Design of Onboard Local Area Networks

Sergey Pakharev, Alexey Syschikov, Suvorova Elena
Saint-Petersburg State University of Aerospace Instrumentation
Saint-Petersburg, Russia
{sergey.pakharev,alexey.syschikov,suvorova}@guap.ru

***Abstract*—Modern onboard networks, especially those that implement Integrated Modular Avionics (IMA/IMA2G) concept, contain up to thousands of nodes. Integrated computation environment requires hard data interconnection with various requirements and constraints. Manual design of onboard networks becomes extremely hard and it's impossible to produce optimal or even efficient network structures. The presented research is intended for creation of method and tool for automated design of onboard local area networks. This will be used to assist engineers and designers of onboard complexes. We design networks using the SpaceWire communication standard, but the research can be easily modified to support other onboard network types. We develop an algorithm for an automated network construction, the prototype implementation of the algorithm, a probabilistic assessment of the intermediate results of the constructed network. We tested the technology on selected examples of real onboard networks and carried out the final evaluation of the development results. The developed prototype is able to build onboard local area networks based on user-defined parameters and constraints; it has a clear interface and performs the computation in the foreseeable time.**

I. INTRODUCTION

Onboard local area networks are used for communication of devices located inside the aircraft or spacecraft. According to SpaceWire standard devices can be connected in different ways, including direct wire connection between them and switched connections through routers or switches. Devices also can be physically located in different parts of a board structure that puts strong limitations on connection parameters. The presented approach is used to be applied in aerospace area during the design and construction of aircrafts or spacecraft. Onboard networks often have a large number of nodes and therefore it is a very complicated task to make efficient design of such networks manually, even having a group of experts working on this. It is required to construct and evaluate lots of possible variants of connections between nodes to provide the required technical characteristics and satisfy given constraints.

It is much more convenient to provide to experts one or more ready variants of network design, which have best characteristics. It enables the selection from these variants for further improvements by experts.

We present the automated tool for design and evaluation of onboard network based on SpaceWire standard. It allows designing network structures for given set of devices (terminal nodes) according to technical requirements and constraints. The tool can construct one or several network structures with different characteristics for the further processing on the next design stages. It enables building large size networks with hundreds of nodes without involving a large number of experts and to obtain optimal structures and characteristics according to user requirements and constraints.

II. EXISTING TECHNOLOGIES

Today there are multiple technologies that can generate local area networks. We consider several of them.

Volcano Network Architect (VNA) [1] is Mentor Graphics design and analysis tool for CAN and LIN communication systems. VNA provides design, analysis and administrative functions and supports designing systems with legacy electronic control units (ECUs) with fixed messaging. VNA is a standalone tool suitable for integration in legacy design processes as well as the ideal foundation for building a system engineering-based communication design process.

NetGen [2] is a PC-based network design and automatic code generation tool for LIN, CAN, SAE J1939 (a version of CAN used in truck and off-highway industries) and FlexRay. The NetGen tool is a rule-based LIN network and node designer that can also be used to automatically generate or configure the MISRA C source code stack for network communications.

ResNet [3] is an automatic pathway building in biological association networks. The algorithm used to the reconstruction of signaling pathways is also described and validated by comparison with manually curated pathways and tissue-specific gene expression profiles.

Automatic Network Design technology [4] was created by Singaporean specialists Bassiri Masoud and Zhang Hua. This automatic network design technique is able to support any wireless technology: 3G, 4G, GSM, UMTS, Wi-Fi, WiMAX, LTE, etc. This technique involves the synthesis of

ISSN 2305-7254

the network based on characteristics such as antennas coverage area, signal quality, the power of each source, etc. The developed method allows optimizing and automating the development of wireless networks inside buildings, as well as linking networks together.

DesignXpert(Netformx) [5] is a graphically driven desktop-based application, which contain over 42.000 network devices with rule-based engine that assists the user in producing recommendations, configurations, orders and solution. This tool helps the network designer quickly develop a number of design options, under a variety of traffic and cost assumptions. The designs are then analyzed to discover which ones best support the requirements in the present and foreseeable future. The complexity of network design and analysis is such that it is not feasible to perform these tasks manually for medium- to large-scale networks.

There are also many programs to test manually created network. However, in the case of a failed creation of a network is often required to build it a new one which is an expensive task in terms of time and effort of specialists. An example of such a program is testing a “Timing analysis for the design of future switched based Ethernet automotive networks” used for the analysis of road networks [6]. This algorithm is used in the construction of the systems on board vehicles Volkswagen. The essence of the algorithm is in the interim analysis of the resulting networks and the confirmation or refutation of their performance, which used a simulation model. However in this approach has a significant disadvantage - manual construction of the network.

Another approach is Knowledge Based Engineering (KBE) techniques to achieve design reuse and automation. In particular, so called High Level CAD templates (HLCts) are suggested to automate geometry generation and updates. HLCts can be compared to parametric LEGO® blocks containing a set of design and analysis parameters. These are produced and stored in libraries, giving engineers or a computer agent the possibility to first topologically select the templates and then modify the shape of each template parametrically [7].

III. THE SPACEWIRE ONBOARD NETWORKS

SpaceWire is a standard for high-speed links and onboard networks used in aircraft and spacecraft [8]. SpaceWire is based on the part of the IEEE 1355 standard [9]. SpaceWire networks are used in many projects. Initially it was brought to use by European Space Agency (ESA), and at the moment it is used in NASA, JAXA, Roscosmos, as well as in private corporations and companies.

Generally SpaceWire network consists of a set of terminal nodes and network nodes - routing switches [10]. Various devices can be used as SpaceWire terminal nodes. From the SpaceWire point of view they transmit and receive data flows. They can be linked with routing switches or with each other by duplex channels called SpaceWire links. Every node has several ports and an interface to a data source (host device - the processor module, sensor, execution unit, peripheral controller, etc.). The processed data is received from the host device and then sent to a transmitter that is connected to a SpaceWire link. At the far end the data is obtained by the receiver which processes it (decodes) and transmits to the destination host device or to the output port (for the switch node). Terminal node is fundamentally different from the switch node: the transmission of data is possible only under control of a host device (i.e. implemented in software). Routing switch provides direct transfer of traffic between its input and output ports. In SpaceWire networks it potentially possible for some node to communicate by direct links between them without use of routing switches. However a complete network communication infrastructure is only possible with the use of routing switches [11].

Summing up, typical SpaceWire network structure consists of terminal nodes (measurement, actuators, computing, entertainment or other devices), network nodes (switches, routers etc.) and communication links which couples two arbitrary nodes. Terminal nodes can be connected directly or through network infrastructure.

IV. NETWORK STRUCTURE REPRESENTATION

For the network design each terminal node is unique within the entire network and is identified by its ID. It is attributed with a node type which represents a type of a real device and with a set of SpaceWire ports. Every terminal node hosts one or more software applications. Thus the terminal node is a kind of a container for workload tasks; its structure is presented on Fig. 1.

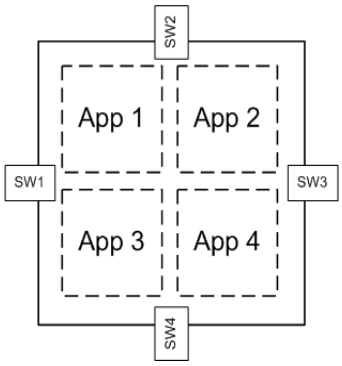

Fig. 1. Terminal node structure

Applications (workload tasks) are identified with IDs (AppID). AppID must be unique within the single terminal node. For applications located at different terminal nodes AppIDs should not be unique. Any terminal node can have arbitrary number of applications, which are limited only with the node type and its characteristics.

Each terminal node must have at least one SpaceWire port for communication with other network components. Every port is characterized by its throughput.

Another important characteristic of a terminal node is a physical cluster number where it is located. Cluster is an abstract entity that reflects a physical co-location of system devices. For example, in the aircraft there are several physical clusters: management cluster, supply cluster, gear cluster and etc. Clustering means that the set of terminal nodes will be closely located in quite a small area and they will be significantly far from other nodes and clusters. Clustered network is often used by network designers to concentrate computational task associated with the physical object in logical groups.

Network synthesis algorithm takes the network structure as input: a set of terminal nodes that are directly connected by logical channels which are virtual representation of data interaction between applications allocated to terminal nodes. Each logical channel is characterized by its data throughput and maximal allowed transmission delay. There can be several logical channels between any two terminal nodes.

A physical channel that will be built by the algorithm connects two nodes in any combination: terminal-terminal, terminal-switch, switch-switch. Maximal physical channel throughput is limited by the lowest throughput of ports to which the physical channel is connected.

One or more logical channel can be routed through a physical channel. Maximal number of logical channels in one physical channel depends on the available throughput of the physical channel and the required throughput of allocated logical channels.

SpaceWire network components (routing switches) are used to build a network infrastructure. SpaceWire switch provides a distribution of data streams in the network. The switch is characterized by the number of ports, their throughput and transmission delay from input to output. Switch also has physical characteristics like weight and power consumption.

During the network design process all characteristics of nodes are taken from a library. There are terminal node and switch node types description, their logical and physical characteristics presented in a library.

All input and output data are described in the XML files according to the specified format.

The input data for network design process are:

- components library with description and characteristics of available terminal and switch node types;
- terminal nodes of the onboard network;
- applications and their allocation to terminal nodes;
- logical channels between terminal nodes and parameters of each logical channel (main are throughput and allowed transmission delay);
- user-defined constraints, for example, maximal physical structure weight, power consumption etc.

The output network structure contains

- terminal nodes;
- switch nodes;
- physical channels which connects nodes;
- logical channels with routing paths through network infrastructure.

Final network structure can be visualized, including physical channels, logical channels and routing paths.

V. PRINCIPLES OF THE AUTOMATED NETWORK DESIGN

The presented program is a part of a full toolchain for designing the onboard SpaceWire networks [12]. Previous tools construct network architecture: set of terminal nodes, applications allocation and logical channels based on the workload specifications. Other tools construct the information-logical structure, perform network modelling and administration during exploitation.

During the network design process we construct the network structure for logical channels according to predefined time characteristics and constraints on weight and power consumption. In the network structure for every logical channel we search a path over network infrastructure. The path is searched according to source and destination nodes, constraints of the logical channel, required throughput and data transmission delay. The delay in the route is calculated as a sum of delays of all switches on the logical channel path [13].

Onboard networks usually have regular logical structure. For example, many sensors send data to a concentrator, control unit sends operation commands to various execution units, etc. In our algorithm we implement a set of network design patterns for identifying regular structures. They are used to simplify the construction of the network and lower the design process time. The basic pattern is a "tree structure".

The tree structure is identified by analysis of logical connections. Ordinal tree structure should have two levels with one terminal node working as a receiver and at least three nodes working as transmitters. This is a typical structure of processing data from a set of sensors. The tree structure can also have inverted direction with one transmitter and multiple receivers: it is typical structure for the control unit and execution units. Logical channels of the

tree structure should have similar properties (allowed deviations are defined by user). All logical channels should have the same direction. An example of the tree structure is presented on Fig 2.

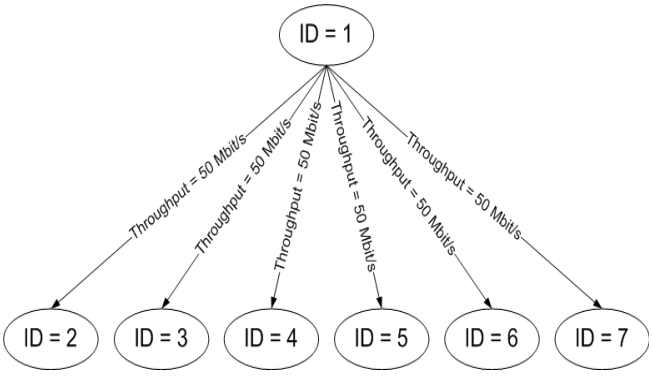

Fig. 2. Tree-structure example

For the identified logical tree we construct the physical structure according to tree channel characteristics, available ports on terminal nodes and available network components. Construction of the tree structure is made in bottom-up manner, i.e. from leaves to the root. Physical tree construction is accompanied with a reservation of ports on nodes: this will assure for nodes involved in other trees or arbitrary structures that there will remain enough throughput for remaining logical channels. On every stage the algorithm tries to finalize the tree construction. If there are not enough ports, the algorithm forms the layer of switches, links all leaves to them and iterates again as if switches on the previous layer are new leaves of the tree. During tree construction all possible combinations of switches are evaluated (Fig. 3).

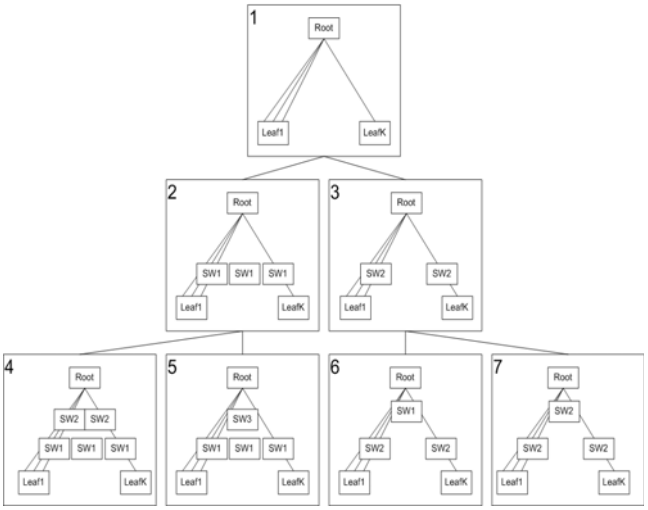

Fig. 3. Tree-structure switch decision tree

Using depth-first search we process the decision tree to find the optimal result (Fig. 4).

All tree structures (one optimal result for every structure) are stored in the intermediate representation allowing user making manual corrections.

The next step after construction of all regular structures is the construction of irregular structure for remaining logical channels. Algorithm uses depth-first search with restrictions [14].

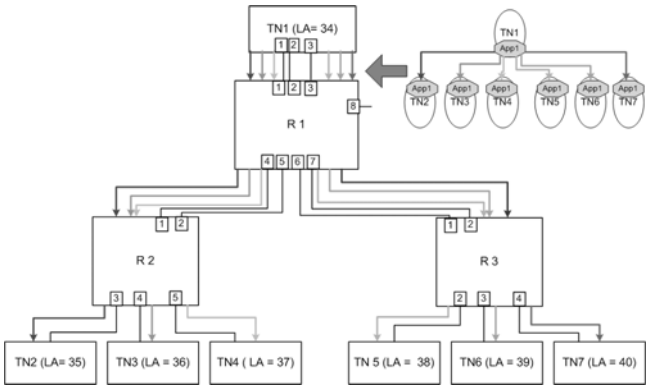

Fig. 4. Tree-structure

It allows evaluating of all possible variants with the limited computation resources; it is necessary for a construction of larger networks [15]. The required time for these computations depends on an amount of remaining logical channels and a variety of available switch types.

The first step in the construction of irregular structure is to build "bridges" between clusters. Bridge is a physical channel that connects two switches from two different clusters. Bridges are used to minimize the amount of wires between clusters in the real aircraft or spacecraft.

The construction of irregular structure is divided into several stages:

- construction of a direct physical link from the last point (node) of the logical channel path to its destination node;
- allocation of a logical channel to an existing physical channel that connects the last point (node) in the logical channel path and any switch not encountered previously in the way of a logical channel or to the receiver node;
- construction of a direct physical channel from the last point (node) in the logical channel path to the switch with direct connection to the destination node and allocation of the logical channel to remaining physical channels.

During the construction of irregular structure the clustering of nodes is taken into account:

- passing a logical channel through the bridge is possible only once for every logical channel;

- logical channel can only pass through the bridge which connects the cluster with the source node and the cluster with the destination node.

Thus if both nodes are in the same cluster, the logical channel will never pass through bridges.

The network construction ends when all logical channels are processed.

If any solution was found during the search, the remaining search function compares intermediate results with founded solution. Search function reduces branching when intermediate results exceed already found solution.

Every found solution is validated for satisfying user-defined constraints.

On the last step solution runtime characteristics are validated with the formal mathematical evaluation method. This method allows calculating not only static network characteristics, but also dynamic characteristics of the functioning network. It calculates the worst case for all logical channels, physical channels, ports and switches. The method evaluates the worst case when each logical channel is the last in the queue for transmission of data on each node. Thus it validates the maximal possible delay that may occur for each channel. This validation ensures the real-time characteristics and characteristics for critical data transferring. If at least one channel exceeds the maximum allowed delay then the solution is dropped from consideration and the program will continue searching for other solutions.

Finally if the solution passes all steps it replaces previous best solution and becomes the current best solution. If the design space exploration will be interrupted, the last best solution will be stored as the network synthesis result.

In addition to physical network structure design the program allows to:

- construct full routing paths for all logical channels;
- generate SpaceWire network node configuration: routing tables, adaptive routings etc.

VI. Examples and Results

As a network synthesis example we demonstrate the synthesis of a network structure for the real spacecraft onboard network [16]. The original manually designed network structure is presented on Fig. 5.

The abbreviations used in the network description are presented on Fig. 6.

The input information for this example is a set of terminal nodes, their interconnections and interconnections intensity. It is presented in the table shown in Fig. 7. We interpret the table as a graph of terminal nodes and logical channels with the specified throughput.

The input structure contains:

- 43 terminal nodes located in two physical clusters;
- 128 logical channels, which are mainly the connections from multiple CC nodes to three main onboard nodes: ATS, OMS and CCM.

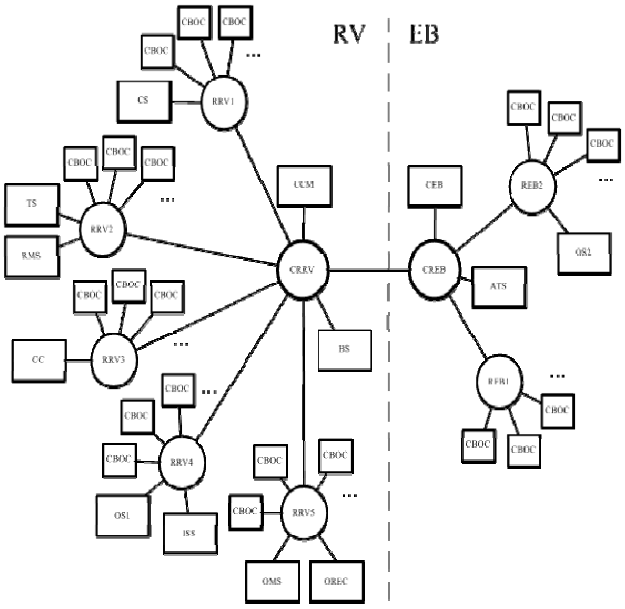

Fig. 5. Manually designed onboard network structure

ATS	Automated Test System
OREC	Onboard Radio Engineering Complex
OS	Onboard Systems
CBOC	Control Block of Onboard Complex
RV	Re-entry Vehicle
EB	Engine Bay
CEB	Computer of the EB
RRV	Router of RV
REB	Router of EB
ISS	International Space Station
CC	Cosmonaut Consoles
OMS	Onboard Measurement System
RMS	Radiation Monitoring System
BS	Bearing System
CS	Communication System
TS	Telemetric System
CCM	Central Computing Machine
CRRV	Central RRV
CREB	Central REB

Fig. 6. Abbreviations used in example

Senders \ Receivers	ATS	OREC	CBOC	CEB	ISS	CC	OMS	TS	CCM
ATS	-	-	-	-	-	·	-	-	-
OREC	-	-	-	-	-	2	-	-	5
OS	-	-	-	-	-	1	1	-	3
CBOC	-	-	-	-	-	·	0.5	-	0.08
CEB	-	-	-	-	-	01	2	-	2
ISS	-	-	-	-	-	15	3	-	2
CC	35	-	-	-	-	·	2	-	0.2
OMS	5	-	-	-	3	01	-	-	5
RMS	-	-	-	-	-	1	1	-	1
BS	-	-	-	-	-	·	0.1	-	-
CS	-	-	-	-	-	·	0.2	-	-
TS	-	15	-	-	15	50	0.1	-	0.01
CCM	5	5	0.08	2	2	5	1	0.01	-

Fig. 7. Logical interconnections and traffic specs

We use 16-port switches to construct the network infrastructure.

It can be easily seen from the logical interconnections structure that most connections are tree structured and will be processed with regular structures algorithm. The remaining connections are constructed with the arbitrary structures algorithm.

The synthesized network structure contains:

- 3 switches;
- 48 physical channels;
- routes for all logical channels;
- minimal weight of a network infrastructure.

Fig. 8 illustrates the visual representation of a designed network.

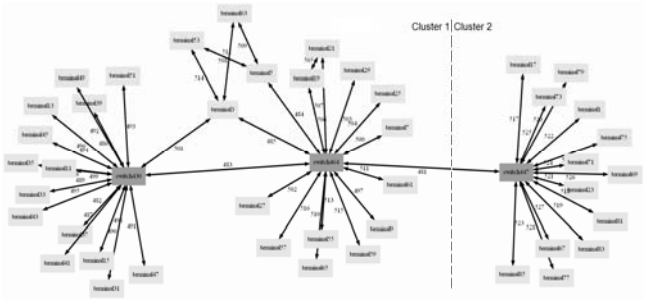

Fig. 8. Visual representation of designed network

As it can be seen from the results of the program, the variant uses both switch-based and direct "terminal-to-terminal" interconnections. Such kind of connections is very rear in manual network design, but in some cases can significantly reduce the network structure. For the presented example the network structure will have one mode switch node (+33% of network mass!) when disabling direct connections.

Additionally the developed program allows inspecting other details of the designed network. For example, visualize the logical structure of the network; draw the data routes for logical channels of a designed network (Fig. 9) etc.

Routes structure that is shown on Fig.9 clearly illustrates that manual development of the network design and configuration is a rather hard and cannot be done efficiently. The automation of such tasks is really necessary.

During the execution of the program more than 100 million of different solutions were checked, but the program was far from evaluating all of them and was interrupted by the user. We estimate that the algorithm complexity increases exponentially with the growth of number of terminal nodes, logical channels and variability of switch types in library.

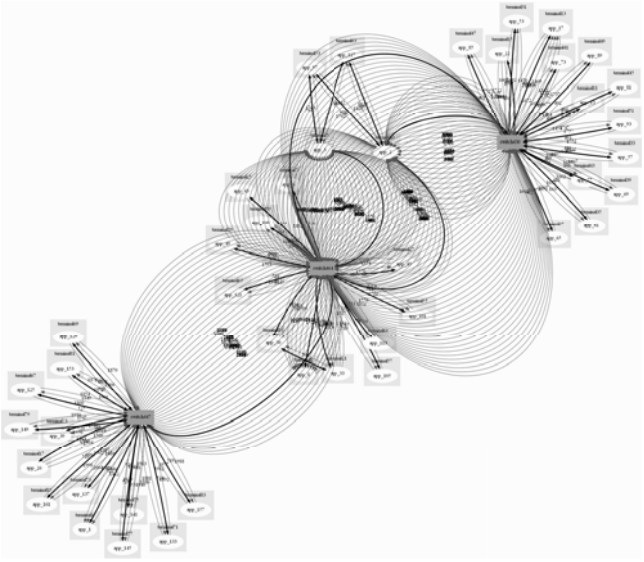

Fig. 9. Visual representation of designed network with data routes

Let's consider a smaller example with of a network which contains:

- 15 terminal nodes;
- 22 logical channels;
- 2 clusters.

Result of the synthesized network is presented on Fig. 10.

Currently the algorithm has its advantages and disadvantages compared with existing algorithms of synthesis of the network.

Advantages:

- implements regular structures for reducing the construction time and building efficient structures;
- implements several types of nodes connections;
- implements physical clustering of elements in the network.

Disadvantages:

- exponential complexity growth depending on the number of nodes, channels, types.

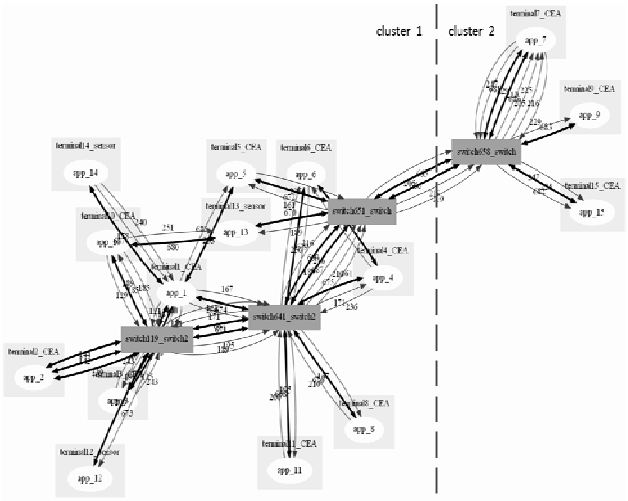

Fig. 10. Visual representation of designed network

The main disadvantage of the presented algorithm is the use of a depth search algorithm that gives the exponential complexity growth. This is the known problem for such kind of algorithms. We have already implemented the set of methods that reduces the complexity of the algorithm including search reduction, and regular structures. We are planning to continue improvements by adding more regular structures, which will also reduce the calculation time.

VII. CONCLUSIONS

The presented algorithm, method and tool prototype allow building large network structures without spending a lot of engineering resources. It allows constructing the optimal solutions according to workload and user-defined constraints. The developed tool is integrated in the toolchain supporting full process of onboard networks design and exploitation.

The presented solution significantly reduces the time required for designing onboard networks. We have plans for extension of the tool with more regular structures, increase the performance and reduce the calculation time.

The developed method and tool utilize all aspects, abilities and features of the SpaceWire standard to achieve best characteristics of designed networks. However we are not limited only with SpaceWire. The core engine of algorithm is applicable for any network type and requires minor tuning for new network type specifics.

ACKNOWLEDGMENT

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] T. Heurung. In-vehicle network design methodology. *The 2nd IEE Conference on Automotive Electronics*, London, pp 47-71, 2006.

[2] C.P. Quigley. *Design of In-Vehicle Networked Control System Architectures through the Use of New Design to Cost and Weight Processes.* Doctoral thesis, University of Warwick, 2011.

[3] A.Yuryev, Z. Mulyukov, E. Kotelnikova. *Automatic pathway building in biological association networks. BMC Bioinformatics*, 2006.

[4] M. Bassiri, H. Zhang, D.K.G. Campbell, T. Forughian, N. Daniel. *Automatic network design. Зatent number WO/2012/036640*,

[5] Teresa C. Piliouras. *Network Design, Second Edition: Management and Technical Perspectives.* CRC Press, 2004.

[6] J. Rox, R. Ernst, P. Giusto. Using Timing Analysis for the Design of Future Switched Based Ethernet Automotive Networks. *Design, Automation & Test in Europe Conference & Exhibition (DATE),* 2012.

[7] K. Amadori. Geometry based design automation applied to aircraft modelling and optimization. *Dissertation, Linköping University Electronic Press*, 2012. No. 1418.

[8] S. Balandin, M. Gillet, "Embedded Network in Mobile Devices", *International Journal of Embedded and Real-Time Communication Systems (IJERTCS),* Issue 1(1), pp 22-36. 2010.

[9] ECSS-E-ST-50-12C - SpaceWire - Links, nodes, routers and networks, *ESA-ESTEC*.

[10] Simpson M., Thompson P.W. DS-Links and C104 Routers. – Networks, Routers and Transputers: Function, Performance and Applications / Ed. by:M.D.May, P.W. Thompson, P.H. Welch. – INMOS, IOS Press, 1993.

[11] Y. Sheynin, T.Solohina, J.Petrichkovich, "SpaceWire technology for parallel systems and on-board systems of distribution ", *Electronics: Science, Technology, Business*, 5/2006.

[12] Alexey Syschikov, Elena Suvorova, Yuriy Sheynin, Boris Sedov, Nadezhda Matveeva, Dmitry Raszhivin. Toolset for SpaceWire Networks Design and Configuration. SpaceWire-2013. Proceedings of the 5th *International SpaceWire Conference, Gothenburg 2013. Editors Steve Parkes and Carole Carrie. ISBN 978-0-9557196-4-6*, Space Technology Centre, University of Dundee, Dundee, 2013. Pp. 149-153.

[13] Samojlenko V.V., *"LANs. The Complete Guide",* 2002.

[14] Kleinberg, Jon; Tardos, Éva, *Algorithm Design*, Addison Wesley –DFS, 2006.

[15] R. Sedgewick , *"Algorithms in C ++. Part 5 Graph Algorithms",* 2001.

[16] Artur Eganyan, Elena Suvorova, Yuriy Sheynin, Alexey Khakhulin, Igor Orlovsky. DCNSimulator – Software Tool for SpaceWire Networks Simulation. SpaceWire-2013. *Proceedings of the 5th International SpaceWire Conference, Gothenburg 2013. Editors Steve Parkes and Carole Carrie. ISBN 978-0-9557196-4-6*, Space Technology Centre, University of Dundee, Dundee, 2013. Pp. 216-221.