SystemC NoC Simulation as the Alternative to the HDL and High-level Modeling

Aleksandr Romanov
National Research University Higher School of Economics
Moscow, Russian Federation
a.romanov@hse.ru

Aleksandr Ivannikov
The Institute for Design Problems in Microelectronics of Russian Academy of Sciences
Moscow, Zelenograd, Russian Federation
adi@ippm.ru

***Abstract*—Actual trends of networks-on-chip research and known approaches to their modeling are considered. The characteristics of analytic and high- / low- level simulation are given. The programming language SystemC as an alternative solution to create models of networks-on-chip is proposed, and SystemC models speed increase methodic is observed.**

I. INTRODUCTION

Continuous development of modern systems on chip (SoC) has led to the emergence of multiprocessor systems. For example, Intel has developed two experimental processors with 48 and 80 cores [1]; a pilot processor with 167 cores is being developed [2]; ZMS-40 100-Core StemCell Media processors with Quad ARM Cortex-A9 cores [3, 4], TILE-Gx72 with 72 C-programmable 64-bit RISC cores processor and TILE-Mx100 targeting networking with 100 64-bit ARM Cortex-A53 cores processor [5] is commercially available. Other companies also pursue their ongoing developments.

Multiprocessor SoCs, whose nodes are combined by the total communication subsystem consisting of routers and short connections between them organized as networks, are called networks on chip (NoCs). Because they are widespread, the problems of modeling, analysis, and simulation of NoCs are very important.

II. STATEMENT OF THE PROBLEM AND OBJECTIVES OF RESEARCH

According to [6], basic directions of current research on the subject of NoCs are:

1) Modeling of network traffic and creating the appropriate test tasks.
2) Display of the problems on NoCs and their planning.
3) Routing and flow control in the NoCs.
4) Ensuring the required quality of service.
5) Management of power, temperature control and timing.
6) Reliability and fault tolerance of the NoCs.
7) Creation of the optimal topology of NoC connections.
8) Development of an effective structure of routers and network channels.
9) Scheduling of NoC deployment.
10) NoC prototyping, testing, and verification.
11) NoC modeling, analysis and simulation.

A large number of research areas reflect the complexity of NoCs as an object of research. It should be also emphasized that on NoC modeling, analysis, and simulation, other areas of search are based. Therefore, the choice of adequate methods and tools for NoC modeling is challenging.

III. ANALYTICAL NOC MODELING

NoC modeling aims to obtain and analyze critical network characteristics such as bandwidth, energy and resource consumption, resistance to bugs and others. Depending on the purpose of the study, the models can be of different level of abstraction and therefore have a different accuracy and the time required for modeling.

A typical approach involves an output, analysis and analytical approximation of formula dependencies that describe the processes occurring in NoCs or their characteristics.

In general, the process of NoC synthesis can be implemented by mapping an application problem characteristic graph (APCG) onto NoC architecture. APCG is $G = G(C, A)$, a directed graph, where C – set of vertices that characterize computing nodes, A – set of communication processes between nodes. In turn, NoC architecture is characterized by: $T(R, Ch)$ topology, where R and Ch – sets of routers and physical links between them; a routing mechanism (P_R); a function of mapping of APCG vertices onto NoC routers ($\Omega(C)$).

According to the above definitions, it is possible to bring out communication energy cost dependence:

$$E = \sum_{\forall a_{i,j}} \upsilon(a_{i,j}) \times E_{bit}(\Omega(c_i), \Omega(c_j)) \qquad (1)$$

where $\upsilon(a_{i,j})$ – capacity of communication process between nodes i, j; $E_{bit}(\Omega(c_i),\Omega(c_j))$ – energy spent on 1 bit of data transfer between nodes c_i and c_j.

Communication energy minimization problem consists in finding such $\Omega(C)$, that arranges for connections of communication process with high capacity to have a low energy consumption to transfer 1 bit. For regular NoC topologies, this problem is solved in [7].

Similarly, the formula of the total volume of data transmitted between nodes in a NoC, is as following:

$$V = \sum_{\forall a_{i,j}} \upsilon(a_{i,j}) \times L_{i,j}(\mathrm{P}_R(r,i,j)) \tag{2}$$

where $L_{i,j}(\mathrm{P}_R(r,i,j))$ – distance between nodes i and j according to the routing algorithm.

The application of routing algorithm that reduces the average distance between nodes makes it possible to reduce the load on the network.

Formulae (1, 2) represent a typical quadratic task of assignments which is the minimization of sum of cost functions products in their weighting coefficients. Similarly, we can find the formulae for determination of other NoC characteristics by substituting of appropriate productivity metrics or resources consumption; by combining some functions into a system, an analytical NoC model can be obtained. This approach is applicable for 2, 3, 4, 5, 6, 7 and 9 lines of NoC research that were defined earlier.

Somewhat detached, analytical models of network traffic are positioned. For NoCs with uniform traffic, stochastic models, as well as models for self-similar traffic, in case of multimedia applications, can be applied [8], [9].

An example of the analytical model is given in work [10] which represents the dependence of parallel data processing on NoC parameters as an expression and analyzes the influence of delays when data transmitting under increase of network dimension.

Analytical NoC modeling has several advantages: it is an obvious approach which does not require the use of special computer-aided design (CAD) systems; the usage of Mathcad, MatLab or other CAD makes calculation process much easier; Simulink even allows to describe a model graphically. However, the analysis and optimization of these models is difficult because of their complexity and nonlinearity of NoC behavior. Commonly analytical model is the first step for more complex model building.

IV. Low-level NoC simulation

The NoC models comprising another group can be referred to simulation ones. Depending on the level of detalization, the models are divided into low-level and high-level classes (to be reviewed below).

Low-level simulation is network emulation at the logic gates. Components of the model are formed by using hardware description languages (for example, Verilog or VHDL). Thus, their functioning is analyzed with the help of specialized software for hardware simulation (for example, ModelSim package), and such a model can be synthesized by using specialized CAD (for example, Quartus II or Synplify Pro). HDL-languages support an interface for high-level programming languages, and this facilitates compatible simulation and verifications (for example, VPI / PLI [11], DPI [12]).

Thus, such an approach is used when simulation, as close to the NoC realization on a physical level as possible (simulation at cycle level, cycle-accurate), is necessary.

This approach is widely spread. Thus, in [13] by using HDL description, various embodiments of NoC Æthereal routers to assess the occupied area on the chip and the maximum clock frequency were synthesized. In [14] there used the routers, described in Verilog for construction, modeling and prototyping of NoC 4x4 mesh MPEG-2 decoder. In [15] NoC VHDL model was used to estimate energy expenditure as well as in [16], where by using VHDL model test sequences and HDL netlist for further SPICE simulation were generated. The classic way from describing in HDL, and then to simulation in ModelSim and compatible emulation in FPGA, for fast hardware-software NoC simulation, was used in [17]. The dissertation [18] offers the VHDL coherent router model and MoCReS NoC, built on its basis, for simulation and synthesis in FPGA. Verilog library with open source – Netmaker [19, 20], implementing the description of classic router with virtual channels and also means of generation of regular topologies should be mentioned. In [21] the capacity of the library was extended to modeling of irregular NoCs by modifying building connections module between the nodes and routing module. Another example is set by Verilog library NoCSimp [22], based on the wormhole router with a simplified structure and FCFS (First Come First Serve) arbitration [23]. The possibility of modeling of irregular NoC topologies by setting up routing tables was realized, and the NoC synthesis, due to the simplicity of implementation, was facilitated.

Low-level approach is applied to virtually all areas of NoC research. Its main advantage is the high accuracy and customizability of models and the possibility of NoC synthesis. However, creation of such models takes significant amount of time; modeling requires specialized programs of hardware simulation (for example, package ModelSim). According to [24], the maximum speed of simulation by using ModelSim comes to about $3{,}2 \cdot 10^3$ cycles/s, which is not enough for analysis of large-scale NoCs (for example, Netmaker, for modeling of NoC with 9 nodes, needed more than 2 hours; NoCSimp usage, under the same conditions of modeling, made it possible to reduce the simulation time to 10 minutes, but with increase of number of nodes in the NoC, the time required for the simulation is growing exponentially, so even the use of simplified HDL models does not solve the problem). Existing approaches of compatible hardware and software simulation and prototyping of NoCs require specialized equipment and special features, and these complicate the use of such methods.

V. HIGH-LEVEL NOC SIMULATION

High-level simulation is data streams distribution modeling in the network. This approach is characterized by speed of development, configuration flexibility, and a relatively small modeling time. In this case, the simulation can be defined as testing of NoC data dissemination model described by a high-level language.

An example of high-level model is set in [25], where transfer of the data to NoC is represented as parallel executable tasks described in C language.

In [26] a universal simulator on Java programming language and modeling results for different regular topologies are represented. The model describes routers and compute nodes of NoCs as separate objects that operate independently of each other. Computing nodes are the generators / consumers of network traffic, and routers perform data transmission and reception according to the routing algorithm.

This approach to the NoCs description is very common and has many advantages: performed network modeling is close to the model experiment; there is a possibility of individual configuration of each router, routing algorithm adjustment, connection of various test sequences of network traffic, and so on.

In [27] a quick high-level NoC OCNS based on OSI network model and the use of the Java language and Qt Jambi framework is represented; it allows to bring processing operations in the graphic interface as a separate stream. Compared with the previous example, this simulator implements irregular topologies modeling: each router contains a routing table, and NoC topology is set on a matrix of links between the routers. Model parameters are specified by using xml configuration file. Simulation results are displayed in the dialog box and selected settings are stored in the summary table. The simulator makes it possible to run multiple iterations of modeling in a row with different configuration. The use of the Java programming language with Qt Jambi framework provides all the advantages of object-oriented programming, cross-platformity of software solutions, and the speed of their development. The complete independence of system components makes it possible to carry out the development, modification and testing of different NoC models. OCNS application gives an opportunity to achieve the simulation time for 9 node NoC faster than in 1 minute (while in Netmaker, under the same conditions, the simulation takes 2 hours) and for 100 node NoC – in 5 minutes.

In the previous model, graphical interface is implemented by software, but there are also ready-made software products which facilitate modeling. For example, in [28] modeling in Petri networks within Visual Object Net simulator is used; in this way, it is possible to analyze competition, cooperation and data conflicts in NoC communication space.

Special attention requires an agent approach described in [29]. Its essence is that the subject area is represented as a set of interacting agents. The developer describes the rules of creation, destruction, and change of agents. An agent is considered to be an object that has memory and ability to take decisions and, therefore, its own behavior of different level of difficulty. The internal structure of the agent can be described in various ways – from formal logic to neural networks. At the time of launching the process of modeling each agent begins to function according to the algorithm of the individual, and the global behavior system appears as the result of the interaction of the whole set of agents. This ensures gradual correction in the working algorithm of the agent, thereby, detailing the model. So, multiple scenarios of agent's functioning of different difficulty level ensure modeling of the operation of the system at different levels of abstraction. Description of models of this type is performed in specialized languages of model description (for example, UML), and the development is done in AnyLogic CAD.

High-level simulation is applicable for most NoC research areas where there is no reference to the hardware implementation, and it is necessary to obtain quick simulation results with sufficient accuracy.

The downside of high-level NoC models is the inability of their synthesis and the relatively low accuracy, and so, there is a need for a hybrid approach, which would be able to combine the benefits of low-level and high-level approaches based on the most common programming language – C.

VI. SYSTEMC AS A COMPROMISE BETWEEN HIGH-LEVEL AND LOW-LEVEL MODELING

SystemC, a language of design and verification of system-level models, is implemented as a C++ library with open code. The library contains a core of event simulation which allows obtaining the executable model of the device. It is used for building transactional and behavioral patterns, as well as for high-level synthesis devices. SystemC uses a number of concepts similar to those applied by hardware description languages VHDL and Verilog (interfaces, processes signals, eventness, hierarchy of modules). SystemC is suitable for behavioral modeling and RTL (Register Transfer Level) – synthesis.

SystemC is widely used by NoC developers. Xpipes library, based on SystemC [17], makes it possible to carry out a complete cycle of NoC simulation and synthesis [31]. In [32], the high-level SystemC-ARTS model for comparative modeling techniques of bus and network methods of SoC building is presented. Noxim [33], NIRGAM [34], and other well-known simulators are also based on SystemC. Some works use a hybrid approach, where test sequences, by using SystemC, are generated, and the model itself is implemented in HDL [35], [36].

SystemC's popularity is due to the fact that it is based on C / C++ language which developed many standard libraries, allowing easier compliant simulation and verification of NoC models. However, C / C++ language is consistent by its nature (instructions are executed one by one), while the hardware processes occur simultaneously and in parallel. This makes the programmer learn a new programming paradigm, as well as specific tools, such as processes, events, signals and others. Although SystemC is a synthetic language, regarding to NoCs it is primarily used as a high-level language for behavioral abstract modeling allowing faster simulation in comparison with the one performed by using HDL-languages (up to $20 \cdot 10^3$ cycles / s) [13, 37], but the NoC synthesis is more complicated.

VII. Translation of HDL-model into SystemC language

Thus, one of the possible ways to increase productivity of NoC models written in HDL-languages, is their translation into SystemC language, which will potentially speed the modeling process up to 7 times (from $3{,}2 \cdot 10^3$ to $20 \cdot 10^3$ cycles / s) while maintaining the high accuracy of the model and the possibility of further NoC synthesis; it will also ensure the opportunity to simplify the model simulation with the help of external libraries. For the translation of HDL into SystemC there are special translators, for example, V2SC [38], as well as a simple hand-by-line translation of HDL-code into SystemC notation.

However, this approach may not yield significant improvements in model's performance that is associated with the fact that the HDL-language design and optimization techniques, effective in it, may not work in SystemC, and even slow the model (for example, the presence of too many nested sub-modules) [39].

To improve the model in SystemC language and to streamline its working, let us formulate a set of techniques and rules based on the analysis of works [40], [41]:

1) By using built-in types of C++ instead of the ones of SystemC types, while describing signals (channels), the bool two-symbol type (“0”, “1”) is better than 4-symbol *sc_logic* (‘0’, ‘1’, ‘x’, ‘z’). Multibit signals can be well described with the help of type *int* and data structures, but not as arrays of *bool* variables.

2) SystemC processes of *SC_METHOD* type application leads to faster simulation in comparison with *SC_THREAD*, since the latter are real threads, and they have their own stack and local variables that requires additional operations for thread context treatment.

3) When transmitting the signal modification information through the ports and connection, it is necessary to call four functions and perform three copy operations (call of *write*(), *request_update*(), *update*() and *read*() functions, copy source variable to *m_new_val*, *m_new_val* to *m_cur_val*, and *m_cur_val* to *dest* variable); so, minimization of quantity of intermodular links leads to decreasing of intensity of mentioned function calls and to decreasing of copy operations which result in shortening of simulation time.

4) To reduce the number of modules and intermodular links it is necessary to replace sub-modules with the help of sequential program that realizes the operation algorithm of particular top-level module.

5) In case it is possible, we will have to use high-level containers instead of the low-level ones. For example, we can employ deque from *STL* to implement the behavior of *FIFO* and this *STL* implementation will operate much faster than hardware one of *FIFO* at *RTL* level of abstraction will do.

6) SystemC ternal operator signal assignments have a better simulation time than SystemC *switch-case* statements, and *switch-case* statements simulate faster than *if-else* statements;

7) To speed up the simulation of SystemC models some techniques of C++ programs optimization can be adopted. Function call causes one of the most inefficiency in simulation time. Therefore, full or partial function inlining can improve the simulator speed. Partial inlining means inlining of simple conditions which may cause the immediate return in the case of function call.

In [39], the comparison results for SystemVerilog and SystemC models realizing the NoCs of mesh 8x8 type, are given. The use of SystemC, and the above SystemC model improvement techniques gives an opportunity to reduce the duration of modeling up to 10.2 times and cut down the amount of occupied memory up to 121 times. All these demonstrate the effectiveness of this approach as an alternative to the high-level simulation while HDL modeling does not satisfy the requirements of the simulation time, but it is necessary to keep synthesizability of the model.

VIII. Comparison of SystemC and high level NoC model

One of the important tasks in the NoC analysis is the impact of the distribution of most intensive tasks of computing load and the exchange of data on NoC nodes. Units, with which the network exchange is the most intense, are called “hot spots”. Since, unlike computer networks, NoC computational cores are compactly arranged, and the data is exchanged at high frequencies, the distribution of “hot spots” is crucial. Another problem is the evaluation of the effect of the geometric shape of regular NoC topologies on their productivity.

To simulate different situations of NoC “hot spots” arrangement in NoCs with different form and the type of topology, it is suitable to use OCNS high-level model [27], a brief description of which is given above, because it supports the connection of different test sequences of network traffic, thin configuration of each router and has a high accuracy and simulation speed.

As an alternative to OCNS, we chose a NoCTweak model [42]. This NoC model is characterized by open code and is designed to study the performance and energy efficiency of networks-on-chip. The use of SystemC and C++ in NoCTweak allows a high speed simulation at the loop level. The simulator is focused on the modeling of the data transition in the NoC communication sub-system of mesh topology and has a large number of adjustable parameters. Open source code of NoCTweak allows performing its modification, optimization, and configuration for a specific application task, which gave an opportunity to adjust the model for our problem, fix some bugs and to optimize NoCTweak source code, and to develop bash scripts to automatically start multiple simulation runs. Collection and analysis of statistics and construction of summary graphs of changes in NoC characteristics are implemented by using the scripts in Matlab.

By using the OCNS model and NoCTweak modified model, NoC simulation with mesh topology, 10x10, 5x20 and 9x11, as well as 7x7 and 5x10 by size, was performed. The simulation results (Fig. 1–3) are generally the same: a reduction in NoC capacity based on mesh topology (up to 25 % in both models), when used with non-optimal topology of geometric dimensions takes place [27], [43]. Almost complete agreement of the results of modeling and commensurate amount of time on simulation demonstrate the effectiveness of the use of SystemC models for rapid NoC simulation as a substitute for a high-level model with appropriate timing win comparing to HDL-modeling. Such features as the use of translators of HDL descriptions in SystemC, maintenance of synthesizability of models, as well as the existence of benefits in the form of a simple integration of third-party libraries and visualization, parallelization, etc., make SystemC a powerful alternative to the high-level modeling.

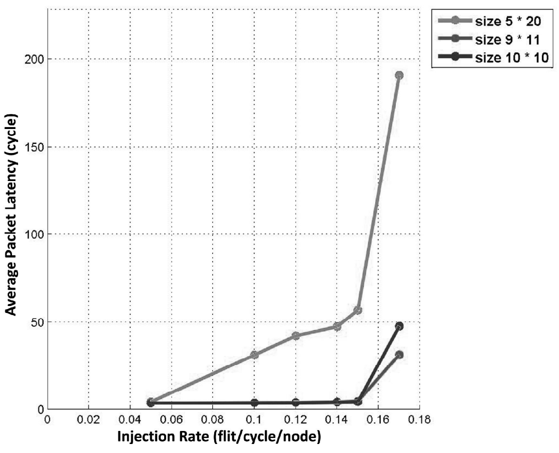

Fig. 1. Dependence of average packet latency on injection rate at 0…0.18 flit / cycle / node

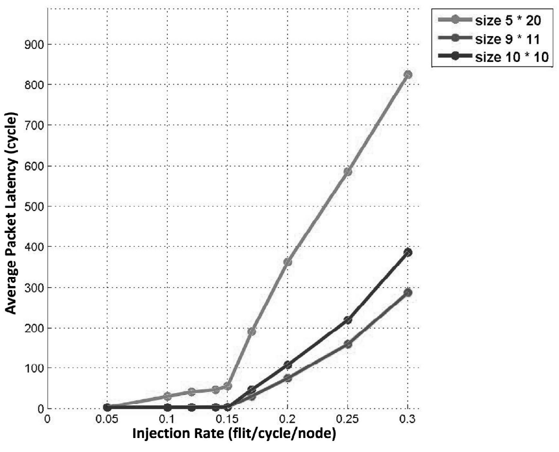

Fig. 2. Dependence of average packet latency on injection rate at 0…0.3 flit / cycle / node

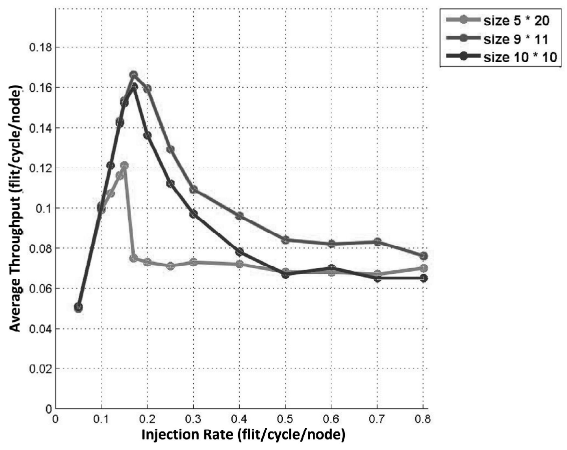

Fig. 3. Dependence of average throughput on flit injection rate at 0…0.8 flit / cycle / node

IX. CONCLUSION

Thus, when developing and researching networks on chip, a choice of the universal approach to their design is challenging. Among the typical approaches there can be distinguished the following: analytical approach (analysis of such models is difficult because of their complexity and nonlinearity of NoC behavior); high-level simulation (applicable for most NoC research areas where there is no reference to the hardware implementation, and obtainment of modeling results with reasonable accuracy is necessary); low-level HDL simulation (high precision, configurability and possibility of NoC synthesis, but high time expenditure on model development and simulation).

The use of SystemC language can be considered to be effective for building NoC models, and this makes it possible to reduce the disadvantages and maximize the advantages of high-level and low-level approaches. To achieve this, methods of improving of SystemC models are formulated; the comparison of results for high-level and SystemC NoC simulation is given.

REFERENCES

[1] J. Howard, S. Dighe, S.R. Vangal, et al., "A 48-core IA-32 processor in 45 nm CMOS using on-die message passing and DVFS for performance and power scaling", *IEEE Journal of Solid-State Circuits*, vol. 201146, no. 1, 2011, pp. 173–183.

[2] D.N. Truong, W.H. Cheng, T. Mohsenin, et al., "A 167-processor computational platform in 65 nm CMOS", *IEEE Journal of Solid-State Circuits*, vol. 44, no. 4, 2009, pp. 1130–1144.

[3] Y. Huangfu, W. Zhang, "Real-Time GPU Computing: Cache or No Cache?", *in Real-Time Distributed Computing (ISORC), 2015 IEEE 18th International Symposium*, April 2015, pp.182-189.

[4] ZiiLABS official website, ZiiLABS ZMS-40 100-Core Quad ARM Cortex-A9, Web: http://www.ziilabs.com/products/processors/zms40.php.

[5] EZchip official website, EZchip Multicore Processors, Web: http://www.tilera.com/products/?ezchip=585.

[6] R. Marculescu, U. Ogras, "Outstanding Research Problems in NoC Design: System, Microarchitecture, and Circuit Perspectives", *IEEE*

Transactions on Computer Aided Design of Integrated Circuits and Systems, vol. 28, no. 1, 2009, pp. 3–21.

[7] J. Hu, R. Marculescu, "Energy-aware mapping for tile-based NOC architectures under performance constraints", *Proceedings of Asia South Pacific Design Automatic Conference*, 2003, pp. 233–239.

[8] V. Soteriou, H.-S. Wang, L. Peh, "A statistical traffic model for on-chip interconnection networks", *Proceedings International Symposium Modeling, Analysis, Simulation Computer Telecommunication Systems*, 2006, pp. 104–116.

[9] G. Varatkar, R. Marculescu, "On-chip traffic modeling and synthesis for MPEG-2 video applications", *IEEE Transactions Very Large Scale Integrated (VLSI) Systems*, vol. 12, no. 1, 2004, pp. 108–119.

[10] X. Chen, Z. Lu, A. Jantsch, S. Chen, "Speedup Analysis of Data-parallel Applications on Multi-core NoCs", *8th IEEE Conference on ASIC (ASICON'09)*, Oct. 2009, pp. 105–108.

[11] A. Saifhashemi, H. Pedram, "Verilog HDL, Powered by PLI: a Suitable Framework for Describing and Modeling Asynchronous Circuits at All Levels of Abstraction", *Design Automation Conference – 2003. Proceedings*, 2003, pp. 330–333.

[12] S. Sutherland, *Integrating SystemC Models with Verilog and SystemVerilog Models Using the SystemVerilog Direct Programming Interface*. SNUG Europe, 2004, 17 p.

[13] K. Goossens, J. Diellisen, A. Radulescu, "Æthereal network on chip: concepts, architectures, and implementations", *IEEE Design & Test of Computers*, vol. 22, no. 5, 2005, pp. 414–421.

[14] U.Y. Ogras, R. Marcillescu, H.G. Lee, et al., "Challenges and Promising Results in NoC Prototyping Using FPGAs", *IEEE Micro*, 2007, vol. 27, no. 5, pp. 86–95.

[15] J.C.S. Palma, C.A.M. Marcon, F.G. Moraes, et al., "Mapping embedded systems onto NoCs: the traffic effect on dynamic energy estimation", *Proceedings of the 18th annual symposium on Integrated circuits and system design (SBCCI'05)*, 2005, pp. 196–201.

[16] C.A.M. Marcon, J.C.S. Palma, N.L.V. Calazans, et al., "Modeling the Traffic Effect for the Application Cores Mapping Problem onto NoCs", *VLSI-SoC: From Systems To Silicon*, vol. 240, 2007, pp. 179–194.

[17] N. Genko, D. Atienza, G. De Micheli, et al., "A Complete Network-On-Chip Emulation Framework", *Design, Automation and Test in Europe – 2005. Proceedings*, vol. 1, 2005, pp. 246–251.

[18] A. Janarthanan, *Networks-on-chip based high performance communication architectures for FPGAs: Ph.D. dissertation: Computer engineering*. Cincinnati: Univ. of Cincinnati, 2008, 143 p.

[19] R. Mullins, A. West, S. Moore, "The design and implementation of low-latency on-chip network", *Proceedings of 11 ASPDAC*, 2006, pp. 164-169.

[20] Fully-synthesizable parameterized NoC implementations library: Netmaker, Web: http://www-dyn.cl.cam.ac.uk/~rdm34/wiki.

[21] O. Romanov, O. Lysenko, "The Comparative Analysis of the Efficiency of Regular and Pseudo-optimal Topologies of Networks-on-Chip Based on Netmaker," *Advances and Challenges in Embedded Computing. Proceedings*, 2012, pp. 13–16.

[22] O.Yu. Romanov, "Porivnjal'nyj analiz rezul'tativ HDL modeljuvannja kvazioptymal'nyh i reguljarnyh topologij merezh na krystali," Problemy informatyzacii' ta upravlinnja. Zbirnyk naukovyh prac'. Kyiv: NAU, 2012, no. 3 (39), pp. 124–129.

[23] A.Yu. Romanov, A.D. Ivannikov, I.I. Romanova, "Simulation and Synthesis of Networks-on-Chip by Using NoCSimp HDL Library", *2016 IEEE 36th International Scientific Conference on Electronics and Nanotechnologies, ELNANO 2016*, in press.

[24] N. Genko, D. Atienza, G. De Micheli, et al., "A Complete Network-On-Chip Emulation Framework", *Design, Automation and Test in Europe, 2005. Proceedings*, vol. 1, 2005, pp. 246–251.

[25] Lv. Mingsong, Y. Guo, N. Guan, Q. Deng, "RTNoC: A Simulation Tool for Real-Time Communication Scheduling on Networks-on-Chips", *International Conference on Computer Science and Software Engineering*, vol. 4, 2008, pp. 102–105.

[26] A. Al-Nayeem, T. Z. Islam, *GpNoCsim 1.0 User's Guide*, 2006, 13 p.

[27] A.Yu. Romanov, S.R. Tumkovskij, G.A. Ivanova, "Modelirovanie setej na kristalle na osnove reguljarnyh i kvazioptimal'nyh topologij s pomoshh'ju simuljatora OCNS", *Vestnik Rjazanskogo gosudarstvennogo radiotehnicheskogo universiteta*, no. 2 (52), 2015, pp. 61–66.

[28] H.C. Feritas, P.O.A. Navaux, "Evaluating On-Chip Interconnection Architectures for Parallel Processing", 11th IEEE International Conference on Computational Science and Engineering Workshops, 2008 (CSE WORKSHOPS'08), 2008, pp. 188–193.

[29] E.V. Korotkij, A.N. Lysenko, "Metod modelirovanija rekonfiguriruemyh setej na kristalle", *Visnik NTUU "KPI". Informatika, upravlinnja ta obchisljuval'na tehnika: Zbirnik naukovyh prac'*, no. 51, 2009, pp. 218–224.

[30] D. Bertozzi, L. Benini, "Xpipes: A network-on-chip architecture for gigascale systems-on-chip", *IEEE Circuits and Systems Magazine*, vol. 4, no. 2., 2004, pp. 18–31.

[31] D. Bertozzi, S. Murali, A. Jalabert, "NoC synthesis flow for customized domain specific multiprocessor systems-on-chip", *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 2, 2005, pp. 113–129.

[32] S. Mahadevan, K. Virk, J. Madsen, "ARTS: A SystemC-based framework for multiprocessor Systems-on-Chip modelling", *Design Automation for Embedded Systems*, vol. 11, no. 4, 2007, pp. 285–311.

[33] F. Fazzino, M. Palesi, D. Patti, *Noxim: Network-on-chip simulator*, Web: http://noxim.sourceforge.net/.

[34] L. Jain., *NIRGAM: A Simulator for NoC Interconnect Routing and Application Modeling*, version 1.1, 2007, 27 p.

[35] J. Chan, S. Parameswaran, "NoCGEN: A Template Based Reuse Methodology for Networks on Chip Architecture", *17th International Conference on VLSI Design, 2004. Proceedings*, 2004, pp. 717–720.

[36] K. Goossens, J. Dielissen, O.P. Gangwal, "A Design Flow for Application-Specific Networks on Chip with Guaranteed Performance to Accelerate SOC Design and Verification", *Proceedings of the conference on Design, Automation and Test in Europe (DATE'05)*, vol. 2, 2005, pp. 1182–1187.

[37] N. Genko, D. Atienza, G. De Micheli, L. Benini, "Feature-NoC emulation: a tool and design flow for MPSoC", *IEEE Circuits and Systems Magazine*, vol. 7, no. 4, 2007, pp. 42–51.

[38] L.M. Ayough, A.H. Abutalebi, O.F. Nadjarbashi, S. Hessabi, "Verilog2SC: A Methodology for Converting Verilog HDL to SystemC", *Proceedings of the 11th International HDL Conference (HDL Con 2002)*, March 2002, pp. 211–217.

[39] Je.V. Korotkyj, O.M. Lysenko, "Visokoproduktivna model' marshrutizatora dlja merezhi-na-kristali z agregacijeju kanaliv", *Zbirnik naukovih prac' VITI NTUU "KPI"*, no. 1, 2012, p. 56–67.

[40] D. Black, J. Donovan, *SystemC: from the ground up*. USA, Boston: Kluwer Academic Publishers, 2004.

[41] H. Alemzadeh, S. Aminzadeh, "Code Optimization for Enhancing SystemC Simulation Time", *Proceedings of 2010 East-West Design & Test Symposium*, Sept. 2010, pp. 431–434.

[42] Tran A.N., *On-Chip Network Designs for Many-Core Computational Platforms: Ph.D. thesis*. USA, Davis: University of California, 2012, 156 p.

[43] A.Yu. Romanov, "Issledovanie setej na kristalle s topologiej mesh s pomoshh'ju modeli NoCTweak", *Information Technologies*, in press.