# Аппаратно-ориентированный генетический алгоритм синтеза конечных автоматов

*М.В. Ляшов, А.Н. Береза, С.А. Коцюбинская*

*Институт сферы обслуживания и предпринимательства (филиал) ДГТУ, Шахты*

**Аннотация:** В статье приведен аппаратно-ориентированный генетический алгоритм синтеза конечных автоматов. Тестирование предлагаемого генетического алгоритма показало, что его использование позволяет повысить качество решения задач синтеза конечных автоматов в сочетании с уменьшением времени получения квазиоптимального решения (по сравнению с известными в этой области алгоритмами). Также описывается аппаратно-программная база, используемая для построения эволюционных аппаратных средств. Работа системы продемонстрирована на примере тестовой задачи синтеза конечных автоматов в задаче об «Умном муравье».
**Ключевые слова:** эволюционный синтез, конечный автомат, генетические алгоритмы, эволюционные алгоритмы, эволюционная электроника.

## Введение

В настоящее время в связи с ускорением темпов технического прогресса в условиях жесткой конкуренции необходимо обеспечить высокое качество проектируемых устройств при минимальных временных затратах. На сегодняшний день при решении сложных задач проектирования невозможно обойтись без использования систем автоматизированного проектирования (САПР). Современные САПР — это многоаспектные и многоуровневые системы, которые включают в себя комплекс программных и аппаратных средств. [1]

На основании исследования научных трудов и опубликованных за последние десять лет патентов [2,3] можно утверждать, что в последнее время в САПР для проектирования цифровых и аналоговых устройств применяются эволюционные алгоритмы (ЭА) [4]. Это направление получило название эволюционная электроника (Evolutionary Electronics) [5]. Применение ЭА на аппаратных платформах, имеющих реконфигурируемые элементы, позволяющие перестраивать систему во время функционирования, получило название эволюционные аппаратные средства (ЭАС) [6-7].

В качестве реконфигурируемой элементной базы для построения ЭАС применяются программируемые логические интегральные схемы (ПЛИС) [8]. При создании цифровых ЭАС в качестве реконфигурируемой части выступают динамически перестраиваемые комбинационные или последовательностные логические схемы. Применяемые в настоящее время методы синтеза конечных автоматов всегда используют специфику решаемой задачи, делая полученную технику генерации автоматов неприменимой к остальным задачам. Возникает вопрос создания универсальных методов синтеза конечных автоматов, применимых к широкому кругу задач. В работах [9,10] показано применение эволюционных алгоритмов для синтеза конечных автоматов. Однако представленные алгоритмы применяются для автоматного программирования, в рамках которого программа описывается с помощью конечных детерминированных автоматов, что не позволяет их использовать в автономных аппаратных системах на реконфигурируемых платформах.

**Постановка задачи эволюционного синтеза конечных автоматов**

Задачу эволюционного синтеза конечных автоматов определим в виде множества:

$$R = \{H, O, F\}$$

где $H$ – генотип синтезируемого решения, $O$ – генетические операторы $O = \{o_1, o_2, \ldots o_n\}$, $F$ – целевая функция.

Генотип синтезируемого решения определяется множеством $H = \{g_1, g_2, \ldots g_p\}$, где $p = S \cdot 2^x$, $S$ – количество состояний конечного автомата, $x$ – количество входов.

Целевая функция определяется выражением $F = w_1 a_1 + w_2 a_2$, где $a_1$ – количество состояний, $a_2$ – количество итераций, $w$ – весовые коэффициенты частных критериев.

Задачей генетического алгоритма является минимизация целевой функции, т.е. $F \to min$.

## Предлагаемый алгоритм

Поскольку предлагаемый генетический алгоритм предназначен для работы в автономных ЭАС, то для уменьшения аппаратных ресурсов ПЛИС хромосомы кодируются битовой строкой. Вместо комбинационной логики, построенной на ЛЭ, в структурной схеме конечного автомата используется блок встроенной памяти ПЛИС (рис.1).

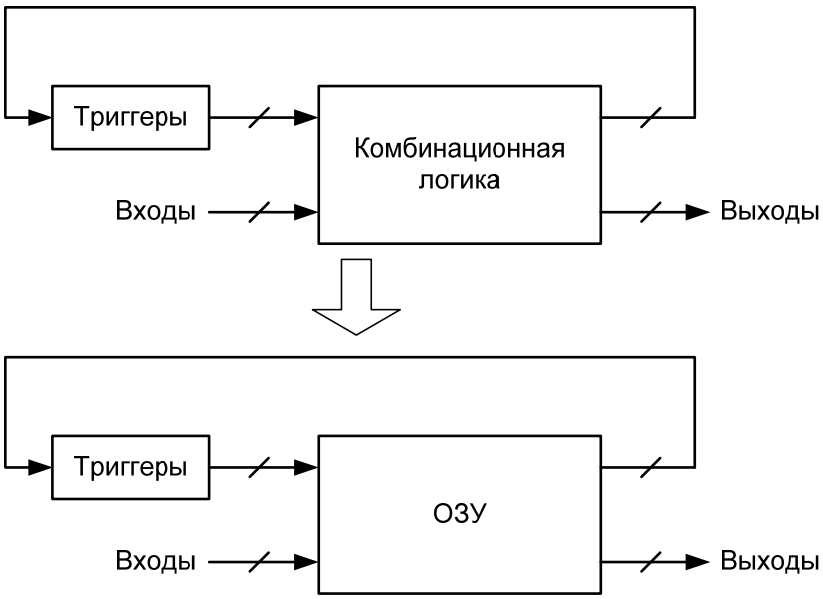


Рис. 1. – Использование памяти в КА

Хромосома кодируется путем преобразования таблицы истинности ОЗУ в битовую строку. Сначала таблица переходов конечного автомата упорядочивается в порядке возрастания начального состояния, а в случае равенства – по номеру входного воздействия. Далее в строчку записываются

значения таблицы истинности ОЗУ. Для таблицы переходов конечного автомата это будет соответствовать новому состоянию, в которое переходит КА и значению выхода.

После генерации нового поколения, а также применения генетических операторов кроссинговера и мутации, переходы и значения выходов КА меняются случайным образом. При этом в конечном автомате могут возникать состояния, в которые при любых последовательностях значений входных переменных невозможно попасть из начального состояния. Также возможно, что из текущего состояния при любых значениях на входе автомата, он всегда переходит в одно и тоже состояние. Для исправления таких ситуаций разработан алгоритм корректировки переходов, состоящий из следующих шагов:

Шаг 1. Формируется список доступных состояний ($N$) с использованием алгоритма рекурсивного обхода графа.

Шаг 2. Выполняется цикл по всем состояниям. Если из текущего состояния при любых значениях на входе автомата, он всегда переходит в одно и тоже состояние ($j^{\forall x} \to a$), то случайным образом выбирается переход этого состояния и состояние, на которое он переходит, меняется на следующее ($P \to j^{+}$).

Шаг 3. Если текущее состояние не входит в список доступных ($j \notin N$), то для него выполняются следующие операции:

Шаг 3.a. Случайным образом выбирается состояние из списка доступных состояний ($i - N_i$).

Шаг 3.b. Один из переходов выбранного состояния ($P_i$) выбирается случайным образом и заменяется на переход, который ведет в текущее состояние ($P_i - P_j$).

Шаг 3.c. Обновляется список доступных состояний. В него добавляется текущее состояние и все состояния, в которые можно попасть из него.

Основными генетическими операторами, используемыми при работе генетического алгоритма синтеза конечных автоматов, являются операторы кроссинговера и мутации. Кодирование хромосомы битовой строкой накладывает ограничения на типы применяемых операторов.

Оператор мутации представляет собой процесс изменения определенных генов в хромосоме и предназначен для формирования нового генетического материала. При этом в заданную популяцию вводятся радикально отличающиеся решения, которые отсутствовали в исходной популяции или были удалены в результате выполнения отбора. Эти новые решения придают популяции необходимое разнообразие и позволяют расширить пространство поиска, тем самым уменьшая время нахождения оптимума [9].

Оператор мутации является случайным, т.е. не зависит ни от степени приспособленности хромосомы, ни от гена, находящегося в хромосоме. В результате применения оператора мутации, случайным образом определяется, что будет изменяться в конечном автомате:

- значение выходной переменной, генерируемое КА (рис. 2);
- номер состояния, в которое переключается автомат по случайно выбранному переходу (рис. 3).

Алгоритм модифицированного оператора мутации состоит из следующих шагов:

Шаг 1. Генерируется случайное число S, равномерно распределённое в интервале [0; $n$], где $n$ – число переходов конечного автомата;

Шаг 2. Случайно выбирается, что будет модифицироваться – значение выходной переменной, генерируемое КА или номер состояния, в которое переключается автомат по выбранному переходу:

Шаг 2.a. Если было определено, что изменяется значение выходной переменной, то оно случайным образом изменяется в переходе с индексом S.

Шаг 2.b. Если было определено, что изменяется номер состояния, то он случайным образом изменяется в переходе с индексом S.

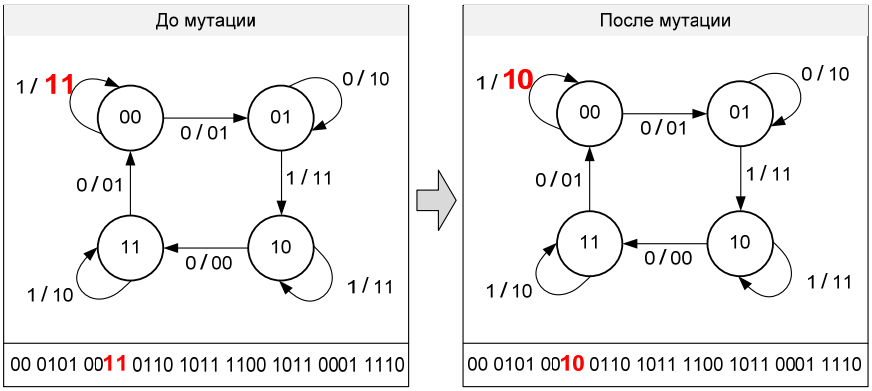


Рис. 2. – Изменение оператором мутации значения выходной переменной КА

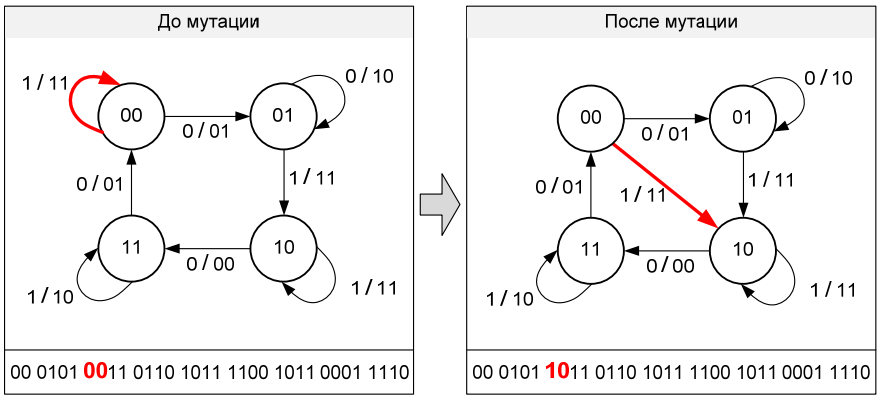


Рис. 3. – Изменение оператором мутации номер состояния, в которое переключается КА

Оператор кроссинговера случайным образом производит обмен генетической информацией между двумя вариантами решений. При этом в популяции сохраняется уже существующая наследственная информация. Качество решений, получаемых генетическим алгоритмом, во многом зависит от выбора типа применяемого оператора кроссинговера.

В разработанном алгоритме синтеза КА были применены одноточечный и двухточечный операторы кроссинговера, поскольку они наиболее просто реализуются аппаратно [4-6]. Экспериментальные исследования показали, что наиболее предпочтительным является двухточечный кроссинговер. Рассмотрим работу двухточечного оператора кроссинговера, показанного на рис. 4. Из популяции выбираются две хромосомы: «Родитель 1» и «Родитель 2». Затем случайным образом определяются две точки разрыва хромосом, и происходит обмен. В результате работы оператора кроссинговера получаются две новые хромосомы: «Потомок 1» и «Потомок 2».

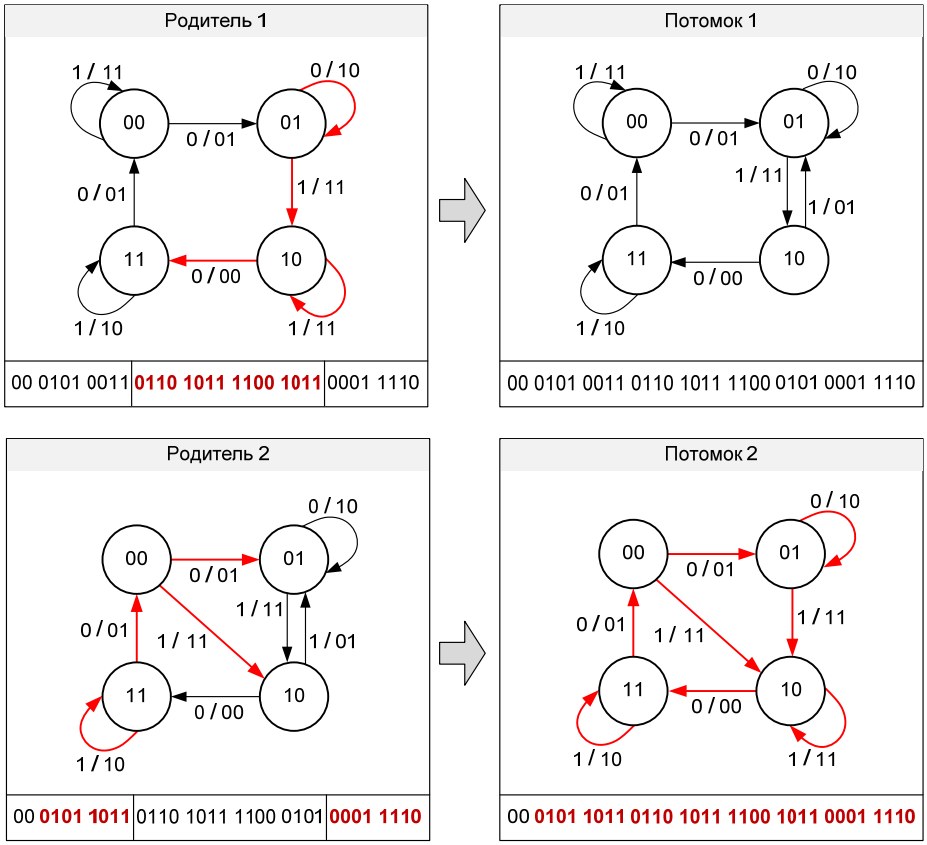


Рис. 4. – Пример работы оператора кроссинговера

Алгоритм функционирования оператора отбора выполнен на основе алгоритма сортировки пузырьком, поскольку при его реализации требуется меньше аппаратных ресурсов по сравнению с другими алгоритмами

сортировок [7]. После выполнения сортировки популяции по убыванию (т.е. хромосомы с большим значением целевой функции перемещаются в верхнюю часть популяции) из популяции удаляется 3 хромосомы (поскольку после оператора кроссинговера и оператора мутации к популяции добавляется еще 3 хромосомы) с меньшим значением целевой функции.

В генетическом алгоритме синтеза конечных автоматов применен стандартный оператор турнирной селекции, поскольку при его реализации требуется меньше аппаратных ресурсов по сравнению с другими типами селекций. При турнирной селекции формируется случайное подмножество из элементов популяции и среди них выбирается один элемент с наибольшим значением целевой функции.

## Экспериментальные исследования

Для применения разработанного алгоритма генетического алгоритма синтеза конечных автоматов в автономных эволюционных аппаратных системах его необходимо реализовать аппаратно. Блок схема аппаратно реализованного генетического алгоритма приведена на рис. 5.

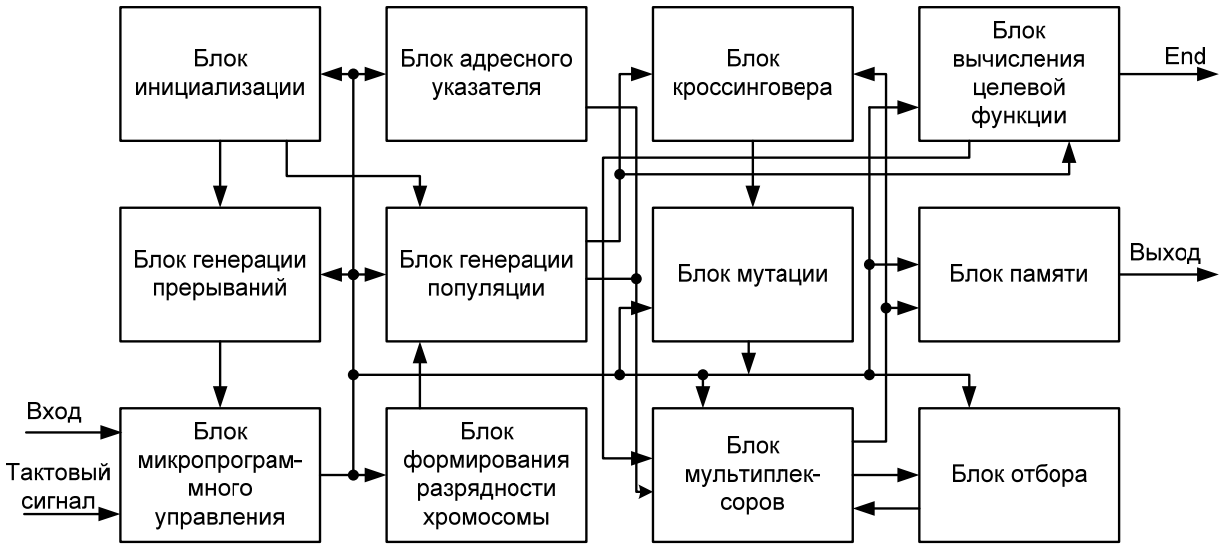


Рис. 5. – Блок схема аппаратно реализованного генетического алгоритма синтеза конечных автоматов

Устройство выполнено в виде единого кристалла ПЛИС. Микропрограммный принцип управления с возможностью настройки параметров функционирования устройства в соответствии со значениями, необходимыми пользователю, дает возможность динамических изменений параметров функционирования и перенастройки принципа функционирования любого блока, не влияя на работу всего устройства в целом.

В качестве тестового примера рассмотрим задачу об «Умном муравье» [11-12]. Муравей находится на поверхности тора размером 32 на 32 клетки (рис. 6). В нескольких клетках (обозначены на рис. 6 черным цветом) находится еда. Она расположена вдоль некоторой ломаной, но не во всех ее клетках. Клетки ломаной, в которых нет еды, обозначены серым цветом. Белые клетки не содержат еду и не принадлежат ломаной. Всего на поле 89 клеток с едой.

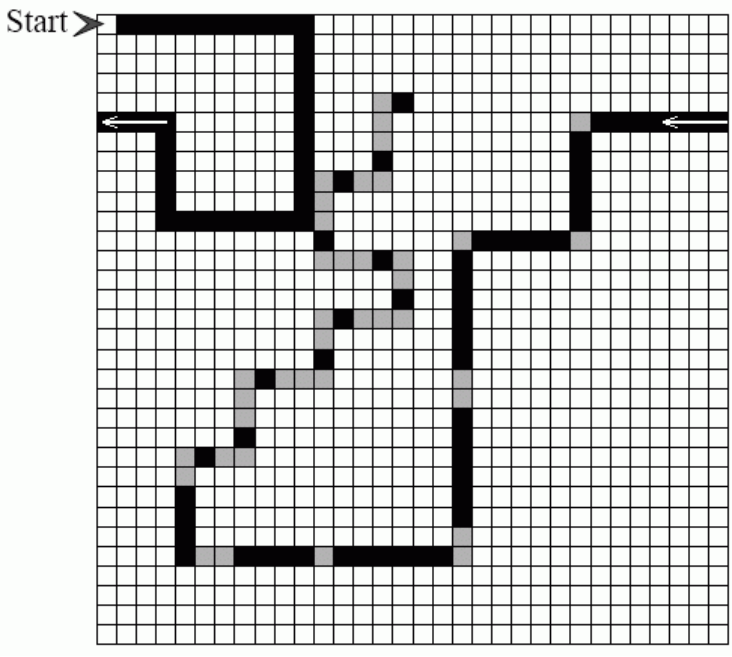


Рис. 6. – Поле в задаче об «Умном муравье»

Исходное положение муравья показано меткой Start. Он занимает одну клетку и смотрит в одном из четырех направлений (вперед, назад, налево, направо).

Муравей умеет определять, находится ли непосредственно перед ним еда. За один игровой ход муравей может совершить одно из четырех действий:

- сделать шаг вперед, съедая еду, если она там находится;
- повернуть налево;
- повернуть направо;

Съеденная муравьем еда не восполняется, муравей жив на протяжении всей игры, еда не является необходимым ресурсом для его жизни. Ломаная не случайна, а строго фиксирована. Муравей может ходить по любым клеткам поля.

Игра длится 200 ходов, на каждом из которых муравей совершает одно из четырех действий. По истечении 200 ходов подсчитывается количество еды, съеденной муравьем. Это значение и есть результат игры.

Цель игры – создать муравья, который за 200 ходов съест как можно больше еды (желательно, все 89 единиц).

Один из способов описания поведения муравья — автомат Мили, у которого имеется одна входная переменная (находится ли еда перед муравьем), а множество выходных воздействий состоит из четырех упомянутых выше элементов.

Автомат, решающий описанную задачу, трудно построить эвристическими методами. Например, эвристически построенный автомат Мили с пятью состояниями задачу не решает [13]. Конечный автомат описывает поведение муравья, который съедает всего 81 единицу еды за 200 ходов, а всю еду — только за 314 ходов.

Экспериментальные исследования для тестовой задачи об «Умном муравье» проводились со следующими параметрами ГА: размер популяции 1200, вероятность применения оператора кроссинговера 0.40 и вероятность применения оператора мутации – 0.25. В таблице 1 приведено сравнение результатов синтеза КА разработанного аппаратно-ориентированного генетического алгоритма синтеза конечных автоматов (ГАСКА) с результатами эвристического алгоритма [13] и результатом ГА, представленного в работе [12].

Таблица № 1

Результаты эксперимента

| Алгоритм | Кол-во состояний | Кол-во ходов | Время синтеза, сек. |
|---|---|---|---|
| Эвристический | 5 | 314 | - |
| Аналог | 7 | 198 | 269 |
| ГАСКА | 7 | 198 | ***29*** |

Как следует из приведенных результатов эксперимента, с помощью разработанного генетического алгоритма синтеза конечных автоматов удалось найти конечный автомат с 7 состояниями который за 198 ходов решает задачу. При этом время синтеза КА в 9 раз меньше, чем у существующих аналогов.

## Выводы

Постоянное увеличение, согласно закону Мура, количества транзисторов на кристалле СБИС ставит перед проектировщиками и создателями систем автоматизированного проектирования все более сложные задачи, которые должны решаться в сжатые сроки. Применение методов эволюционного поиска в системах автоматизированного проектирования микроэлектронных средств позволит находить квазиоптимальные решения

задач, которые являются труднорешаемыми для классических методов. Использование аппаратного ускорения для процедур эволюционного поиска позволит значительно увеличить производительность создаваемых САПР. В дальнейшем планируется создание аппаратного ускорителя для увеличения быстродействия процедур эволюционного моделирования [14, 15].

*Работа выполнена при поддержке РФФИ (проекты № 17-07-01323, № 18-07-01054)*

## Литература

1. Р. Точчи, Рональд, Дж. Точчи, Нил. С. Уидмер. Цифровые системы. Теория и практика. 8 - е изд.. пер. с англ. М.: Вильямс, 2004. 1024 с.

2. Европейская патентная классификация : информационная система. URL: worldwide-i.espacenet.com

3. Российская патентная классификация : информационная система. URL: new.fips.ru

4. Marley Maria Bernard Vellasco, Ricardo Salem Zebulum, and Marco Aurelio Pacheco Evolutionary. Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms (1st ed.). 2001. CRC Press, Inc., Boca Raton, FL, USA. 320 p.

5. R. Moreto, C. Thomaz, S. Gimenez. AGSPICE: A New Analog ICs Design Tool Based On Evolutionary Electronics Used For Extracting Additional Design Recommendations. 8th International Caribbean Conference on Devices, Circuts and Systems (ICCDCS), 2012. pp. 143-148.

6. X. Yan, et al. Electronic Circuit Automatic Design Based on Genetic Algorithms // Procedia Engineering. Volume 15, 2011. pp. 2948-2954.

7. Nedjah N. Evolvable Machines: Theory and Practice. Studies in Fuzziness and Soft Computing. 2004. 260 p.

8. P.A. Simpson. FPGA design: Best practices for team-based reuse, second edition. Cham : Springer International Publishing : Imprint: Springer, 2015. 257 p.

9. Лобанов П.Г. Использование генетических алгоритмов для генерации конечных автоматов: дис. канд. тех. наук. Санкт-Петербург, 2008. 114 с.

10. Поликарпова Н. И., Точилин В. Н., Шалыто А. А. Применение генетического программирования для реализации систем со сложным поведением // Сборник трудов IV-ой Международной научно-практической конференции «Интегрированные модели и мягкие вычисления в искусственном интеллекте». Том 2. М.: Физматлит. 2007, С. 598–604.

11. Шалыто А. А. Технология автоматного программирования // Труды первой Всероссийской конференции «Методы и средства обработки информации». МГУ. 2003, С. 150-152.

12. Давыдов А. А., Соколов Д. О., Царев Ф. Н., Шалыто А. А. Применение островного генетического алгоритма для построения автоматов Мура и систем взаимодействующих автоматов Мили на примере задачи об «умном муравье» // Сборник докладов XI международной конференции по мягким вычислениям и измерениям (SCM'2008). СПб.: СПбГЭТУ. 2008. С. 266-270.

13. Царев Ф. Н., Шалыто А. А. О построении автоматов с минимальным числом состояний для задачи об «умном муравье» // Сборник докладов X международной конференции по мягким вычислениям и измерениям. СПбГЭТУ "ЛЭТИ". Т.2. 2007. С. 88–91.

14. Бегляров В.В., Берёза А.Н. Гибридный эволюционный алгоритм решения систем линейных алгебраических уравнений, описывающих электрические цепи. Инженерный вестник Дона, 2013, №1 URL: ivdon.ru/magazine/archive/n1y2013/1540

15. Антонова А.С., Аксенов К.А. Многокритериальное принятие решений в условиях риска на основе интеграции мультиагентного, имитационного, эволюционного моделирования и численных методов. Инженерный вестник Дона, 2012, №4 (часть 2) URL: ivdon.ru/magazine/archive/n4p2y2012/1466

## References

1. R. Tochchi, Ronal'd, Dzh. Tochchi, Nil. S. Uidmer. Cifrovye sistemy. Teoriya i praktika [Digital Systems: Principles and Applications]. 8 V. transl. M.: Vil'yams, 2004. 1024 p.

2. Evropejskaya patentnaya klassifikaciya [European Patent Classification: Information System]. URL: worldwide-i.espacenet.com

3. Rossijskaya patentnaya klassifikaciya [Russian patent classification: information system]. URL: new.fips.ru

4. Marley Maria Bernard Vellasco, Ricardo Salem Zebulum, and Marco Aurelio Pacheco Evolutionary. Electronics: Automatic Design of Electronic Circuits and Systems by Genetic Algorithms. 320 p.

5. R. Moreto, C. Thomaz, S. Gimenez. AGSPICE: A New Analog ICs Design Tool Based On Evolutionary Electronics Used For Extracting Additional Design Recommendations. 8th International Caribbean Conference on Devices, Circuts and Systems (ICCDCS), 2012. pp. 143-148.

6. X. Yan, et al. Procedia Engineering. Volume 15, 2011. pp. 2948-2954.

7. Nedjah N. Evolvable Machines: Theory and Practice. Studies in Fuzziness and Soft Computing. 2004. 260 p.

8. P.A. Simpson. FPGA design: Best practices for team-based reuse, second edition. Cham: Springer International Publishing : Imprint: Springer, 2015. 257 p.

9. Lobanov P.G. Ispol'zovanie geneticheskih algoritmov dlya generacii konechnyh avtomatov [Using genetic algorithms to generate finite state machines]: dis. kand. tekh. nauk. Sankt-Peterburg, 2008. 114 p.

10. Polikarpova N. I., Tochilin V. N., SHalyto A. A. Sbornik trudov IV-oj Mezhdunarodnoj nauchno-prakticheskoj konferencii “Integrirovannye modeli i myagkie vychisleniya v iskusstvennom intellekte” : trudy (International Symp. “Integrated models and soft computing in artificial intelligence”) vol. 2. M.: Fizmatlit. 2007, p. 598–604.

11. Shalyto A. A. Trudy pervoj Vserossijskoj konferencii “Metody i sredstva obrabotki informacii” : trudy (“Methods and means of information processing”). MGU. 2003, p. 150-152.

12. Davydov A. A., Sokolov D. O., Carev F. N., SHalyto A. A. Sbornik dokladov XI mezhdunarodnoj konferencii po myagkim vychisleniyam i izmereniyam : trudy (“ The XI Inernational Symp. at soft computing and measurements) (SCM'2008). SPb.: SPbGEHTU. 2008. p. 266-270.

13. Tsarev F. N., Shalyto A. A. Sbornik dokladov X mezhdunarodnoj konferentsii po myagkim vychisleniyam i izmereniyam : trudy (“ The XI Inernational Symp. at soft computing and measurements) SPbGEHTU "LEHTI". vol. 2. 2007. p. 88–91.

14. Beglyarov V.V., Beryoza A.N. Inzhenernyj vestnik Dona (Rus), 2013, №1. URL: ivdon.ru/magazine/archive/n1y2013/1540

15. Antonova A.S., Aksenov K.A. Inzhenernyj vestnik Dona (Rus), 2012, №4. (part 2) URL: ivdon.ru/magazine/archive/n4p2y2012/1466