10. *Vatutin E.I., Kolyasnikov D.V., Martynov I.A., Titov V.S.* Metod sluchaynogo perebora v zadache postroeniya razbieniy graf-skhem parallel'nykh algoritmov [The method of random search in the task of building a splits graph-schemes of parallel algorithms], *Mnogoyadernye protsessory, parallel'noe programmirovanie, PLIS, sistemy obrabotki signalov* [Multi-core processors, parallel programming, FPGA, signal-processing system]. Barnaul, 2014, pp. 115-125.
11. *Vatutin E.I., Valyaev S.Yu., Dremov E.N., Martynov I.A., Titov V.S*. Raschetnyy modul' dlya testirovaniya kombinatornykh optimizatsionnykh algoritmov v zadache poiska kratchayshego puti v grafe s ispol'zovaniem dobrovol'nykh raspredelennykh vychisleniy [Calculating unit for testing combinatorial optimization algorithms in the task of finding the shortest path in a graph using voluntary distributed computing], *Svidetel'stvo o gosudarstvennoy registratsii programmy dlya EVM № 2014619797 ot 22.09.14.*

Статью рекомендовал к опубликованию д.т.н., профессор А.С. Сизов.

Ватутин Эдуард Игоревич – Юго-Западный государственный университет; e-mail: evatutin@rambler.ru; 305040, г. Курск, ул. 50 лет Октября, 94; тел.: 84712587105; кафедра вычислительной техники; к.т.н.; доцент.

Титов Виталий Семенович – e-mail: titov-kstu@rambler.ru; тел.: 84712587112; кафедра вычислительной техники; зав. кафедрой д.т.н.; профессор.

Vatutin Eduard Igorevich – Southwest State University; e-mail: evatutin@rambler.ru; 94, 50 let Oktyabrya street, Kursk, 305040, Russia; phone: +74712587105; the department of computer engineering; cand. of eng. sci.; associate professor.

Titov Vitaly Semenovich – e-mail: titov-kstu@rambler.ru; phone: +74712587112; the department of computer engineering; head of department; dr. of eng. sc.; professor.

УДК 004.382.2

В.А. Гудков, А.А. Гуленок, В.Б. Коваленко, А.И. Дордопуло

ОБЪЕКТЫ В ИЕРАРХИИ СОФТ-АРХИТЕКТУР РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ*

Описывается структура софт-архитектур, создаваемых на основе софт-процессоров и других динамически-перестраиваемых устройств и позволяющих без перезагрузки конфигурации ПЛИС, только путем программной настройки ее компонентов создавать вычислительные структуры, необходимые для решения прикладных задач. Софт-архитектуры имеют иерархическое строение. На верхнем уровне иерархии размещены софт-архитектуры (СА), состоящие из макрообъектов. Макрообъекты, находящиеся на более низком уровне, могут состоять из узлов и объектов. Объекты, расположенные внизу иерархии, являются элементарными и неделимыми устройствами, используемыми при построении более сложных элементов архитектуры – узлов и макрообъектов. Формализация описаний различных элементов СА позволила разработать язык описания софт-архитектур SADL, содержащий конструкции для описания информационного, управляющего и синхронизационного компонентов СА. Разработанные средства описания софт-архитектур позволяют сократить время отладки прикладных программ для реконфигурируемых систем в 2–3 раза по сравнению с существующими языками программирования.

Софт-архитектуры; макрообъекты; время отладки; софт-процессоры; реконфигурируемые системы; ПЛИС.

* Работа выполнена при поддержке Министерства образования и науки РФ по Соглашению с уникальным идентификатором RFMEFI57814X0006.

V.A. Gudkov, A.A. Gulenok, V.B. Kovalenko, A.I. Dordopulo

OBJECTS IN THE HIERARCHY OF SOFT-ARCHITECTURES FOR RECONFIGURABLE COMPUTER SYSTEMS

The paper contains description of structures of soft-architectures, which are based on soft-processors and other dynamically reconfigured devices which allow creation of different computing structures, required for applications, without reprogramming of FPGAs, but using only program adjustment of their components. Soft-architectures have hierarchical structure. The top hierarchical level contains soft-architectures (SA), which consist of macro-objects. Macro-objects of the lower levels can consist of nodes and objects. Objects of the lowest hierarchical level are basic and indivisible units, used for creation of more complex elements of the architecture such as nodes and macro-objects. Owing to formalization of description of various SA elements we could develop a soft-architecture description language SADL, which contains structures for description of informational, control and synchronizing SA components. The developed SA description tools help to reduce the debugging time of applications for reconfigurable systems in 2–3 times in comparison with existing programming languages.

Software-architecture; macro-objects; debugging; soft-processors; reconfigurable systems; PLD.

Введение. Современные языки описания аппаратуры не позволяют программировать вычислительные архитектуры, ограничиваясь лишь описанием их структурной составляющей, в то время как остальные составляющие архитектуры остаются недоступными для явного описания и скрыты от пользователя [1]. В отличие от языков HDL-группы новый язык программирования SADL (Soft Architecture Description Language) позволяет в едином языковом пространстве описывать структурный, синхронизационный и управляющий компоненты софт-архитектуры за счет дифференциации связей по их предназначению.

Для создания языка SADL потребовалось выполнить формализацию элементов, составляющих софт-архитектуры реконфигурируемых вычислительных систем.

1. Иерархия софт-архитектур. Софт-архитектуры, построенные на макрообъектном принципе, имеют иерархическое строение, при котором устройства, расположенные на более низких уровнях, являются элементами для построения устройств на более высоком уровне. Иерархия элементов построения софт-архитектур приведена на рис. 1.

Рис. 1. Иерархия элементов софт-архитектур

На нижнем уровне иерархии находятся объекты, которые являются элементарными и неделимыми устройствами, используемыми при построении более сложных элементов архитектуры – узлов и макрообъектов. При этом объекты являются единственными элементами архитектуры, разрабатываемыми схемотехнически. Все остальные элементы формируются при помощи языка программирования софт-архитектур SADL. Объекты являются элементарными частицами при построении узлов и макрообъектов. Как правило, при построении макрообъектов и узлов используются сле-

дующие типы объектов: процессоры, элементарные процессоры (ЭП), динамические и статические коммутаторы, контроллеры распределенной памяти (КРП), интерфейсы, функциональные устройства (ФУ), арифметико-логические устройства (АЛУ), внутренняя память, регистры, преобразователи команд [2]. Приведенный перечень типов объектов сложился на основе опыта программирования РВС и в течение пятнадцати лет не претерпел значительных изменений [2, 3]. Каждый из перечисленных типов имеет свои особенности в структуре, описании и управлении.

На следующем уровне располагаются узлы, которые могут быть сформированы из объектов и описанных ранее узлов. Узлом будем называть совокупность объектов, соединенных пространственной коммутационной средой и требующих для решения задач внешнего управляющего воздействия. В свою очередь, узлы могут выступать в качестве элементов при построении макрообъектов. Однако они не могут непосредственно участвовать в построении софт-архитектуры (СА), так как не обладают возможностью самостоятельно настраиваться на решение задач и требуют внешнего управления.

На следующем уровне располагаются макрообъекты. В качестве элементов макрообъектов выступают объекты и узлы. Формирование макрообъектов выполняется с учетом того факта, что макрообъект должен быть самодостаточным в плане управления.

2. Граф софт-архитектуры. На верхнем уровне располагаются софт-архитектуры, в качестве элементов которых могут выступать только макрообъекты [4]. СА может быть представлена в виде графа Т, вершинами которого являются макрообъекты, а дугами – информационные и синхронизирующие связи между ними:

T[M, D, S], где $M = \{m_1, m_2,\ldots,m_i\}$ – множество макрообъектов, составляющих структуру СА; $D = \{d_1, d_2, \ldots, d_j\}$ – множество информационных связей; $S = \{s_1, s_2, \ldots, s_k\}$ – множество синхронизирующих связей.

На рис. 2 представлен граф Т структурной составляющей софт-архитектуры.

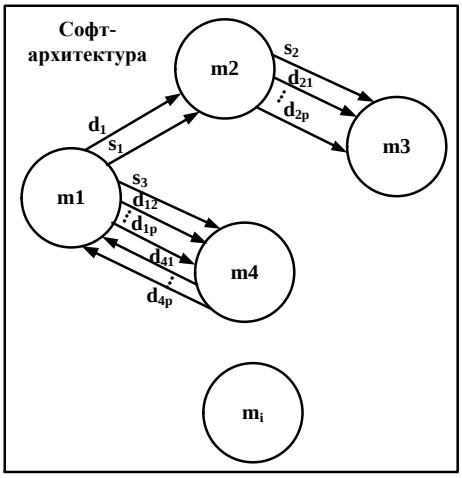


Рис. 2. Граф структурной составляющей макрообъектной софт-архитектуры

3. Структура макрообъектов. Макрообъекты, составляющие СА, могут обмениваться данными и синхронизировать свою работу посредством информационных и синхронизирующих связей. Все управляющие связи расположены внутри каждого макрообъекта и не выходят за его пределы. По сути, макрообъект представляет собой аппаратно-программную заготовку, с помощью которой может быть создана вычислительная структура, способная решать поставленную задачу.

В состав макрообъекта обязательно должны быть включены контроллеры распределенной памяти, осуществляющие управление информационными потоками в пределах макрообъекта и обращение к внутренней и внешней памяти макрообъекта. Выполнение этого требования необходимо для реализации самостоятельной настройки на решение задач.

Структура макрообъекта может быть представлена в виде графа, вершинами которого являются узлы и объекты, а дугами – связи между ними:

$$M\,[O, E, D, S, H, DI, SI, DO, SO],$$

где $O = \{o_1, o_2, \ldots, o_i\}$ – множество объектов; $E = \{e_1, e_2, \ldots, e_i\}$ – множество узлов; $D = \{d_1, d_2, \ldots, d_k\}$ – множество информационных связей; $S = \{s_1, s_2, \ldots, s_l\}$ – множество синхронизирующих связей; $H = \{h_1, h_2, \ldots, h_p\}$ – множество управляющих связей; $DI = \{di_1, di_2, \ldots, di_q\}$ – множество информационных входов макрообъекта; $SI = \{si_1, si_2, \ldots, si_{q1}\}$ – множество синхронизирующих входов макрообъекта; $DO = \{do_1, do_2, \ldots, do_r\}$ – множество информационных выходов макрообъекта; $SO = \{so_1, so_2, \ldots, so_{r1}\}$ – множество синхронизирующих выходов макрообъекта. На рис. 3 приведен пример структурной составляющей графа макрообъекта, включающий в себя в качестве вершин как объекты (O), так и узлы (E).

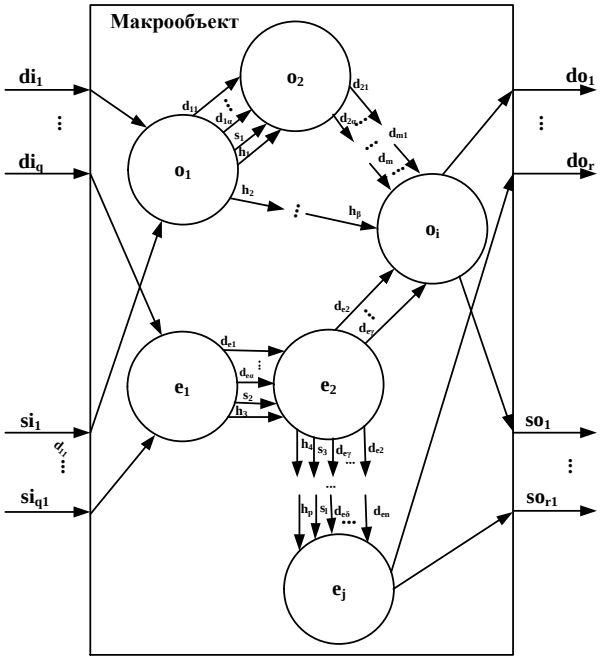


Рис. 3. Граф структурной составляющей макрообъекта

4. Элементы софт-архитектур. Элементы макрообъекта, в отличие от СА, соединены между собой тремя видами связей: информационными, синхронизирующими и управляющими. Таким образом, элементы, способные генерировать управляющие сигналы, выполняют настройку остальных элементов.

В качестве формализации описания объектов рассмотрим типовое описание функционального устройства. Объекты типа функциональных устройств (рис. 4) предназначены для исполнения единственной функции F. Они не требуют предварительной настройки или какого-либо другого управления и являются пассивными объектами. По причине того, что ФУ не имеют возможностей к перестроению, в их структуре отсутствуют входы управления.

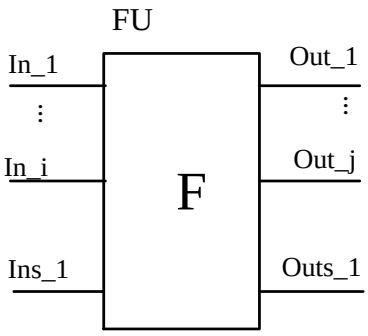


Рис. 4. Схема функциональных устройств

Функциональные устройства можно представить следующим образом:

FU[F, DI, si, DO, so],

где F – функция преобразования; DI – множество информационных входов; DO – множество информационных выходов; Ins_1 – синхронизирующий вход; Outs_1 – синхронизирующий выход.

К функциональным устройствам относятся такие объекты, как сумматоры, вычитатели, умножители и пр.

Все объекты могут быть разделены на три класса, отличающихся друг от друга подходами к их управлению. Таблица классов приведена на рис. 5.

Непрограммируемые	**Операционные**	**Процессорные**
• Функциональные устройства • Регистры	• Элементарные процессоры • Арифметико-логические устройства • Память • Статические коммутаторы	• Процессоры • Динамические коммутаторы • Контроллеры памяти • Интерфейсы

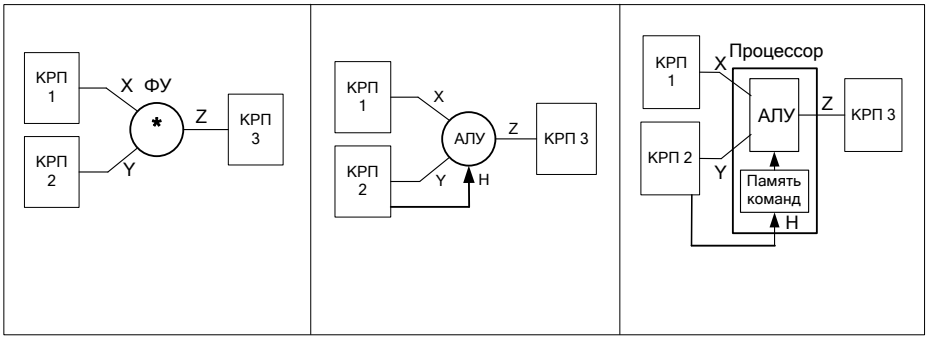
Рис. 5. Таблица классов объектов

Непрограммируемыми будем называть объекты, выполняющие одну операцию или группу операций и не имеющие возможностей к перепрограммированию. Операционные объекты имеют возможность к перепрограммированию и способны выполнять различные операции в зависимости от управляющего сигнала.

Операционные объекты являются статическими и могут перепрограммироваться перед началом работы очередного кадра задачи. Объекты, требующие предварительной настройки или управления в процессе работы, содержат управляющий вход *H,* к которому подключен выход общего интерфейса настройки. Объекты, имеющие в своем составе память команд и работающие под управлением интерфейсов, объединены в процессорном классе. Такие объекты являются динамическими и имеют возможность перепрограммирования в процессе выполнения одного кадра задачи, работая автономно под управлением собственной памяти команд.

На рис. 6,а-в приведены примеры непрограммируемого, операционного и процессорного классов соответственно.

Существенные различия в структуре и системе управления элементами СА не позволяют использовать для их описания единого формата. По этой причине для каждого из типов элементов и классов объектов предлагается использование собственного формата описания.



а б в

Рис. 6. Примеры объектов трех классов

5. Синхронизация элементов софт-архитектур. Синхронизация объектов выполняется посредствам синхронизационного компонента СА, который включает в себя синхронизационные связи и наборы регистров. Пользователь имеет возможность объединять синхронизационными связями различные элементы СА, организовывая таким образом собственную синхронизационную структуру. Синхронизация информационных потоков в пределах СА выполняется на нескольких уровнях.

Первый уровень – синхронизация потоков данных, поступающих на вход одного устройства. При этом в СА заложена возможность синхронизации данных на входах одного устройства. Подобная синхронизация возможна в пределах нескольких сотен тактов и реализуется установкой блока синхронизации на входах каждого из объектов (рис. 7).

Синхронизация на уровне входов (рис. 8) объекта позволяет архитектору не замечать незначительные сдвиги при поступлении потоков данных от различных ветвей схемы.

Вторым типом синхронизации является установка блока синхронизации посредствам языка описания СА SADL. В этом случае архитектор, анализируя структуру СА, самостоятельно устанавливает блоки регистров в требуемых местах.

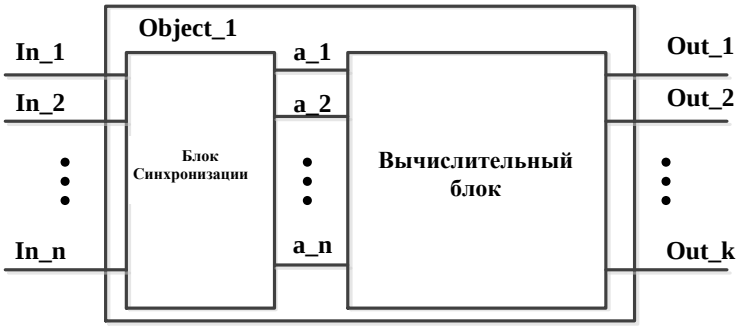


Рис. 7. Синхронизация на уровне входов одного устройства

Третьим типом синхронизации (рис. 9) является синхронизация при помощи синхронизационных входов/выходов. Подключая синхронизационные входы одних элементов к синхронизационным или информационным выходам других элементов, пользователь имеет возможность синхронизировать информационные потоки внутри софт-архитектур.

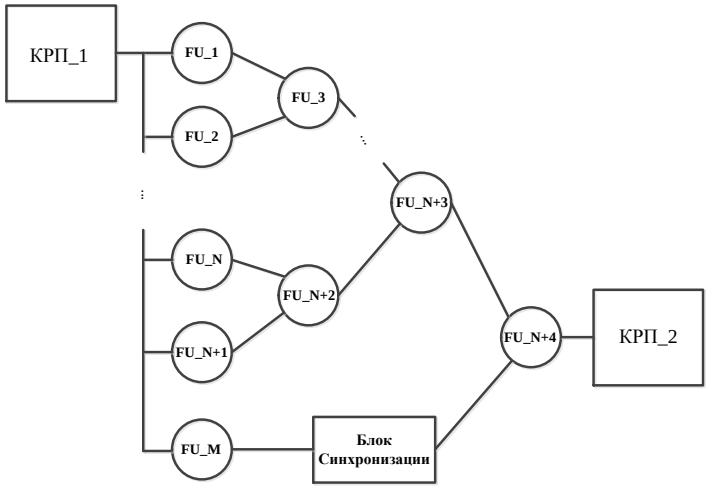


Рис. 8. Использование отдельного синхронизирующего блока

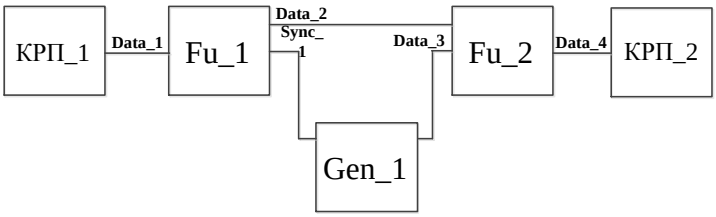


Рис. 9. Использование синхронизирующих связей

Такая синхронизация может заменить два предыдущих типа синхронизации, однако требует дополнительных связей между элементами, что не всегда является приемлемым при построении крупных софт-архитектур.

Формализация элементов, составляющих софт-архитектуры, позволяет выделить основные отличительные характеристики для каждого из типов элементов, которые найдут свое отражение в языковых конструкциях языка описания софт-архитектур. При этом из формализованного описания элементов видно, что язык описания софт-архитектур должен содержать конструкции для формирования структурного, управляющего и синхронизационного компонентов софт-архитектур.

Заключение. Применение программных средств разработки и модификации софт-архитектур не потребует привлечения высококвалифицированного специалиста-схемотехника для изменения вычислительной структуры. При этом время, затрачиваемое на создание или модификацию софт-архитектуры, значительно сокращается, а получаемые многокристальные архитектурные решения сравнимы по эффективности с решениями, выполненными специалистами-схемотехниками вручную. Автоматизация отображения графов на ресурс РВС позволяет разработчикам прикладных задач мыслить не несколькими ПЛИС, а одной виртуальной ПЛИС с большим логическим объёмом. Комплекс разработанных программных средств [5–10] позволяет программисту РВС разрабатывать и выполнять отладку прикладных параллельных программ для РВС, не вникая в особенности архитектуры РВС, а также самостоятельно создавать и модифицировать различные софт-архитектуры, ориентируясь на предметную область, которой принадлежит решаемая задача.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Зотов В.Ю.* Проектирование встраиваемых микропроцессорных систем на основе ПЛИС фирмы Xilinx. – М.: Горячая линия-Телеком, 2006. – 520 с.
2. *Каляев И.А., Левин И.И., Семерников Е.А., Шмойлов В.И.* Реконфигурируемые мультиконвейерные вычислительные структуры. Монография / Под общ. ред. И.А. Каляева. – 2-е изд., перераб. и доп. – Ростов-на-Дону: Изд-во ЮНЦ РАН, 2009. – 344 с.
3. *Kalyaev I.A. (Ed.), Levin I.I., Semernikov E.A., Shmoilov V.I.* Reconfigurable Multipipeline Computing Structures. Nova Science Publishers, Inc. New York, USA. – 330 p.
4. *Семерников Е.А., Коваленко В.Б.* Организация многоуровневого программирования реконфигурируемых вычислительных систем // Вестник компьютерных и информационных технологий. – 2011. – № 9. – С. 3-10.
5. *Kalyaev I.A., Levin I.I., Dordopulo A.I., Slasten L.M.* Reconfigurable Computer Systems Based on Virtex-6 and Virtex-7 FPGAs // IFAC Proceedings Volumes (ISSN 14746670), Programmable Devices and Embedded Systems. – 2013. – Vol. 12. –Part № 1. – P. 210-214.
6. *Kalyaev I.A., Levin I.I. Dordopulo A.I., Slasten L.M.* FPGA-based Reconfigurable Computer Systems // Science and Information Conference (SAI) 2013, Oct 7, 2013 - Oct 9, 2013, London, UK. – P. 148-155 (Scopus).
7. *Gudkov V.A., Gulenok A.A., Kovalenko V.B., Slasten L.M.* Multi-level Programming of FPGA-based Computer Systems with Reconfigurable Macroobject Architecture // IFAC Proceedings Volumes (ISSN 14746670), Programmable Devices and Embedded Systems. – 2013. – Vol. № 12, part № 1. – P. 204-209.
8. *Левин И.И., Дордопуло А.И., Каляев И.А., Гудков В.А.* Высокопроизводительные реконфигурируемые вычислительные системы на основе ПЛИС VIRTEX-7 // Труды Международной научной конференции «Параллельные вычислительные технологии (ПаВТ’2014)», Ростов-на-Дону, 1–3 апреля 2014 г. – Челябинск: Издательский центр ЮУрГУ, 2014. – С. 131-139.
9. *Левин И.И., Дордопуло А.И., В.Б. Коваленко, Гудков В.А., Гуленок А.А.* Программирование реконфигурируемых вычислительных систем на основе ПЛИС VIRTEX-7 с использованием софт-архитектур // Труды Международной суперкомпьютерной конференции «Научный сервис в сети Интернет: многообразие суперкомпьютерных миров». (22-27 сентября 2014 г., г. Новороссийск). – М.: Изд-во МГУ, 2014. – С. 296-300.
10. *Levin I.I., Kalyaev I.A., Dordopulo A.I., Slasten L.M., Gudkov S.V.* Virtex-7 FPGA-based Reconfigurable Computer System RCS-7 for digital image processing // 18th International Conference on Circuits, Systems, Communications and Computers (CSCC 2014), Santorini, Greece. – P. 146-149.

REFERENCES

1. *Zotov V.Yu.* Proektirovanie vstraivaemykh mikroprotsessornykh sistem na osnove PLIS firmy Xilinx [The design of embedded microprocessor systems based on FPGA company Xilinx]. Moscow: Goryachaya liniya-Telekom, 2006, 520 p.
2. *Kalyaev I.A., Levin I.I., Semernikov E.A., Shmoylov V.I.* Rekonfiguriruemye mul'tikonveyernye vychislitel'nye struktury [Reconfigurable multiconference computational patterns]. Monografiya. 2nd ed., pererab. i dop. Rostov-on-Don: Izd-vo YuNTs RAN, 2009, 344 p.
3. *Kalyaev I.A. (Ed.), Levin I.I., Semernikov E.A., Shmoilov V.I.* Reconfigurable Multipipeline Computing Structures. Nova Science Publishers, Inc. New York, USA, 330 p.
4. *Semernikov E.A., Kovalenko V.B.* Organizatsiya mnogourovnevogo programmirovaniya rekonfiguriruemykh vychislitel'nykh sistem [Organization multi-level programming of reconfigurable computing systems], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Bulletin of the computer and information technology], 2011, No. 9, pp. 3-10.
5. *Kalyaev I.A., Levin I.I., Dordopulo A.I., Slasten L.M.* Reconfigurable Computer Systems Based on Virtex-6 and Virtex-7 FPGAs, *IFAC Proceedings Volumes (ISSN 14746670), Programmable Devices and Embedded Systems*, 2013, Vol. 12, Part No. 1, pp. 210-214.
6. *Kalyaev I.A., Levin I.I. Dordopulo A.I., Slasten L.M.* FPGA-based Reconfigurable Computer Systems, *Science and Information Conference (SAI) 2013, Oct 7, 2013 - Oct 9, 2013, London, UK*, pp. 148-155 (Scopus).

7. *Gudkov V.A., Gulenok A.A., Kovalenko V.B., Slasten L.M.* Multi-level Programming of FPGA-based Computer Systems with Reconfigurable Macroobject Architecture, *IFAC Proceedings Volumes (ISSN 14746670), Programmable Devices and Embedded Systems*, 2013, Vol. 12, Part No. 1, pp. 204-209.
8. *Levin I.I., Dordopulo A.I., Kalyaev I.A., Gudkov V.A.* Vysokoproizvoditel'nye rekonfiguriruemye vychislitel'nye sistemy na osnove PLIS VIRTEX-7 [High-performance reconfigurable computing system based on FPGA VIRTEX-7], *Trudy mezhdunarodnoy nauchnoy konferentsii «Parallel'nye vychislitel'nye tekhnologii (PaVT'2014)», Rostov-na-Donu, 1–3 aprelya 2014 g* [Proceedings of international scientific conference "Parallel computing technologies (Pushchino'2014)", Rostov-on-don, April 1-3, 2014]. Chelyabinsk: Izdatel'skiy tsentr YuUrGU, 2014, pp. 131-139.
9. *Levin I.I., Dordopulo A.I., V.B. Kovalenko, Gudkov V.A., Gulenok A.A.* Programmirova-nie rekonfiguriruemykh vychislitel'nykh sistem na osnove PLIS VIRTEX-7 s ispol'-zovaniem soft-arkhitektur [Programming of reconfigurable computing systems based on FPGA VIRTEX-7 with the use of software-architectures], *Trudy Mezhdunarodnoy superkomp'yuternoy konferentsii «Nauchnyy servis v seti Internet: mnogoobrazie superkomp'yuternykh mirov». (22-27 sentyabrya 2014 g., g. Novorossiysk)* [Proceedings of the International supercomputer conference "Scientific service in the Internet: diversity supercomputer worlds". (22-27 September 2014, , Novorossiysk)]. Moscow: Izd-vo MGU, 2014, pp. 296-300.
10. *Levin I.I., Kalyaev I.A., Dordopulo A.I., Slasten L.M., Gudkov S.V.* Virtex-7 FPGA-based Reconfigurable Computer System RCS-7 for digital image processing, *18th International Conference on Circuits, Systems, Communications and Computers (CSCC 2014), Santorini, Greece*, pp. 146-149.

Статью рекомендовал к опубликованию д.т.н., профессор Я.Е. Ромм.

Гудков Вячеслав Александрович – НИИ многопроцессорных вычислительных систем ЮФУ; e-mail: gudkov@mvs.tsure.ru; 347928, г. Таганрог, ул. Чехова, 2; тел.; 88634315491; к.т.н.; с.н.с.

Гуленок Андрей Александрович – e-mail: andrei_gulenok@mail.ru; к.т.н.; с.н.с.

Коваленко Василий Борисович – Южный научный центр РАН; e-mail: vereten@hotbox.ru; тел.: 88634318910; отдел информационных технологий и процессов управления; к.т.н.; м.н.с.

Дордопуло Алексей Игоревич – Учреждение Российской академии наук Южный научный центр РАН; e-mail: scorpio@mvs.tsure.ru; 347900, г. Таганрог, 10-й переулок, 114/1, кв. 6; тел.: 88634315491; к.т.н.; с.н.с.

Gudkov Vyacheslav Aleksandrovitch – SRI MCS SFU; e-mail: gudkov@mvs.tsure.ru; 2, Chekhov street, GSP-284, Taganrog, 347928, Russia; phone: +78634315491; cand. of eng. sc.; senior staff scientist.

Gulenok Andrey Aleksandrovitch – e-mail: andrei_gulenok@mail.ru; cand. of eng. sc.; senior staff scientist.

Kovalenko Vasiliy Borisovitch – Southern scientific centre of the RAS; e-mail: vereten@hotbox.ru; phone: +78634318910; the department of information technologies and control processes cand. of eng. sc.; junior scientific associate.

Dordopulo Alexey Igorevich – Southern Scientific Centre of the Russian Academy of Sciences; e-mail: scorpio@mvs.tsure.ru; 10th lane, 114/1, ap. 6 Taganrog, 347900, Russia; phone: +78634315491; cand. of eng. sc.; senior staff scientist.