1, 1900. Time presentation precision is 0.2 ns. Within the framework of this protocol, symmetric mode is possible, in which two devices synchronize the time of each other, and client-server mode, in which the client's time is synchronized with the server's time. In large networks, a hierarchical structure of servers can be built. The higher the server hierarchy level, the more accurate its time. The servers at the higher levels of the

hierarchy synchronize the time in the servers at the lower levels of the hierarchy. Those, in turn, synchronize the time in the clients.

In all modes, time synchronization is performed at regular intervals. The value of period (duration of synchronization interval) is selected for a specific network in accordance with the required time characteristics.

To synchronize the time, the devices exchange messages containing information about the current time and time stamps to account for the transmission time of messages between devices. It is assumed that the intrinsic drift rates of both peers or of both the client and server clocks are small and close to the same value. It is assumed that the time of transmission of the message from the first device to the second is approximately equal to the time of transmission of the message from the second device to the first. This protocol can only achieve synchronization accuracy on the order of milliseconds [3].

B. The IEEE 1588 standard

In 2002, the standard and protocol IEEE 1588 was developed [4]. This protocol is protocol capable of nanosecond synchronization accuracy. In accordance with this standard, the network has a time master that synchronizes the time in other devices. The time master and the slave device being synchronized exchange messages. (The time master and this device are directly connected to each other.)

The time master sends Sync and Followup messages. Sync contains the value of time in the time master and Followup contains the value of time when the Sync message was hardware-sent from the master. These values are needed to estimate the transmission time between the master and the synchronized slave device. Time estimation is performed according to local time in the time master. Then the synchronized slave device sends a DelayReq message, which allows us to estimate the time of data transfer between the synchronized slave device and the time master. In response, the time master sends a DelayResp, which contains information about the time it received the DelayReq.

As a result, the synchronized slave device can estimate the time of message transmission between it and the time master in accordance with its local time. Further, assuming that the physical transmission time from the time master to the slave and from the slave to the time master is the same (transmission channels are symmetric), based on the estimated transmission time, the time difference in the time master and the slave is determined. The calculation of the divergence is performed on the synchronized slave device.

If the time difference is big, absolute time adjustment is applied. In opposite case, a percentage change of frequency is applied to slave clock.

In 2008, a new version of this standard was developed, which allows to perform time synchronization between a pair of devices, the communication between which includes one or more transit routers [5].

It should be noted that the accuracy of time synchronization when using this standard very much depends on the implementation features of network devices (characteristics of the local clock signal, the implementation of data transmission paths), the data transmission standard used in the network, the data transmission rate over physical channels, the size of the network (the distance between the synchronization source and the synchronized device).

IEEE 1588 is the basis for time synchronization methods in several standards widely used in aerospace networks: TTEthernet, FiberChannel, Serial Rapid IO. However, along with this protocol, other methods of time synchronization can be implemented in these standards.

C. The TTEthernet standard

IEEE 1588 can be implemented over TTEthernet.

Additional functionality can be realized on top of a TTEthernet device that generates IEEE 1588 clock synchronization frames. TTEthernet provides means to compensate for delays through the TTEthernet network. The achievable synchronization accuracy is several μs [6], [7], [8].

Along with this, another method of time synchronization can be used in TTEthernet. TTEthernet relies on a redundant hierarchical master-slave method that has a distributed fault-tolerant majority of master nodes and master switches to provide the time in the system. This method is unique for TTEthernet and can be combined with other mechanisms such IEEE 1588 [8].

TTEthernet takes a two-step approach to synchronization. In the first step synchronization masters send protocol control frames to the compression masters. The compression masters then calculate an averaging value from the relative arrival times of these protocol control frames and send out a new protocol control frame in a second step. This new protocol control frame is then also sent to synchronization clients [8].

The decision on which devices are configured as synchronization masters, synchronization clients, and compression masters arises from the requirements on the system architecture. End systems can be configured as synchronization masters and switches as compression masters. But system configurations with end systems configured as compression masters and switches as synchronization masters are also possible. Switches and end systems not configured either as synchronization or compression masters will be configured as synchronization clients [8].

TTEthernet distinguishes four different levels in synchronization topology. On the lowest level, TTEthernet defines the device level that comprises synchronization masters, synchronization clients, and compression masters. The cluster level groups devices with the same synchronization priority and the same synchronization domain to a single cluster. On the multi-cluster level, several clusters with different synchronization priorities but same synchronization domain are grouped together. Finally, the network level groups different clusters (potentially multi-clusters) with different synchronization priorities and different synchronization domains [8].

TTEthernet specifies the concept of a cluster. A TTEthernet cluster is a group of end systems and switches that have the same synchronization priority and synchronization domain. TTEthernet clusters could be used in large TTEthernet networks, where different clusters shall be able to run in isolation, but shall be able to operate in a master-slave mode, once a high priority cluster joins the network or is powered on [8].

D. The FibreChannel standard

Fibre Channel can be configured as a deterministic network, by using an in-band synchronization protocol to synchronize the clocks of attached nodes thus allowing autonomous synchronized data transmissions. A synchronization protocol based on the IEEE 1588 standard is used. Clock Synchronization over Fibre Channel is attained through a Clock Synchronization Server that contains a reference clock [9].

A synchronized message sent out periodically can be used to trigger the relative time start of each network time frame or window. Using this approach, the beginning time is determined each time the synchronization message is transmitted, as opposed to setting up a global clock between each of the nodes. The Server synchronizes Client's clocks to the reference clock on a periodic basis using either Primitive Signals or ELS frames

The Fibre Channel clock synchronization service allows clocks located within nodes to be synchronized to microsecond accuracies.

With ELS Clock Sync, individual client ports on a network can request that a Clock Sync server port located within a node or switch provide periodic updates of network time. For this service, Fibre Channel defines an option for resolution up to 64 bits, with an LSB value of 73 ps, and a rollover time of 43 years. Typically, the synchronization period is 10 - 20 ms [10].

For FibreChannel, other time synchronization protocols are also implemented, in particular the Network Time Protocol. However, the achievable time synchronization accuracy for them is not high. For Network Time Protocol, it is usually from 1 millisecond to 50 microseconds.

E. The Serial RIO standard

The time synchronization mechanism is not specified in the Serial RapidIO standard. There are implementations of IEEE 1588 based clock synchronization method for Serial RIO based networks. These implementations satisfies the synchronization precision of 20 ms without using any extra hardware for synchronization [11].

F. The SpaceWire standard

The time synchronization mechanism, specified in the SpaceWire standard, is significantly different from IEEE 1588. This standard uses special control codes - time markers for time synchronization. These control codes have the highest priority when transmitting to the physical channel, which minimizes network transmission delays. These control codes are broadcast over the network from the source to all routers and terminal nodes on the network. The time source sends out time markers periodically, and they are used to synchronize clocks across all network devices (terminal nodes and routers). Sending one time marker allows us to synchronize the time across all devices on the network [12]. This makes this approach significantly different from IEEE 1588, in which each device is synchronized separately. Accordingly, when using this approach, the overhead (network load), and the synchronization time for all devices is significantly less than when using IEEE 1588. However, when using the SpaceWire approach, there is no way to estimate the transfer time between devices to improve synchronization accuracy.

The accuracy of time synchronization when using the SpaceWire approach depends on the size of the network (the maximal number of transit routers from the time source to the network nodes), the transmission speed over physical channels. If the speed is 400 Mbit/s, then in a network in which the maximal distance between the time source and the nodes does not exceed 10 transit routers, the time synchronization accuracy is about 2 μs. The achievable accuracy when using the SpaceWire approach is no worse than for the standards discussed above with the implementation of IEEE 1588. This is achieved due to the fully hardware implementation of the mechanism, the highest priority of time markers and the small length of these symbols, that allows their quickly transmission over network.

G. The SpaceFibre standard

This standard does not specify a time synchronization mechanism [1]. In the following sections, we propose various mechanisms for time synchronization for SpaceFibre, and evaluate achievable synchronization accuracy with their use.

As you can see from this overview, most of the reviewed standards support IEEE 1588 mechanisms. However, a number of standards implement other mechanisms as well. Best synchronization accuracy is achieved for TTEthernet and Fibre Channel. It is 1 μs.

III. PROPOSED TIME SYNCHRONIZATION MECHANISMS FOR THE SPACEFIBRE STANDARD

A. The first synchronization mechanism

The first of the proposed mechanisms is based on the use of broadcast time codes for synchronization, periodically sent by the time source to all devices (routers and terminal nodes) of the network.

There is a local time counter in each device (router, terminal node). In order to synchronize the values of the counters in all devices, there must be a device (group of devices) on the network that is the time source. (In this paper, we do not consider the issues of ensuring resilience to faults of the time source; this is the subject of separate consideration.)

The time source sends broadcast time codes, which are used to synchronize local time counters in all network devices (routers and terminal nodes) with a period T (synchronization period). Time markers or Broadcast messages can be used as time codes for synchronization (below we will consider the

implementations for the mechanism using time markers and using Broadcast messages).

Synchronization period (T) is a parameter defined by the designer of network. The synchronization period depends on the size, structure of the communication network, the transmission rate, the required value of the synchronization accuracy (JT).

The local time counter value in each device can be corrected in accordance with receiving broadcast codes from the time source.

After power on, after is resetting of the device, the time counter is set to 0.

Time counting in all devices except the time source is prohibited until the arrival of the first correct broadcast time code. The time counter starts when the correct time code is received. Then the time is counted cyclically in the range from 0 to T.

We propose the following approach for correcting the time counter (however, other approaches can be used in this mechanism, for example [3], [8], the choice of the approach is the subject of further research).

(1) If the time counter reaches T-1, the next time code has not arrived, the time counter is reset to 0 and the time counting continues.

(2) If the next time marker arrives at the device when the counter value $\in$ (T-1-DP, 0 + DP), where DP is a parameter whose value depends on the delivery time jitter in the network, then when it is received, no additional actions are performed. (It is believed that this situation is caused by jitter of the delivery time of the time code and / or jitter of the counting accuracy in the considered device, with both jitter being within acceptable limits).

Situation (2) is potentially absorbing for situation (1) if the time code came in the interval of counter values (0, 0 + DP)

(3) If the next time code arrives at the device when the counter value $\notin$ (T-1-DP, 0 + DP), the counter value is adjusted, in current variant we set it to 0.

In the future, it is planned to consider other variants of adjustment.

As noted above, time markers or broadcast messages can be used as time codes to implement this mechanism.

The time marker format is presented in the Fig. 1.

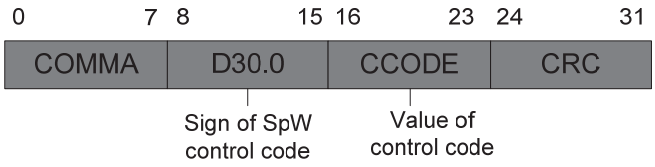

Fig. 1. The time marker format

The broadcast message format is presented in in the Fig. 2.

0 7	8 15	16 23	24 31
COMMA	SBF	BC	B_TYPE
DATA1_LS	DATA1	DATA1	DATA1_MS
DATA2_LS	DATA2	DATA2	DATA2_MS
EBF	STATUS	SEQ_NUM	CRC

Fig. 2. The broadcast message format

Time markers are 4 times shorter than Broadcast messages. (Time marker length is 4 bytes, length of Broadcast messages is 16 bytes). Therefore, the distribution of time markers over the network is faster than distribution of broadcast messages. In the next section, we'll look at how this affects timing accuracy.

However, the Broadcast message includes an 8 byte data field that may contain additional information. We suggest using this field to improve timing accuracy.

Since a Broadcast message has a fixed length, its transmission time through a physical channel between neighbouring devices is constant (according to the SpaceFibre standard, the transmission rate over a physical channel does not change during operation).

According to the SpaceFibre standard, device ports are implemented as sets of state machines. A broadcast message has the highest priority among SpaceFibre data objects, so the only source of non-determinism in its transmit / receive timing over the SpaceFibre port is the transmission of a character that was already started when the Broadcast arrived to port. Thus, for each port of each device at the design stage, the minimum transmission delay of the Broadcast message can be determined (without taking into account the above non-determinism). At the stage of network development for each physical channel, the transmission delay of the Broadcast message in it can be determined. Each device can store a table of coefficients corresponding to these delays (output port delay, physical channel delay and input port delay).

As a broadcast message goes over the network, the data field of broadcast message can be used to store information about these delays. In the time source, the data field of broadcast message is set to the delay value corresponding to the output ports of this device. Further, in each transit router, this value is incremented taking into account the delays in its ports and physical channels between devices.

Thus, a constant part of the Broadcast transmission delay can be accounted for when synchronizing each device.

B. The second synchronization mechanism

The second proposed mechanism based on the IEEE 1588 standard (master-slave mode).

Broadcast messages are used to implement it. As noted above, they include an 8 byte data field that allows the transmission of a timestamp, that required for implementation of IEEE 1588 based mechanism.

However, to implement this mechanism, it was necessary to change the broadcast message distribution rules. The broadcast message used for time synchronization shall not broadcast over the network, but is transmitted only between a pair of neighbouring devices (master and slave) during synchronization.

The following sequence is used to synchronize the clocks of network devices. The time source device in turn adjusts the time in the neighbouring devices (devices directly connected to it by a physical channel). These devices can be both routers and terminal nodes. The time source acts as a master, these devices act as slaves. Further, each of these devices performs time synchronization of neighbouring devices for which it has not yet been performed. It now acts as master. Thus, the process of setting the time spreads over the network in waves from the time source.

The graph of physical connections between devices of the SpaceFibre network can have cycles. In order to exclude the looping of the process of propagation of time synchronization over the network, we use a numbering synchronization cycles. (We call a synchronization cycle the process in which a wave of time adjustments from a time source will propagate across the network, reaching each device.) Each synchronization cycle has a sequence number from 0 to 3 (the number changes cyclically). Messages exchanged between master and slave during synchronization includes the sequence number of the synchronization cycle. If the device has already synchronized time in the i cycle of synchronization, and it receives a synchronization message again, it ignores it (does not perform the synchronization action again).

An example of several steps of a sequence of time synchronization in devices for a network (graph of this network contains cycles) is shown in the Fig. 3. This example shows the first three synchronization steps of a synchronization cycle with sequence number 0. Devices that have already been synchronized are marked in black. Devices that are under synchronization are marked in grey. Black arrows mark the links between master and slave, between which the time synchronization process is performed. The grey arrows mark the links through which the master sends synchronization messages to the device that is already synchronized, so they are ignored (no re-synchronization is performed).

Consider the process of exchanging messages between master and slave, performed at each step. According to IEEE 1588, it includes four stages. At the first stage master sends messages to slave, at the second stage slave sends messages to master, at third stage master sends messages to slave. At the fourth stage, the slave corrects its time counter (if necessary).

Since in our implementation all actions are performed in hardware, at the first stage of synchronization there is no need to send two messages from master to slave. A hardware-generated time stamp can be placed in the very first message. The sequence of messages exchanged between master and slave is shown in the Fig. 4

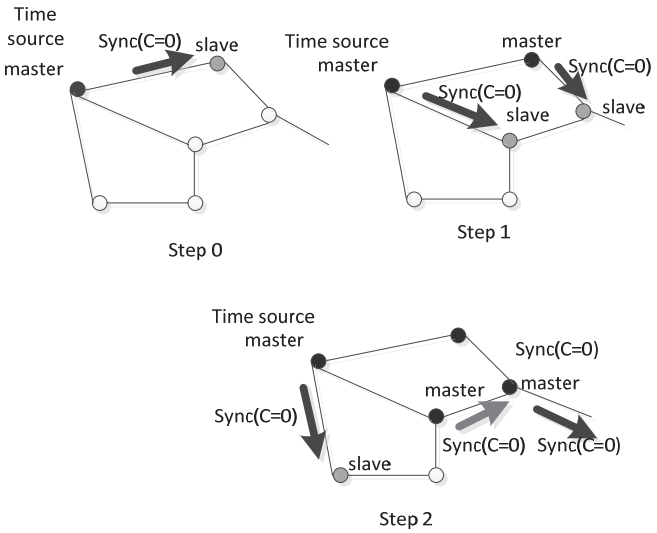

Fig. 3. An example of several steps of a sequence of time synchronization

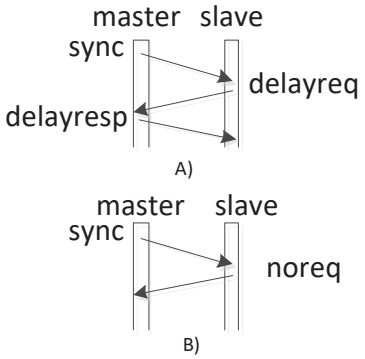

Fig. 4. The sequence of messages exchanged between master and slave

If the time has not yet been synchronized for the slave, it sends a delayreq message. In case the time has already been synchronized, the slave sends a noreq message indicating to master that synchronization is not required. In this case, steps 3 and 4 are not performed.

The algorithm from IEEE 1588 is used to calculate the time counter difference between the master and the slave in the fourth step.

In order to transmit information about the number of the synchronization cycle and the type of message (sync, delayreq, delayresp, noreq) we have made additions to the Broadcast format.

In the Status field, which bits 5 - 0 are not used in the basic version of the standard, we have designated two subfields - cycle_number (bits 1: 0) and message_type (bits 3: 2). The Fig. 5 shows the Broadcast format with these modifications.

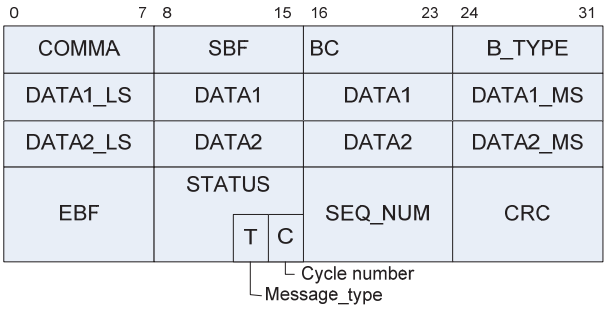

Fig. 5. A proposed modification to broadcast format

IV. PROPOSED IMPLEMENTATION OF THE MECHANISMS

The achievable time synchronization accuracy depends not only on the synchronization mechanism used, but also on whether it is implemented in software or hardware. Since software implementations provide significantly worse achievable synchronization accuracy [3], [10], [11], we use the hardware implementation, which is described in this section.

Since the evaluation of characteristics will be performed for different mechanisms, we have developed a dynamically reconfigurable unit for their implementation - a local time controller unit. This unit performs time counting, processes the received time codes (time markers, broadcast messages), controls their further distribution (according to the rules corresponding to the considered synchronization mechanism), if necessary, adjusts the time counter (in accordance with the considered synchronization mechanism).

Structure of this unit is similar to that proposed by us in [13]. In this case, unlike the variant proposed in [13], memory is used not for storing processed data (since in this case the size of data objects does not exceed 16 bytes), but for storing a table of transmission delays through ports and physical channels. Memory size (Bytes) is

$$Msize = 2 * 2 * Np \quad (1)$$

Where Np - the number of SpaceFibre ports in the device.

For each port, the total input delay, the processing delay and the physical channel connected to it (2 bytes) and the output delay (2 bytes) are stored.

The local time controller unit includes 16 registers and 20 flags.

Reconfigurable DataPath includes a set of FU (Functional Unit) for executing addition/subtraction/comparison, multiplication and division commands, because this set of commands is used in all proposed mechanisms. In the current implementation there are four addition/subtraction/comparison FU, 2 multiplication FU, 1 division FU.

The maximum number of states of a reconfigurable automata is 16 (no more than 10 states are required to implement each of the proposed synchronization mechanisms).

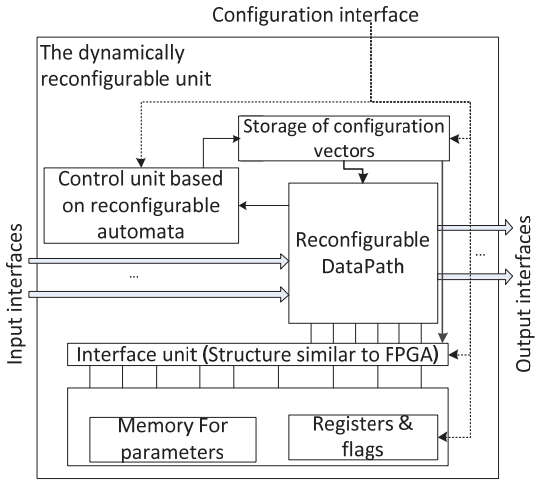

Fig. 6. Structure of a dynamically reconfigurable local time controller

The input interface includes a time marker received from the network with a validity flag, and the port number from which it came, the Broadcast received from network with a validity flag and the port number from which it came.

The output interface includes a time marker to be sent to the network, a set of port flags to which it must be sent, a Broadcast to be sent to the network, a set of port flags to be sent to, the current time counter value.

This reconfigurable unit was used to implement all the proposed mechanisms. Area of this unit (implementation with ASIC) is less than summary area of implementations of every mechanism.

Then we plan use it to implement various algorithms for correcting the time counter, to implement synchronization mechanisms that ensure the fault mitigation mechanisms for further research.

V. ESTIMATION OF THE ACHIEVABLE ACCURACY OF TIME SYNCHRONIZATION USING THE PROPOSED MECHANISMS

As noted above, the deviation between time counters in different devices is due to PLLs parameters and due to time codes propagation delays.

Let's evaluate the component that appears due to the characteristics of the PLL. It does not depend on the used synchronization mechanism, but depends on the properties of the PLLs used.

The achievable accuracy of counting the time periods in each network device (router, terminal node) depends on the PLL used, which is the clock source for the counter.

Currently existing PLLs cannot generate signal with an ideal waveform without jitter and phase (frequency) drift. Let's consider the main parameters characterizing the clock signal generated in the PLL.

Phase (frequency) drift means a change (decrease or increase) in the duration of the clock signal period in time.

Phase (frequency) drift and jitter are related [14], [15], [16], [17].

A particular waveform of clock signal can be characterized by period jitter, cycle-to-cycle jitter, time interval jitter [15], [16], [17].

Period jitter measures the maximum deviation of each single period of jittered clock from that of the ideal clock.

$$PeriodJitter = \max_{k=0,1,2,3}(|P_k - C_0|) \quad (2)$$

Where C0 – period of ideal clock

Pk – duration of k-th period of the jittered clock.

Period jitter is typically specified over a set number of clock cycles. It is recommended to measure period jitter over 100000 cycles to better represent jitter over an "infinite" time span [15, 16, 17].

Cycle-to-cycle jitter measures the maximum deviation of each single period of the jittered clock from the previous period of this clock [14], [15], [16], [17].

$$CycleToCycleJitter = \max_{k=0,1,2,3}(|P_{k+1} - P_k|) \quad (3)$$

Time interval jitter measures the maximum deviation of the edge of the jittered clock from the corresponding edge of the ideal clock [14], [15], [16], [17].

$$TIEJitter = \max_{k=0,1,2,3}(|T_k|) \quad (4)$$

Where Tk – deviation of k-th edge of the jittered clock from k-th edge of the ideal clock.

If Cycle-to-cycle jitter = C, where C is always greater than 0 or C is always less than 0 for any values of k

$$\begin{aligned} &\forall k, P_{k+1} - P_k = C, C > 0 \\ &or \\ &\forall k, P_{k+1} - P_k = C, C < 0 \end{aligned} \quad (5)$$

then we can say that there is a Phase (frequency) drift up or down.

According to the SpaceFibre standard [1], the data signalling rate shall be the same ±0.01%. Let's designate this parameter as A (A = 0.01% = 0.0001). In the implementations, PLLs are used, which provide this strict requirement. And these PLLs (PLLs with the same characteristics) can be used to generate the clock signal used for timing counter.

This requirement excludes the possibility of the clock signal drifting for any extended period of time. However, jitter is possible for clock signal. The actual period of the clock signal can deviate from the ideal clock within the A * C value, both upward and downward. Accordingly, for this clock signal:

$$\begin{aligned} &PeriodJitter = TIEJitter = \\ &2 * A * C_0 = 2 * 0{,}0001 * C_0 \end{aligned} \quad (6)$$

Moreover, the deviation of the period of the clock signal from the ideal can occur in different directions, or predominantly / constantly in the same direction.

The time counter in the device is incremented once in the specified number of clock cycles. If the deviation of the clock signal period from the ideal occurs in different directions, then the deviations are compensated during the counting of time. If the deviation occurs always in the same direction, then the effect of it accumulates.

Accordingly, to assess the divergence between the time counters in the devices, we will consider the worst cases, when there is an accumulation of deviations up or down.

For a period of time t from the moment of synchronization, the maximum divergence of the time counters due to the characteristics of the PLL will be:

$$\begin{aligned} &Jp(t) = 2 * t * A, \\ &Jp^+(t) = Jp^- = t * A \end{aligned} \quad (7)$$

For example, at t = 10 ms $Jp^+(t) = Jp^-(t) = 1$ μs

As a result, the actual duration of the time period with the reference duration T will be within

$$\begin{aligned} &Tr^+(t) = T + Jp^+(T) = T * (1 + A) \\ &Tr^+(t) = T - Jp^-(T) = T * (1 - A) \end{aligned} \quad (8)$$

where Tr – actual duration of time period.

The second component of divergence between time counters of devices is the delay associated with the delivery of the time code (Jd). Jd depends on the synchronization mechanism used.

A. The first proposed mechanism.

The propagation delay (delivery time) of the time code from the time source to the device depends on the number of transit routers and communication lines, on the transmission delays through these routers and communication lines.

Since delivery time of the time code over the network provides only the delay of the time counter in the i device relative to the time source, when estimating Jd, we will consider only the maximum delivery time $Ttmax_i$ of the time code from the time source to the i-th node.

Jd can be estimated using the following formula:

$$Jd = \max_{i=1}^{K-1}(Ttmax_i) \quad (9)$$

where K is the number of devices (routers, terminal nodes) in the network.

We assume that i = 0 for the time source, respectively, $Ttmax_0 = 0$. $Ttmax_i$ can be estimated by the following formula:

$$Ttmax_i = \sum_{j=0}^{Mi-1}(Toutm_j + Tch_j + Tin_j + Tw_j) \quad (10)$$

Where M_i is the number of transit routers;

$Toutm_j$ - maximum time code delay at the output port;

Tch_j - time code delay in the physical transmission channel;

Tin_j - time code delay at the input port4;

Tw_j - time code processing time.

As noted above, since the input port and the local time controller unit are a set of automata and there is no resource conflicts, Tin_j and Tw_j are constants for a particular router implementation. Since the transmission rate on the physical channel does not change, Tch_j is also a constant. The output port, like the input port, is implemented as a set of automata, but resource conflict is possible in it - if another symbol is already transmitted at the time of arrival of the time code, then its transmission is not interrupted. Therefore, we evaluate $Toutm_j$ for the worst case: taking into account the waiting time for the transmission of one character:

$$Toutm_j = Tout_j + Ts \tag{11}$$

where

$Tout_j$ – minimum delay of time code in the output port;

Ts - transmission time of one symbol.

It can be determined by the following formula:

$$Ts = Ls * Tb = 40 * Tb \tag{12}$$

where Ls is the character length (in bits), according to the SpaceFibre standard it is 40 bits [1].

Tb is the transmission time of one bit over the physical channel.

Let's estimate the deviation of time counting in devices from the time source (we denote it as Jt) taking into account Jp and Jd. Jd determines the lag of the start of time counting in the device relative to the time source. Jp defines the deviation of the duration of the period to be counted from the period in the time source (the count in the device can either be lagging behind the account in the time source, or ahead of it).

Let's denote the deviation of the time count in the i device from the time count in the time source $Jt_i(t)$. The maximum upward deviation can be estimated using the following formula:

$$Jt_i^{+}(t) = Tt\max_i + Jp^{+}(t) \tag{13}$$

The maximum downward deviation can be estimated using the following formula:

$$Jt_i^{-}(t) = Tt\min_i - Jp^{-}(t) \tag{14}$$

Where $Ttmin_i$ is the minimum time to transfer the time code from the time source to the device. It can be determined by the following formula:

$$Tt\min_i = \sum_{j=0}^{Mi-1}(Tout_j + Tch_j + Tin_j + Tw_j) \tag{15}$$

If $Jp^{-}(t) <= Ttmin_i$, then the current value of the time counter in the device will be less than in the time source. If $Jp^{-}(t) > Ttmin_i$, then the current value of the time counter in the device will be greater than in the time source (the counter in the device will be ahead of the counter in the time source).

This is illustrated in Fig. 7. At the top of this figure is the time source, time counting in it. The bold lines mark the times when the time source sends time codes. The lower part of the figure shows the device (one of the devices on the network) in which the time is synchronized, the time counting in it. The dashed lines mark the times when the time code arrives at this device with the minimum and maximum delivery times. The dash-dot lines mark the times when in this device the time counter has a value of t1 with a maximum delay relative to the time when the time counter in the time source has a value of t1 and with a maximum advance.

Jt can be determined by the following formula:

$$\begin{aligned}
&if \forall i \in (1,K), Jp^{-}(T) \le Tt\min_i \Rightarrow \\
&Jt = Jd + Jp^{+}(T) = \\
&\max_{i=1}^{K-1}(Tt\max_i) + Jp^{+}(T) \\
&if \exists i \in (1,K), Jp^{-}(T) > Tt\min_i \Rightarrow \\
&Jt = Jd + Jp^{+}(T) + \\
&\max_{i=1}^{K-1}(|Ttm\min_i - Jp^{-}(T)|) = \\
&\max_{i=1}^{K-1}(Tt\max_i) + Jp^{+}(T) \\
&+ \max_{i=1}^{K-1}(|Ttm\min_i - Jp^{-}(T)|)
\end{aligned} \tag{16}$$

Let's evaluate Jt for variant with the propagation time correction scheme.

In this case, the content of the Broadcast message data field is equal to the Ttmin value. The value of this field is taken into account when correcting the time count in the device, therefore

$$Jd = \max_{i=1}^{K-1}(M_i * Ts) \tag{17}$$

And Jt, respectively, can be evaluated by the following formula:

$$\begin{aligned}
&Jt = Jd + Jp^{+}(T) + Jp^{-}(T) = \\
&\max_{i=1}^{K-1}(M_i * Ts) + Jp^{+}(T) + Jp^{-}(T)
\end{aligned} \tag{18}$$

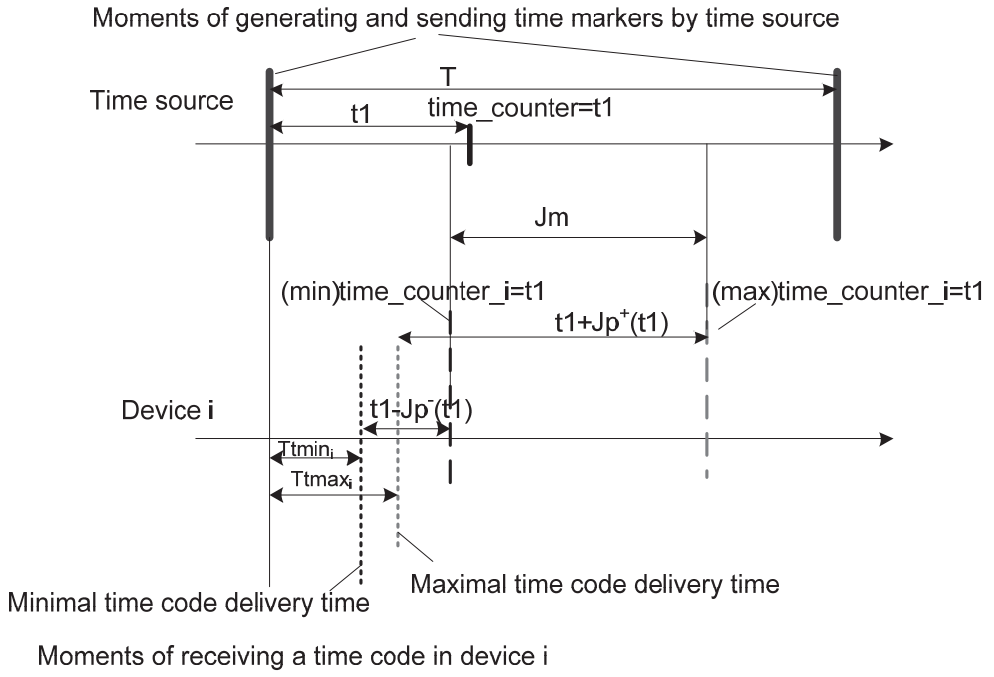

Fig. 7. Illustration of the deviation of a time counter in a network device from a time counter in a time source

B. The second proposed mechanism

As noted above, in this implementation, the local time controller is located at the network level; accordingly, time stamps are generated and analysed at the network level. Local time counters are also located at the network level. The transmission delay of Broadcast message between the master and the slave device includes the transmission time through the output port, the transmission time through the physical link, and the transmission time through the input port. A time stamp placed in the Broadcast message allows take into account all time delays between time counters in the master and in the slave.

As noted above, the Broadcast transmission time can vary by an amount equal to the transmission time of one symbol, which can be determined by formula 12. Accordingly, the broadcast transmission time from master to slave and from slave to master can differ by the value of Ts. Further, during the synchronization period, the values of the counters in the master and slave may differ depending on the characteristics of the PLL. As a result, the deviation of the time count between a pair of neighbouring master and slave devices (Jt2) can be determined by the following formula:

$$Jt2 = 2 * Ts + Jp^{+}(T) + Jp^{-}(T) \qquad (19)$$

Synchronization of time counters in network devices is carried out in chains, in which one device configures the next.

The synchronization time of time counters in all devices is significantly less than the synchronization period. Accordingly, during the synchronization process, the deviation of the time counters in pairs of devices is not large, it can be neglected.

Then the maximum deviation between the time counters in the time source and the rest of the network devices can be estimated by the following formula:

$$Jt = \max_{i=0}^{K-1}(Mi * 2 * Js) + Jp^{+}(T) + Jp^{-}(T) \qquad (20)$$

We have estimated the dependence of Jt on the number of transit routers, on the synchronization period, on the length of physical communication lines, on transmission rates along the physical lines for the proposed mechanisms. The Fig. 8 shows the graphs of the dependence of Jt on the number of transit routers at T = 1ms, short physical channels (the delay in them is close to 0), the transmission rate over physical channels 1.25GBit / s. The graph named "time markers" corresponds Tj obtained using the first mechanism and with time markers as time codes. The graph named “Broadcasts” corresponds Tj obtained using the first mechanism and with Broadcast messages as time codes. The graph named “Broadcasts with correction” corresponds Tj obtained using the first mechanism, with Broadcast messages as time codes and using the proposed propagation time correction scheme. The graph named “IEEE 1588” corresponds Tj obtained using the second mechanism.

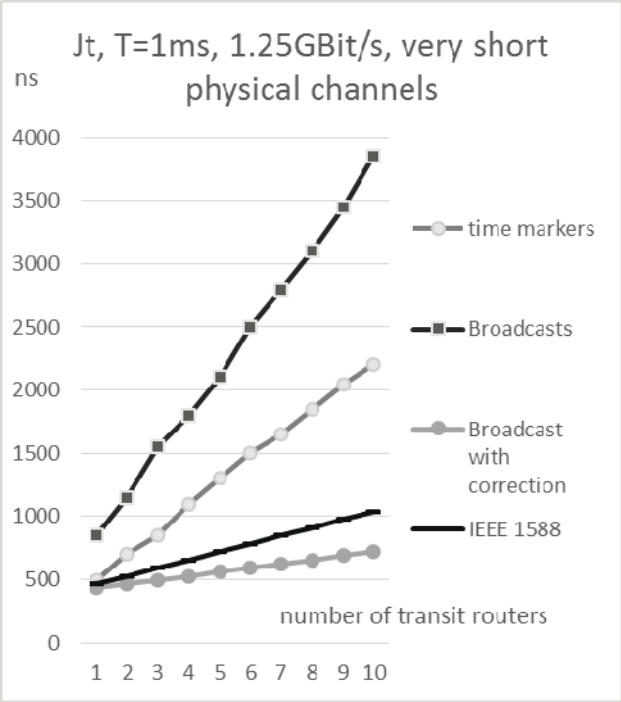

Fig. 8. Dependence of achievable time synchronization accuracy on number of transit routers when very short physical channels are used

The Fig. 9 shows similar graphs obtained for long physical communication lines (the delay in the communication line is 500ns, which corresponds to the delay in an optical cable with a length of about 100 m).

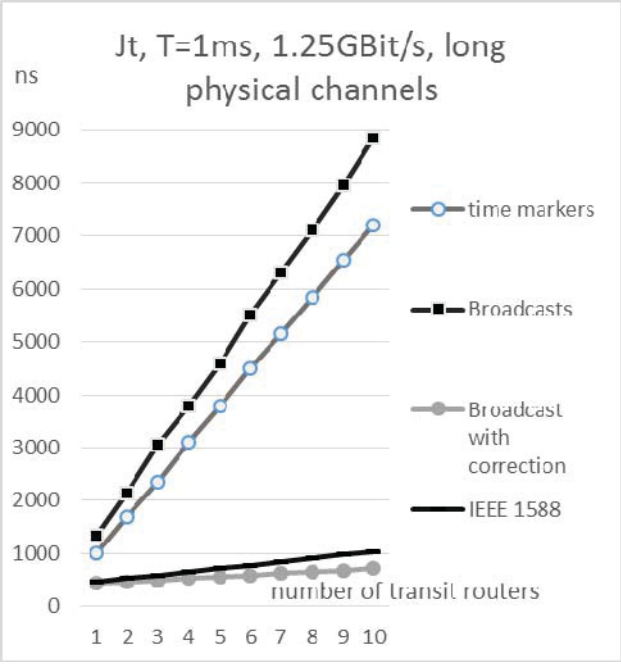

Fig. 9. Dependence of achievable time synchronization accuracy on number of transit routers when very long physical channels are used

These graphs correspond to the transmission speed over physical channels of 1.25Gbps. The trends presented in these figures remain the same at higher bit rates (2.5Gbps, 3.125Gbps and higher).

As can be seen from these graphs, the Jt values obtained using the first mechanism without propagation time correction are significantly inferior to the characteristics obtained using the second mechanism. Moreover, with increasing of transit routers number, with increasing of communication lines delay, Jt increases very significantly for the first mechanism without propagation time correction.

As you can see from these graphs, the time synchronization accuracy obtained using the first mechanism with propagation time correction is better than the time synchronization accuracy obtained using the second mechanism. When using the first mechanism with propagation time correction, in all considered cases, achievable time synchronization accuracy of less than 1 μs is attainable.

Next, we compared these two mechanisms at different values of the synchronization period (1ms, 2ms, 5ms). The corresponding graphs are shown in the Fig. 10. As you can see from these graphs, using the first mechanism with propagation time correction achieves better performance than using the second mechanism. Moreover, for the first mechanism, the value of Jt grows more slowly with an increase in the number of routers. This shows the preference for using it for large networks.

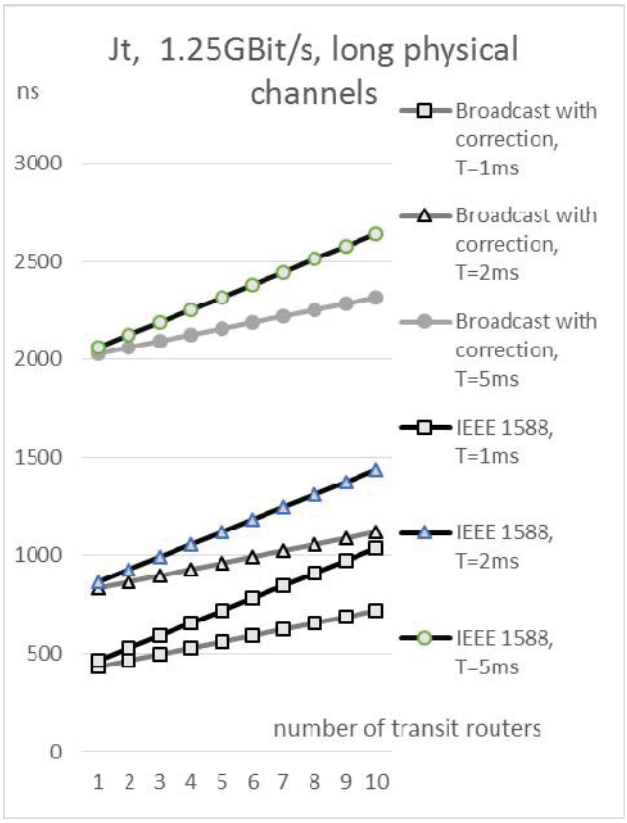

Fig. 10. Dependence of achievable time synchronization accuracy on number of transit routers for different values of the synchronization period

These results were obtained using mathematical models. Next, simulation modelling was performed. For simulation, a network model was used, which includes RTL models of routers and terminal nodes developed in the VHDL language and models of communication channels with the ability to

introduce different transmission delay. Simulation modelling was carried out using the Cadence Incisive 19.2 toolkit. The values of the achievable synchronization accuracy Jt obtained using simulation are 2 - 5% less than the calculated ones (the accuracy is higher). This was because the calculations performed for the worst cases, but the more good cases occurred during the simulation.

Next, let's estimate the amount of service information transmitted when using the proposed mechanisms. In general, it can be estimated by the following formula:

$$B = Kt * Lt \quad (21)$$

Where B – the amount of transmitted service information (time codes)

Kt – the number of time codes transmitted over the network in each synchronization cycle

Lt – the length of time code

When all variants of the first mechanism are used, the number of time codes transmitted over the network is the same. (It depends on the connection graph of a particular network.) Let's denote it as Kt1. Let's designate Ltm - the length of the time marker, it is equal to 40bit. Let's denote Ltb - the length of the Broadcast message, Ltb = 4 * Ltm [1]. Then, the amount of information when using time markers as time codes can be estimated using the following formula:

$$Bm1 = Kt1 * Ltm \quad (22)$$

The amount of information when using Broadcast messages as time codes can be estimated by the following formula:

$$Bb1 = Kt1 * Ltb = Kt1 * 4 * Ltm = 4 * Bm1 \quad (23)$$

It is 4 times more than when time markers used as time codes.

When using the second mechanism, the number of transmitted time codes is 3 times greater than when using the first mechanism (when using the second mechanism, three time codes are transmitted between a pair of neighbouring devices, when using the first mechanism – only one time code is transmitted). Let's designate the number of transmitted time codes when using the second mechanism Kt2. Kt2 = 3 * Kt1.

The amount of information when using the second mechanism can be estimated using the following formula:

$$Bb2 = Kt2 * Ltb = 3Kt1 * Ltb = 3 * Bb1 = 12 * Bm1 \quad (24)$$

It is three times more than using the first mechanism with Broadcast messages as time codes and 12 times more than using the first mechanism with time markers as time codes.

The main features of proposed mechanisms are represented on Table I.

TABLE I. COMPARISSON OF PROPOSED MECHANISMS

	Achievable synchronization accuracy	**Service information transfer overhead**
Mechanism 1, time markers	4(worst)	1(best)
Mechanism 1, Broadcast messages	3	2
Mechanism 1, Broadcast messages, propagation time correction	1(best)	2
Mechanism 2	2	3(worst)

In cases where it is especially critical to minimize the network load with service messages, it is preferable to use mechanism 1 and time markers as time codes. mechanism 1 with propagation time correction is preferred when high precision of synchronization is required.

The use of the second mechanism (which implements a synchronization mechanism that is supported in other standards) may be advisable in hybrid networks, different data transfer protocols are used in different parts of which.

VI. CONCLUSION

The paper discusses the existing methods of time synchronization for networks with real-time requirements.

We propose mechanisms of time synchronization for a network based on the SpaceFibre standard, a variant of their implementation is proposed This variant based on dynamic reconfigurable local time controller unit.

For the proposed mechanisms, the achievable accuracy of time synchronization, the amount of service information that is sent over the network is estimated.

The achievable synchronization accuracy does not exceed 10 μs for all proposed mechanisms, which is acceptable for many systems with hard real time requirements.

Achievable synchronization accuracy for first mechanism with propagation time correction and for second mechanism is less than 1 μs. This result is better than Achievable synchronization accuracy in TTethernet and Fibre Channel.

The comparison of proposed for the SpaceFibre synchronization mechanisms and other considered synchronization mechanisms are represented on Table II.

TABLE II. COMPARISSON OF SPACEFIBRE SYNCRONISATION

	Sync is embedded in the standard	**Achievable synchronization accuracy**	**Support of IEEE 1588**	**Support of other sync schemes**
TTEthernet	-	Several μs	+	+ (Multi Level Synchronization)
Fibre Channel	+	Several μs	+	+ (NTP)
Serial RIO	-	ms	+	+ (NTP)
SpaceWire	+	μs	-	+ (Broadcast distribution scheme for time codes)
SpaceFibre	+	μs	+	+ (Broadcast distribution scheme for time codes)

It is shown that in different networks, it may be advisable to use different synchronization mechanisms depending on user requirements.

The implementation of the local time controller that based on a reconfigurable automata allows us to support different synchronization mechanisms in the device. Respectively, the device can be used in different networks, with different synchronization requirements.

In the future, it is planned to develop mechanisms for time synchronization with support of faults in the time source mitigation.

We plan to consider the implementation of various network layer mechanism, transport layer mechanisms, application layer mechanisms (for example, a scheduling mechanism for guaranteed data delivery time) using the proposed time synchronization mechanisms. It is planned to consider how the method of time adjustment affects the characteristics of these high-level mechanisms.

ACKNOWLEDGMENT

The paper was prepared with the financial support of the Ministry of Science and Higher Education and of the Russian Federation, grant agreement No. FSRF-2020-0004, “Scientific basis for architectures and communication systems development of the on-board information and computer systems new generation in aviation, space systems and unmanned vehicles”.

REFERENCES

[1] SpaceFibre - Very high-speed serial link. ECSS-E-ST-50-11C.ESA-ESTEC.Noordwijk, The Netherlands. 15 May 2019. 233 p
[2] Mills D.L. RFC 958. Network Time Protocol (NTP). 1985. 13p.
[3] Wu B.J., Peloquin R. Synchronizing Device Clocks Using IEEE 1588 and Blackfin Embedded Processors. Analog Dialogue. 2009. 5 p.
[4] IEEE Std.1588 - 2002. IEEE Standard for a Precision Clock Syncronization Protocok for Networked Measurement and Control Systems
[5] IEEE Std.1588 - 2008. IEEE Standard for a Precision Clock Syncronization Protocok for Networked Measurement and Control Systems
[6] Time-Triggered Ethernet. European Space Agency. 2020
[7] AS6802: Time-Triggered Ethernet
[8] TTethernet - A powerful Network Solution for All Purposes. TTTech. 2010. 15p.
[9] Fibre Channel — Switch Fabric 6 (FC-SW-6)
[10] Catrine L. Fibre Channel Time Synchronization for Automated Testing IEEE AUTOTESTCON. 2018
[11] IEEE1588-based clock synchronization for embedded networked system with sRIO. International Conference on ICT Convergence (ICTC). Jongmok Jeon, Donggil Kim, Dongik Lee. 2013
[12] ECSS-E-50-12C. SpaceWire - Links, nodes, routers and networks. - European Cooperation for Space Standardization (ECSS), 31 July 2008
[13] Suvorova E. An Approach to Dynamic Reconfigurable Transport Protocol Controller Unit Development. FRUCT 26. 2020. 8p.
[14] Understanding SYSCLK Jitter. Freescale Semiconductor. Document Number: AN4056. 2010. 10p.
[15] Lerner B., Lowenberger A. Understanding Jitter REquirements of PLL-Based Processors. Analog Devices. EE-261. 2005. 9p.
[16] MC926002. Quard 1.25 Gbaud Reduced Interface SerDes Reference Manual. Rev. 2. 2005. 92 p.
[17] New Concept of Low-Noise and Low-Drift PLL Synhesizer Based on Composite Phase D etector. P. Gontarek, M. Zukocinski, K. Antoszkiewicz, F. Ludwig.20th International Conference on Microwaves, Radar and Wireless Communications (MIKON). 2014. 4 p.
[18] ESA. TN03: OSRA communication network specification. 2017, 66 p.
[19] ESA. SAVOIR On-board Communication System Requirement Document (SAVOIR-GS-008). 2019. 68 p.
[20] CCSDS. 853.0-M-1 SPACECRAFT ONBOARD INTERFACE SERVICES — SUBNETWORK SYNCHRONISATION SERVICE. Magenta book, 2009. 23 p.