ISSN 0361-7688, Programming and Computer Software, 2021, Vol. 47, No. 4, pp. 235–239. © Pleiades Publishing, Ltd., 2021.
Russian Text © The Author(s), 2021, published in Programmirovanie, 2021, Vol. 47, No. 4.

Constructing Onboard Switched Networks of Minimum Complexity

V. A. Kostenko[a],* and A. A. Morkvin[a],**

[a] *Moscow State University, Moscow, 119991 Russia*

**e-mail: kost@cs.msu.su*

***e-mail: mr.andrej1102@yandex.ru*

Received October 26, 2020; revised January 20, 2021; accepted January 26, 2021

Abstract—In this paper, we formulate the problem of constructing an onboard switched network of minimum complexity required for real-time transmission of periodic messages, as well as propose algorithms for its solution (generating the network structure and system of virtual links). Results of the experimental evaluation of the proposed algorithms for constructing onboard AFDX networks are also presented.

DOI: 10.1134/S0361768821040046

1. INTRODUCTION

Modern real-time information and control systems are distributed systems that include computational resources, sensors, actuator controllers, storage devices, and displays, which communicate with each other. In these systems, three levels of data processing can be distinguished: level of preprocessing, level of primary processing, and level of secondary processing. To enable real-time execution of functional programs, the network must provide the maximum performance and bandwidth at the level of primary data processing.

Currently, there is a transition from federated architecture to integrated modular architecture. The second approach is known as integrated modular avionics (IMA). There are a number of standards that regulate the construction of IMA systems:

1. ARINC 651 describes the basic principles of constructing IMA systems [1];

2. ARINC 653 is a specification for operating systems [2];

3. FC-AE-ASM-RT is a specification for networks based on Fibre Channel [3];

4. ARINC 664 (AFDX) is a specification for Ethernet-based networks [4].

The ARINC 651 standard describes basic principles of constructing IMA systems. In accordance with this standard, an onboard computer is built on a set of standardized computational modules.

The ARINC 653 standard describes the design of operating systems that enable real-time program execution and isolation of programs of different onboard subsystems when they are executed on the onboard computer. The isolation extends to all resources, including register memory, CPU caches, and I/O buses (ports). Programs of different subsystems are isolated by introduction of sections and windows. Programs of each onboard subsystem have their own section and set of time windows (non-overlapping time intervals). A schedule for closing and opening the windows is generated in advance, before initiating the system. Section programs can be executed only within their time windows, and each section is allocated necessary amount of memory, which cannot be accessed by programs from other sections. Within the window, section programs are executed in accordance with their priorities once the corresponding data are available. The program can be interrupted and then rerun in this window or in one of the subsequent windows of the section. Programs from different sections can communicate only by passing messages.

The FC-AE-ASM-RT standard regulates the construction of exchange networks based on Fibre Channel. The ARINC 664 (AFDX) standard describes the design of onboard exchange networks for civil aircrafts. The basic topology is a switched network based on Ethernet 802.3. Real-time message passing is achieved by introducing virtual links and traffic control mechanism for each link. When constructing an onboard network, characteristics of virtual links and their paths are set in such a way as to ensure real-time transmission of periodic messages via each virtual link, i.e., a message must be passed once in each period.

In [5], considering the example of a radar system with phased antenna arrays, it was shown that the transition from federated architectures to IMA architectures (i.e., when porting primary processing programs from the system computer to the onboard computer) increases data flow in the network by a factor of 10^3–10^5, depending on the characteristics of the radar

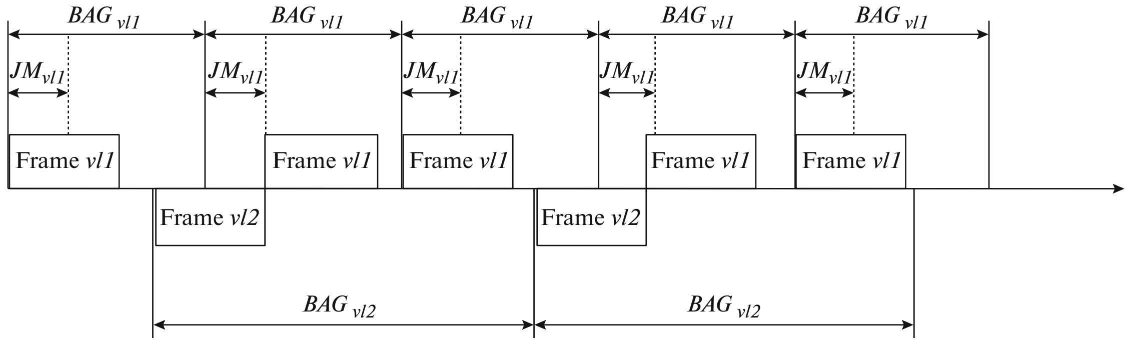

Fig. 1. Virtual links multiplexing using jitter.

system. That is why the construction of onboard exchange networks with minimum complexity is important.

In this paper, we formulate the problem of constructing an onboard switched network of minimum complexity required for real-time transmission of periodic messages, as well as propose algorithms for its solution (generating the network structure and system of virtual links). Results of the experimental evaluation of the proposed algorithms for constructing onboard AFDX networks are presented.

2. PROBLEM OF CONSTRUCTING AN ONBOARD EXCHANGE AFDX NETWORK

In [6], an algorithm for constructing a system of virtual links in an onboard exchange network with a predefined topology was proposed. Here, we formulate the problem of constructing an onboard exchange network of minimum complexity required for real-time transmission of periodic messages (generation of the network structure and system of virtual links).

Suppose that we have a set N of locations of end systems, set K of possible locations of switches, set E_{sw} of possible links between the switches, set E_{end} of possible links between the end systems and switches, and set V of links weights (cable length). Each link[1] $e \in E_{sw}E_{end}$ has its bandwidth R_e. For each end system $n \in N$, a set of subscribers A_n is defined (one subscriber can be connected only to one end system).

Network workload is given by a set of periodic messages MSG each message in which is characterized by the following parameters:

1. T_{msg} (ms) is a message transmission period;

2. $size_{msg}$ (bytes) is the size of a message;

3. J_{msg} (μs) is the maximum release jitter of message generation within the period (i.e., the maximum time interval, starting from the beginning of the period, during which a message can be generated);

4. src_{msg} is the subscriber that sends a message (sender);

5. $\{dst\}_{msg}$ is a set of subscribers that receive messages (receivers), which are connected to the end systems.

For real-time message transmission, the following conditions must hold:

1. a message must be transmitted at least once in a period;

2. t_{msg} (ms) is the end-to-end transmission time;

3. J^*_{msg} (ms) is the end-to-end jitter.

The onboard exchange network is described by a weighted graph $G^* = (N^* \cup K^* E^*, V^*)$, where $N^* \subseteq N$ is a subset of the end systems the subscribers of which are either senders or receivers; $K^* \subseteq K$ is a subset of switches such that at least one data path passes through each switch; $E^* \subseteq E_{sw} \cup E_{end}$ are links along which there is at least one data path; and $V^* \subseteq V$ are weights of all links $e \in E^*$.

Let us introduce a measure of network complexity S for the onboard exchange network, which is defined as the total length of all links:

$$S(G^*) = \sum_{e \in E^*} V^*(e).$$

Then, the maximal network (complete graph) is an onboard exchange network of maximum complexity, i.e., an onboard network G that includes all switches from the set K and all links from the set $E_{sw} \cup E_{end}$.

Constructing a virtual link vl means defining the following parameters.

1. LM_{vl} (bytes) is the maximum frame size for a virtual link, $64 \le LM_{vl} \le 1518$ bytes; the size of the frame header is $c = 47$ bytes and the time gap between frames is $gap = 12$ μs.

[1] According to the standard [4], redundancy is achieved by using two independent networks. For each virtual link, the end system sends a data copy to both the networks. In this paper, we analyze data transmission only through one network with the results for the other network being similar.

2. BAG_{vl} (ms) is the start-to-start time gap between frame transmissions for a zero frame generation jitter; according to the standard, its value is from 1 to 128 ms and is a power of two.

3. JM_{vl} is the maximum frame generation jitter on the sender end system.

4. S_{vl} is the sender end system.

5. D_{vl} is the set of recipient end systems.

6. $Tree_{vl}$ is a tree of frame transmission routes in the network.

7. MSG_{vl} is a set of messages sent via a given virtual link by one subscriber connected to S_{vl}.

Figure 1 shows that frames of virtual links vl_1 and vl_2 cannot be transmitted strictly regularly (with the time gap BAG). When shifting the frames of vl_1 with respect to BAG_{vl1} by the value that does not exceed JM_{vl1}, the frames can be passed without loosing regularity of transmission.

To construct and route virtual links, the following constraints must be satisfied.

1. The total bandwidth reserved for the virtual links passing through the link e must not exceed its bandwidth:

$$\forall e \in E^* : \sum_{vl \in e} \frac{LM_{vl}}{BAG_{vl}} \le R_e. \qquad (1)$$

2. For each virtual link, the frame sending frequency required for transmitting its periodic messages (split into frames) must not exceed the maximum frame sending frequency:

$$\forall vl \in VL : \Sigma_{msg \in MSG_{vl}} \left\lceil \frac{size_{msg}}{(LM_{vl} - c)} \right\rceil * \frac{1}{T_{msg}} \le \frac{1}{BAG_{vl}}. \qquad (2)$$

This constraint is due to the fact that all frames of a message msg must get to the link before the next message msg arrives, i.e., during the period T_{msg}, taking into account that the message msg is split into $\left| \frac{size_{msg}}{LM_{vl} - c} \right|$ frames.

3. The maximum frame jitter on the sender end systems must not exceed 500 μs:[2]

$$\forall vl \in VL : JM_{vl} \le 0.5 \text{ μs}. \qquad (3)$$

4. The maximum end-to-end transmission time and the maximum end-to-end jitter must not violate the following constraints:

$$\forall msg \in MSG : \begin{cases} Dur(msg) \le t_{msg}, \\ Jit(msg) \le J^*_{msg}, \end{cases} \qquad (4)$$

where $Dur(msg)$ and $Jit(msg)$ are procedures for estimating the message transmission times and jitters, respectively.

The solution of the problem consists in constructing a minimal onboard exchange network G^* and a set of virtual links VL for the maximal subset $MSG^* \subseteq MSG$. Thus, we have two constrained optimization problems with constraints (1)–(4):

$$\begin{array}{cc} \max(|MSG^*|) & \min(S(G^*)) \\ MSG^* \subseteq MSG & G^* \in U \\ \text{constraints (1)–(4)} & \text{constraints (1)–(4)} \end{array}$$

where U is a set of all subgraphs of the maximal network. As the first criterion, we select the maximization of the number of transmitted messages; as the second criterion, we select the minimization of the complexity of the onboard network.

This choice of optimization criteria implies two classes of problems:

1. the problem with such input data that all messages can be allocated in the maximal network;

2. the problem with such input data that all messages cannot be allocated even in the maximal network.

3. ALGORITHM FOR CONSTRUCTING AN ONBOARD EXCHANGE NETWORK OF MINIMUM COMPLEXITY

The algorithm consists of the following steps.

Step 1. Create the maximal network G.

Step 2. For each message from MSG, assign a virtual link with such LM, BAG, and JM that constraint (2) is fulfilled.

Step 3. For all virtual links, check the fulfillment of constraint (3). If the constraint is not satisfied for all virtual links, then perform the aggregation procedure. This procedure aggregates the messages from one subscriber into a virtual link (while deleting the old virtual links of these messages) in such a way that constraints (2) and (3) are fulfilled.

Step 4. For each virtual link, build a route taking into account constraint (1) and the total length of the links already used for data transmission; virtual links are processed in decreasing order of the required bandwidth.

Step 4.1. Perform the routing procedure. If the routing is successful, then go to Step 5.

Step 4.2. Perform the limited enumeration procedure for the routes of the already assigned virtual links. If the procedure is successfully completed, then proceed to the next virtual link.

Step 4.2.2. Else, in the case where only one message is transmitted via the virtual link, its assignment is considered impossible. Otherwise, the message with the maximum LM/BAG ratio is removed from the virtual link (its allocation is considered unsuccessful),

[2] This constraint stems from the recommendations described in the standard [4].

and the link is reconfigured using the aggregation procedure. Then, step 4.1 is performed for the reconfigured virtual link.

Step 5. For each message, perform the following steps.

Step 5.1. Estimate the message transmission time and jitter; check the fulfillment of constraint (4). If the constraint holds, then proceed to the next message.

Step 5.2. Reconfigure the virtual link by retuning the parameters *LM*, *BAG*, and *JM* while taking into account the estimates obtained at step 5.1. If, upon reconfiguration, constraint (4) is fulfilled, then proceed to the next message.

Step 5.3. Aggregate virtual links and, for each aggregated link, find its route and check the fulfillment of constraints (1)–(4). In the case of the success, the old virtual links are replaced with a new virtual link. Otherwise, message allocation is considered unsuccessful.

Step 6. Minimize the complete graph *G* as follows.[3]

Step 6.1. For each link, check if it is used in any rout of the assigned virtual links. If it is not, then remove this link from the graph.

Step 6.2. For each switch, check the number of links to other switches. If there are no links, then remove this switch from the graph.

Procedure for Evaluating the Properties of Virtual Links.

Input: virtual link vl.

Output: vl with the desired parameters *LM*, *BAG*, and *JM*.

Procedure for Virtual Links Aggregation.

Input: virtual link vl.

Output: aggregated virtual link for vl with the desired parameters *LM*, *BAG*, and *JM*; otherwise, the procedure returns a failure.

Limited Enumeration Procedure.

Input: set of assigned virtual links; virtual link vl for which no paths have been found.

Output: virtual link vl with a route; otherwise, the procedure returns a failure.

Procedure for Estimating the Maximum Message Transmission Time and Jitter.

Input: message *msg* passed via vl.

Output: $Dur(msg)$ and $Jit(msg)$.

Procedure for Virtual Link Reconfiguration.

Input: virtual link vl with *msg* for which (4) does not hold.

[3] At this step of the algorithm, it is not required to delete links used in the network with subsequent reconfiguration of virtual routes because the routing procedure takes into account not only the length of the links, but also their involvement in the transmission of the already allocated messages; thus, at each iteration of the procedure, the complexity of the network either does not change or increases only slightly.

Output: virtual link vl with new parameters *LM*, *BAG*, and *JM*; otherwise, the procedure returns a failure.

These procedures are described in detail in [6].

Procedure for Virtual Link Routing.

Input: graph *G*; virtual link vl.

Output: virtual link vl with a route; otherwise, the procedure returns a failure.

Step 1. Modify the graph *G* by removing all links the free bandwidth of which is lower than the required bandwidth.

Step 2. For the resulting subgraph, run Yen's algorithm, which uses the following two criteria.

• The edge weight criterion is

$$I(e) = \frac{V_e}{k+1},$$

where $e \in E$, V_e is the length of the link e, and k is the number of virtual links already assigned to this link. This criterion is used to find a path by Dijkstra's algorithm.

• The path comparison criterion is

$$I(path) = C_1 * \sum_{path} I(e) + C_2 * cost(path, vl),$$

where $I(e)$ is the edge weight criterion,

$cost(path, vl) = \sum_{e \in path(S_{vl}, n)} \frac{LM_{vl}}{R_e} + \sum_{v \in path(S_{vl}, n), v \neq vl} \frac{LM_{vl}}{\max\limits_{e \in path}(R_e)}$ is the heuristic [6] that estimates the cost of a path based on the duration of frame transmission via the current virtual link over all transmission links of this path, while taking into account the estimate of expected frame transmissions via other virtual links, $C_1, C_2 \in [0,1]$ are coefficients such that $C_1 + C_2 = 1$.

Among k paths found, the path with minimal $I(path)$ is selected.

The algorithm terminates upon finding the shortest path to one of the receiving end systems to which no paths have yet been found.

Step 3. If paths to all receivers are found, then return the union of these paths, which is the route of a virtual link. Otherwise, go to step 2.

4. EXPERIMENTAL EVALUATION OF THE ALGORITHM

The experimental evaluation was aimed at testing the efficiency of the proposed algorithm on input data of different classes. The efficiency criteria were the total length of links used in the exchange network and the number of allocated messages.

For each data class described in Section 1, message sets of the following three types were generated.

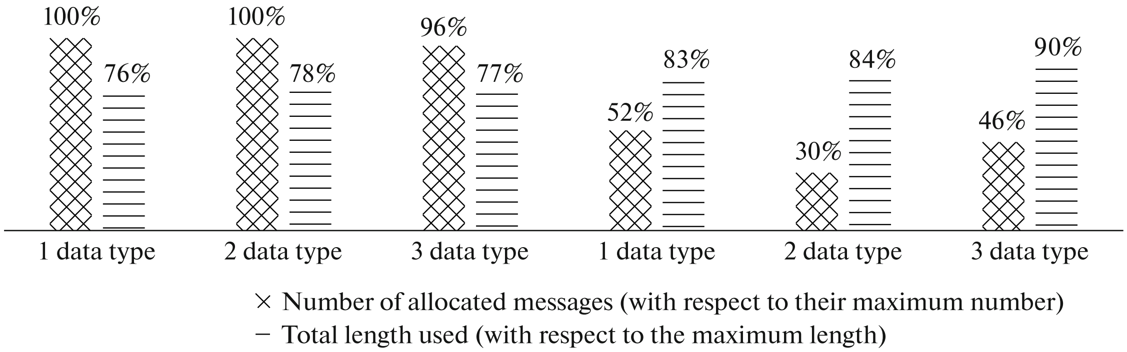

Diagram 1. Results for two data classes (left: 1, 2, 3 data types for 1 data class; right: 1, 2, 3 types data for class 2 data).

1. A large set of messages with low transmission time requirements: 1500 messages from 1 to 1000 bytes in size with the period from 1 to 1000 s and maximum end-to-end transmission time from 0.1 to 100 s.

2. A small set of large messages: 100 messages from 1 Kb to 1 Mb in size with the period from 1 to 10 s and maximum end-to-end transmission time from 1 to 10 s.

3. A set of messages with low transmission time requirements: 100 messages from 1 to 1000 bytes in size with the period from 10 to 100 ms and maximum end-to-end transmission time from 1 to 10 ms.

The experiments were carried out on topologies typical of the IMA systems (e.g., the topology from [11]), to which redundancy was added.

For each topology, data of all classes with all subtypes were generated. For each subtype, 100 random sets were generated; then, the results were averaged. The main results of the investigation are shown in Diagram 1 for all data classes.

Based on the experimental results, the following conclusions can be made:

1. the algorithm minimizes the complexity of the original network by 23% on average;

2. for the second data class, the algorithm constructs the maximal network in the majority of cases.

5. CONCLUSIONS

The algorithm proposed in this paper, for the initial set of periodic messages, constructs a switched network of minimum complexity required for real-time message transmission and a set of virtual links via which the messages are passed.

The algorithm can be adapted to AFDX switches from different manufacturers by modifying the algorithm for tuning virtual link parameters and by using different methods to estimate the maximum end-to-end transmission time and jitter for message transmission.

6. FUNDING

This work was supported by the Russian Foundation for Basic Research, project no. 19-07-00614.

REFERENCES

1. ARINC 651-1: Design guidance for integrated modular avionics, 1997.
2. ARINC 653 specification, Airlines Electronic Engineering Committee. http://www.arinc.com.
3. INCITS 373: Information Technology – Fibre Channel, Framing and Signaling Interface (FC-FS), International Committee for Information Technology Standards, 2003.
4. Aircraft DataNetwork, Part 7, Avionics full duplex switched Ethernet (AFDX) network, Aeronautical Radio Inc. 2012.
5. Kostenko, V.A., Architecture of hardware–software complexes of onboard equipment, *Izv. Vyssh. Uchebn. Zaved., Priborostr.,* 2017, vol. **60**, no. 3, pp. 229–233.
6. Vdovin, P.M. and Kostenko, V.A., Organizing message transmission in AFDX networks, *Program. Comput. Software,* 2017, vol. **43**, pp. 1–12.
7. Al Sheikh, A., et al., Optimal design of virtual links in AFDX networks, *Real-Time Syst.,* 2013, vol. **49**, no. 3, pp. 308–336.
8. Scharbarg, J.L. and Fraboul, C., Methods and tools for the temporal analysis of avionic networks, *New Trends in Technologies: Control, Management, Computational Intelligence and Network Systems,* 2010, pp. 413–438.
9. Frances, F., Fraboul, C., and Grieu, J., Using network calculus to optimize the AFDX network, *Proc. Eur. Congr. Embedded Real-Time Software (ERTS),* pp. 25–27.
10. Bauer, H., Scharbarg, J.L., and Fraboul, C., Applying trajectory approach with static priority queuing for improving the use of available AFDX resources, *Real-Time Syst.,* 2012, vol. **48**, no. 1, pp. 101–133.
11. AFDX from component to application. http://www.prosoft.ru/cms/f/467416/Презентация+про+AFDX+продуктам.pdf. Accessed February 15, 2020.

Translated by Yu. Kornienko