UDC 519.1(075.8)+510.6(075:8)

S.F. TYURIN, A.V. GREKOV, O.A. GROMOV, I.S. PONUROVSKIY

Perm National Research Politechnical University, Russia

ADAPTABLE LOGICAL FPGA-ELEMENTS

The paper proposes the restoration of FPGA (field-programmable gate array) logic for critical applications by adapting to failures of logical elements. The principle of adaptation FPGA is to switch to residual features LUT (Look Up Table), with the possibility of hardware and software to be used in case of hardware failure after massive failures. In the case of failure of hardware (logic elements) after massive failures, for example, in catastrophic situations, it is also possible software-hardware utilization failed elements. In addition, it is useful to explore the possibility of using partial functionality for diagnosing FPGA.

Keywords: *adaptation, failures, field-programmable gate array, look up table, logic element, fault tolerance, software-hardware solution.*

Introduction

Modern FPGA, containing several billion of transistors [1], provide wide opportunities for logic reconfiguration, but do not use them to adapt to failures. Thus, one of the leading experts in FPGA area Yervant Zorian said: "Now the main problem of system on a chip repair is development of embedded technologies and methods of the logic repair that occupies no more than 10% of chip area" [2].

To solve this problem we may provide retaining of basic in the terms of Post theorem [3] logic functions that allow to calculate the input for a longer time at a given pattern of failures, that is - the reservation bases elements, the use of elements with an excess basis [4, 5].

In case of failures it is possible to calculate the initial logic functions – all or only critical parts of the residual bases of all or a subset of items with the possible use of software and hardware implementation [6]. With that the scheme is adapted to the conditions of a fault with the appropriate reconfiguration.

Contemporary programmable logic – FPGA (field-programmable gate array) provide wide opportunities of logic reconfiguration, but do not use them to adapt to the failures and logic recovery [13].

Let us consider the proposed principle and characteristics of recovery logic FPGA for critical applications by adapting to failures of logic elements.

1. The principle of adaptation to failure of 8-1 multiplexer

Let us consider the gate FPGA - multiplexer with three address inputs x_1, x_2, x_3 – 8 channels a, b, c, d, e, f, g, h, (8-1), consisting of seven elementary multiplexers 2-1 (Fig. 1).

On the assumption that there is not a single failure data inputs a, b, c, d, e, f, g, h, or not more than one failure in seven elementary multiplexers 2-1 propose the switch to the "half" of the scheme.

Let there be a failure in the element, which is connected to the input channels c, d. Then it is necessary to do after finding out the transition to the second half of the scheme – channels e, f, g, h. And the failures may occur on the input – but not of the last element.

When a fault is detected, for example, by external means, it is necessary to perform two tests – on the one and the other half. But this is allowed only in case of failure of elements and data inputs (one failure).

If there is half the items (allow refusal on the inputs of all the elements and even by choice but out on the exit of the last element), for example, the older variable is equal to zero:

$$z = \bar{0}\ \overline{\bar{x}_2(cx_1 \vee dx_1) \vee x_2(ax_1 \vee bx_1)} \vee \bar{0}\ \overline{\bar{x}_2(gx_1 \vee hx_1) \vee x_2(ex_1 \vee fx_1)} \quad (1)$$

Then we get:

$$z = \bar{0}\ \overline{\bar{x}_2(cx_1 \vee dx_1) \vee x_2(ax_1 \vee bx_1)} \quad (2)$$

or

$$z_1 = \overline{\bar{x}_2(cx_1 \vee dx_1) \vee x_2(ax_1 \vee bx_1)} \quad (3)$$

The second half of channels will be implemented similarly:

$$z_2 = \overline{\bar{x}_2(gx_1 \vee hx_1) \vee x_2(ex_1 \vee fx_1)} \quad (4)$$

That is, to restore one eight-channel multiplexer of the three "half" of the four channels is necessary, so that the third multiplexer plug on the leading variable either one or the other half, that is operated in a two-channel multiplexer.

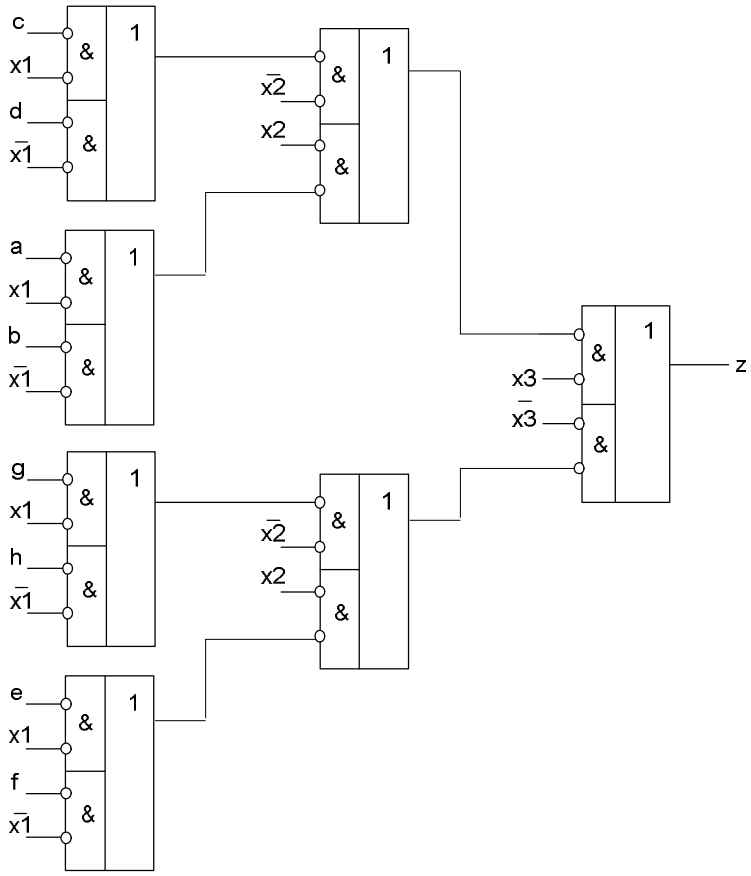


Fig. 1. The 8-1 multiplexer (eight channels), consisting of seven elementary multiplexers 2-1

Therefore, to set up the third component is necessary to:

$$z_3 = \bar{x}_3(\overline{\bar{z}_1 \bar{0} \vee \bar{1}0}) \vee x_3(\overline{\bar{z}_2 \bar{0} \vee \bar{1}0}), \qquad (5)$$

which corresponds to the two-channel multiplexer functions

$$z_3 = z_1 \bar{x}_3 \vee z_2 x_3. \qquad (6)$$

If there is a failure (one-time constant) to the address inputs – everything is much more complicated.

Table 1 shows how to rewire channels to counter such denial. Accordingly, the process of failure detecting is getting slow.

Table 1

Required reconnect channels with constant denials eight-channel multiplexer address inputs

x_2	x_1	x_0	**No fault**	**Failure**					
				x_2^0	x_2^1	x_1^0	x_1^1	x_0^0	x_0^1
0	0	0	0	0	4	0	2	0	1
0	0	1	1	1	5	1	3	0	1
0	1	0	2	2	6	0	2	2	3
0	1	1	3	3	7	1	3	2	3
1	0	0	4	0	4	4	6	4	5
1	0	1	5	1	5	5	7	4	5
1	1	0	6	2	6	4	6	6	6
1	1	1	7	3	7	5	7	6	6

"Half" of the logical elements can be used alone, but to restore a full multiplexer requires three "half" of the multiplexer.

2. Features of FPGA logic elements

Currently, FPGA contain configurable logic blocks (CLB) [1, 7], consisting of the logic elements, programmable local and global matrix connections MC – Fig. 2.

Logic gate FPGA - is a super redundant basis, and it is constructed as a read-only memory ROM (LUT - Look Up Table), which is a variable for the four multiplexer 16-1 (tree multiplexers), input data is set up so-called configurable memory cells [1] – Fig. 3.

In Fig. 3 inputs – S_0, S_1, S_2, S_3, the element is set to implement the sum modulo two $S_0 \oplus S_1 \oplus S_2 \oplus S_3$. On a specific set of variables is realized the only way from input to output, for example, from input 14: S3S2S1 (not S_0) [1].

Elementary multiplexers 2-1 is implemented as a switch (this is also a multiplexer) for example, on the basis of two chains of two transmit MOS transistors [1] – Fig. 4. Memory configuration (configuration data logic elements and matrices compounds) - this is the configurational cells, each of them contains six transistors [1, 8].

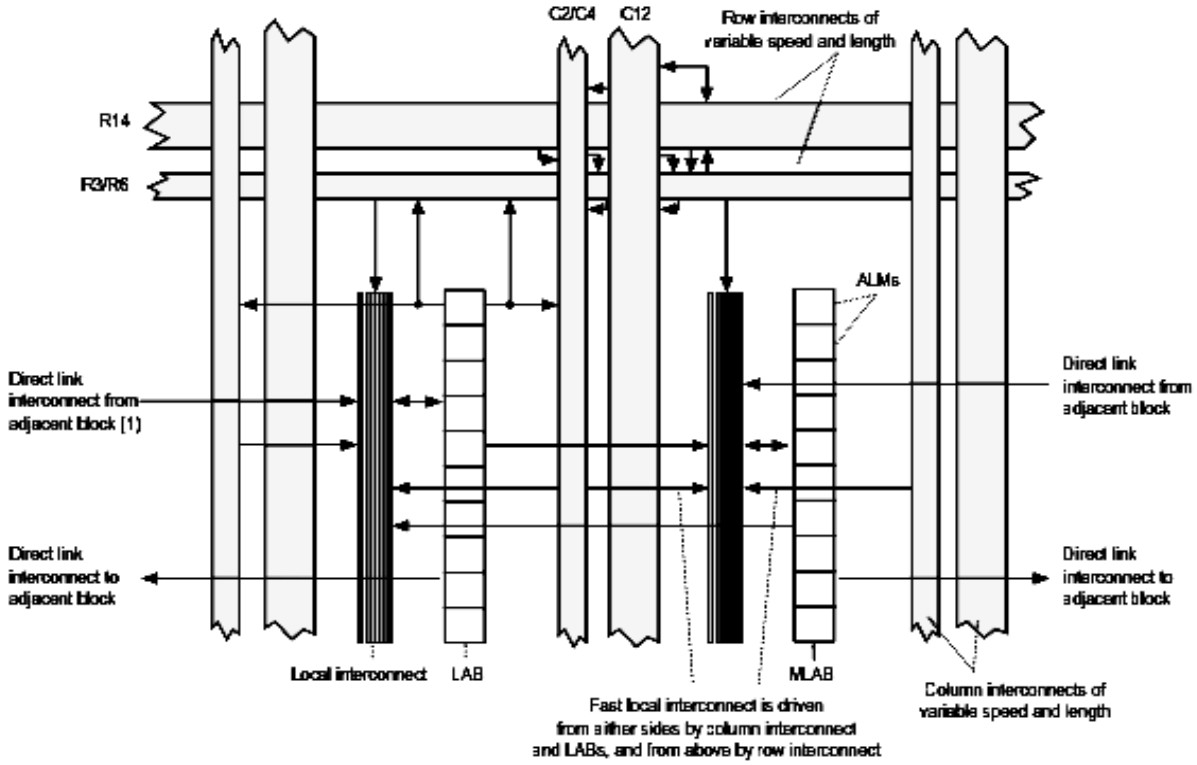


Fig. 2. Configurable logic block of Altera's FPGA

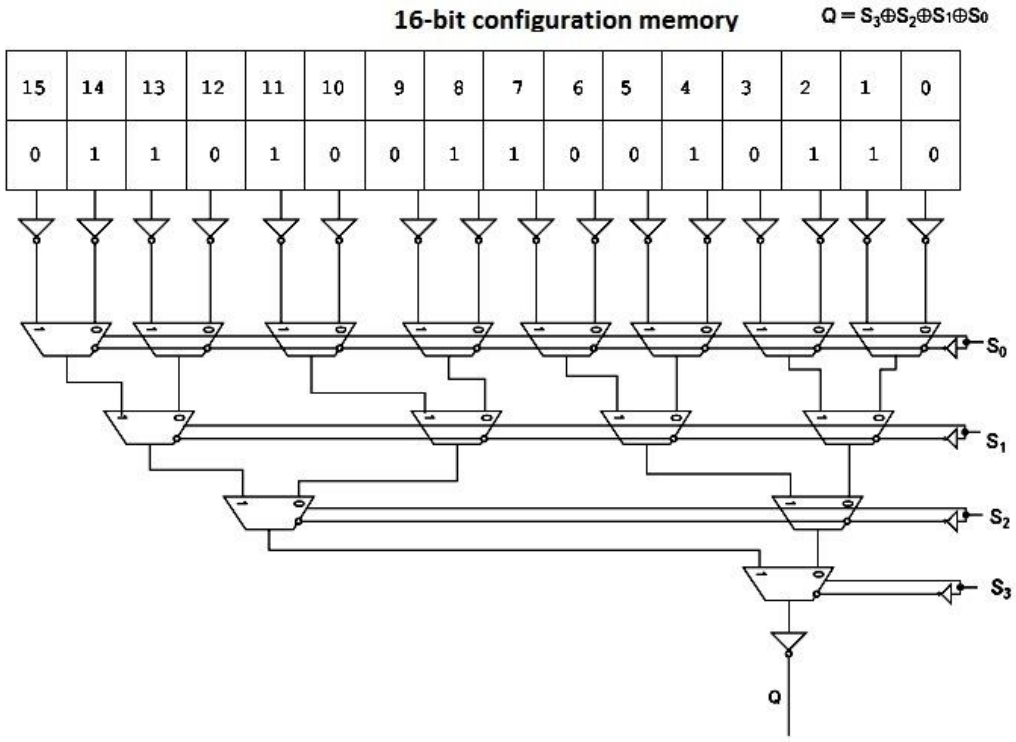
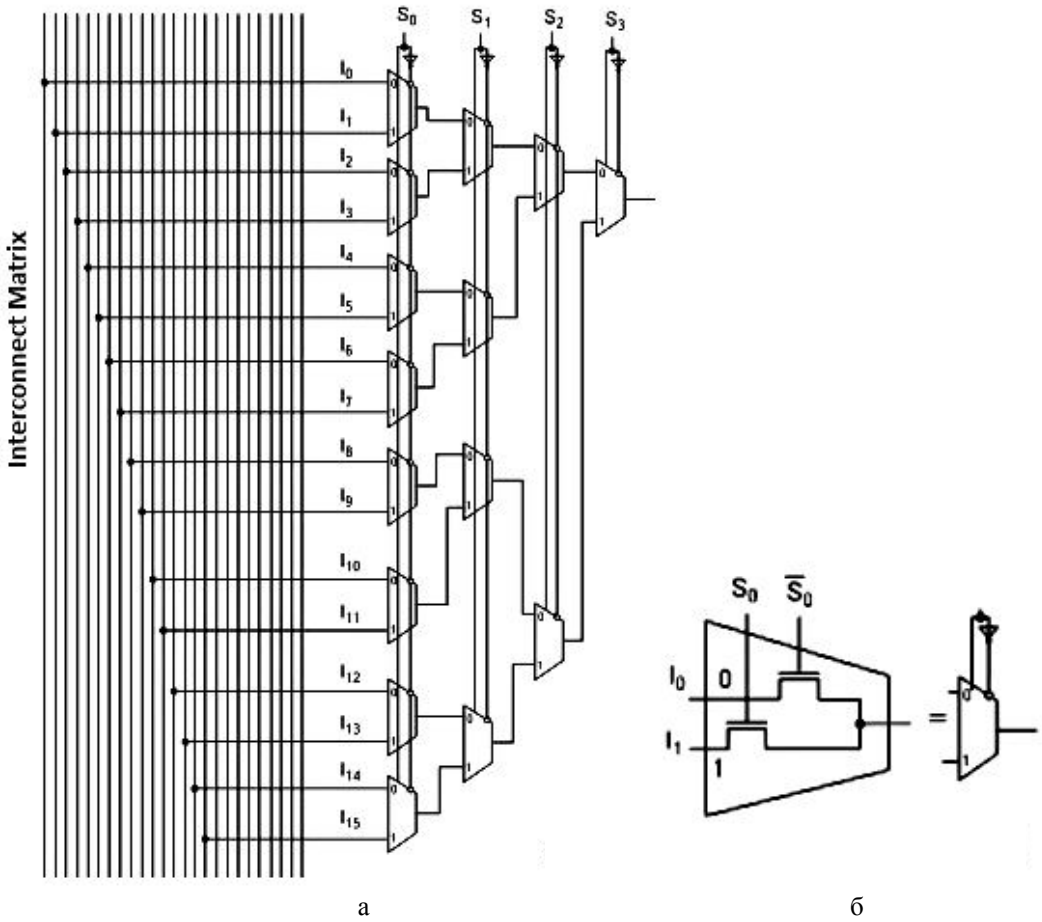


Fig. 3. Conversion table (Look-up Table) of FPGA



а б

Fig. 4. The switch signals from the local interconnect to the input CLB:
a – multiplexers' tree, b – implementation of the multiplexer transmitting MOSFETs

3. Features of adaptation to failures of transistors and inputs of LUT FPGA

Given the great redundancy logic elements on the basis of the conversion tables LUT, it is possible to restore the faulty conversion table. It is obvious that in this case there is a loss of functionality, but even LUT with limited functionality can be used for the synthesis of a large number of Boolean functions.

Let us consider a simple model of single constant failures. And we shall consider themselves as failures based transistors, which are built LUT, and the failure of address inputs. Input failure provides that the address input LUT has fixed logic level "0" and "1." A constant refusal to "1" in the CMOS transistor circuit includes sample source-drain or latching gate. A constant refusal to "0" CMOS transistor – is an open circuit source-drain or open the shutter.

Consider the possible cases of failure of the transistor.Suppose there was a single constant denial of transistor VT29 (Fig. 5).

If you set a single constant refusal to "0", in this case, the upper part of the network goes down, because the information from the SRAM cells that are connected to transistors VT1-VT8, can not be transferred to the output. But setting D=0, we can always connect the bottom of the exit and realize the function of three variables A, B and C. If you set up once the constant refusal of "1", the top part of the circuit is always connected to the output. In this case, setting D=1, turn off the lower part of the scheme and prevent the occurrence of faults.

At the top of the chart can also be synthesized function of three variables A, B, C.

Consider the failures in transistors connected to the line C. Suppose there was a single constant refusal to "0" in the transistor VT27. This means that the information from the SRAM cells that are connected to the transistors VT9-VT11, will never be passed on to the input. However, you can set C=0 and for all open transistors VT28 and VT26. Then we can implement the function of three variables A, B and D. In the event of a "1" on the transistor VT27, can not turn this thread from the transistor VT30. But setting C=1, we will close the transistors VT26 and VT28. Here again we can implement arbitrary functions of three variables A, B and D.

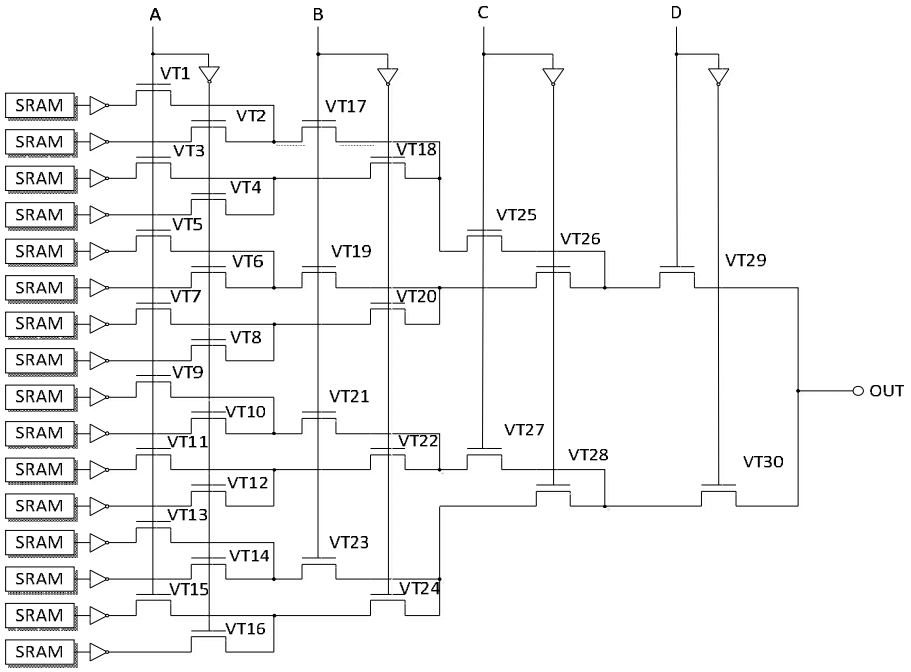


Fig. 5. Modeling failure of transistors in LUT

Consider the failures in transistors, which are connected to the line B. Let there was a failure to "0" in the transistor VT19. This means that the transistors and VT5 VT6, can never be switched to the input of the transistor VT26. Filed B=0 we open transistor VT18, VT20, VT22 and VT24. In this case the cells 1,2,5,6,9,10,13,14 will not be used when using LUT. However, in the remaining 8 SRAM can build the function of 3 variables A, C, and D. In the event of a "1" in the transistor VT19, is served B=1 and also implement a function of three variables A, C, and D from the remaining cells and SRAM transistors.

Suppose there was a failure in the transistor of the first stage, for example, not "0" in the transistor VT7. In this case, we can work with even the SRAM cell, it needs to set A=0. In this case, we can construct a function of the variables B, C and D. In case of refusal to "1" in the transistor VT7, similarly set A=1 and work with odd SRAM cells. Here, again, the remaining elements can be synthesized function of three variables B, C and D.

It is obvious that a similar situation will occur during the failure of the other transistors. Thus, a single failure in any transistor reduces the functionality of the item, however, due to the large redundancy is still possible to build a function of three input variables.

At single constant failures of the inputs LUT actually enforce the same situation arises that in case of failure of transistors. That is forcibly turned off one half of the LUT and a conversion table is converted from the multiplexer 16 to 1 in 8 to 1 multiplexer. But in this case it is also possible to synthesize a function of three variables.

Conclusion

Thus, this article presents an approach for FPGA logic partial recovery for critical applications by adapting to failures of logic elements based on the Look Up Table. The principle and the example of recovery of the LUT in single constant failures of transistors and inputs - go to the "half" or, more precisely, a "partial" functional. Use classical LUT for 4 variables, however, the same procedure can be applied to a larger number of LUT inputs. This gives rise to new opportunities parry multiple failures in a more complex tree transistors.

In the case of failure of hardware (logic elements) after massive failures, for example, in catastrophic situations, it is also possible software-hardware utilization failed elements.

In the future, should spread this approach, which could be called "partial firmware functionality" in other areas – for the implementation of energy-efficient FPGA.

In addition, it is useful to explore the possibility of using partial functionality for diagnosing FPGA.

References

1. Цыбин, С. Программируемая коммутация ПЛИС: взгляд изнутри [Электронный ресурс] / С. Цыбин. – Режим доступа: http://www.kit-e.ru/articles/plis/2010_11_56.php – 12.11.2012 г.

2. Yervant, Z. Gest editors' introduction: Design for Yield and reliability [Text] / Z. Yervant, G. Dmytris // IEEE Design & Test of Computers. – May-June 2004. – P. 177 – 182.

3. Post, E.L. The two-valued iterative systems of mathematical logic [Text] / E.L. Post /[Text] / Annals of Mathematics studies, no. 5. – Princeton University Press, Princeton, 1941. – P 122 - 134.

4. Tyurin, S.F. Retention of functional completeness of boolean functions under "failures" of the arguments [Текст] / S.F. Tyurin // Automation and Remote Control. – 1999. – Т. 60. № 9, part 2. – С. 1360 – 1367.

5. Tyurin, S. Redundant Basises for Critical Systems and Infrastructures [Text] / S. Tyurin, V. Kharchenko // General Approach and Variants of Implementation Proceedings of the 1st Intrenational Workshop on Critical Infrastructures Safety and Security, Kirovograd, Ukraine 11-13, May, 2011 / Kharchenko V., Tagarev V. (edits). – Vol. 2. – P. 300 – 307.

6. Программно-аппаратная реализация логических алгоритмов в микропроцессорных системах [Текст] / С.Ф. Тюрин, В.С. Харченко, Г.Н. Тимонькин, В.А. Мельников // Зарубежная радиоэлектроника. – 1992. – № 2. – С. 24 – 36.

7. Look up table implementation of fast carry for adders and counters [Text]: US 005274581A, 28.12.1993.

8. 6T SRAM Cell [Electronic resource]. – Access to: http://www.iue.tuwien.ac.at/phd/entner/node34.html – 12.12.2012 г.

Поступила в редакцию 15.02.2013, рассмотрена на редколлегии 13.03.2013

Рецензент: д-р техн. наук, проф. каф. компьютерных систем и сетей В.В. Скляр, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина.

АДАПТАЦИЯ FPGA К ОТКАЗАМ ЛОГИКИ

С.Ф. Тюрин, А.В. Греков, О.А. Громов, И.С. Понуровский

В статье предлагается восстановление логики FPGA (field-programmable gate array) для критических применений путём адаптации к отказам логических элементов. Принцип адаптации FPGA заключается в переходе на остаточные функциональные возможности LUT (Look Up Table), с возможностью программно-аппаратного использования их в случае недостаточности аппаратных средств после массовых отказов, т.е. адаптация проводится в случае невозможности использования технических средств после отказов логических элементов, например, при частичном отказе аппаратных средств. Также предложенный метод адаптации можно использовать при частичной функциональности системы для диагностирования FPGA.

Ключевые слова: адаптация, отказы, программируемая пользователем вентильная матрица, таблица поиска, логический элемент, отказоустойчивость, программно-аппаратная реализация.

АДАПТАЦІЯ FPGA ДО ВІДМОВИ ЛОГІКИ

С.Ф. Тюрін, А.В. Греков, О.О. Громов, І.С. Понуровський

У статті пропонується відновлення логіки FPGA (field-programmable gate array) для критичних застосувань шляхом адаптації до відмов логічних елементів. Принцип адаптації FPGA полягає в переході на залишкові функціональні можливості LUT (Look Up Table), з можливим використанням програмно-апаратного їх використання у разі недостатності апаратних засобів після масових відмов, тобто адаптація проводиться у разі неможливості використання технічних засобів після відмов логічних елементів, наприклад, при частковій відмові апаратних засобів. Також запропонований метод адаптації можна використовувати при частковій функціональності системи для діагностування FPGA.

Ключові слова: адаптація, відмови, програмована користувачем вентильна матриця, таблиця пошуку, логічний елемент, відмовостійкість, програмно-апаратна реалізація.

Тюрин Сергей Феофентович – Заслуженный изобретатель РФ, доктор техн. наук, профессор, профессор кафедры «Автоматика и телемеханика» Пермского национального исследовательского политехнического университета, Пермь, Россия, e-mail: tyurinsergfeo@rambler.ru.

Греков Артем Владимирович – канд. техн. наук, докторант Пермского национального исследовательского политехнического университета, Пермь, Россия, e-mail: grekartemvl@mail.ru.

Громов Олег Александрович – аспирант кафедры «Автоматика и телемеханика» Пермского национального исследовательского политехнического университета, Пермь, Россия, e-mail: ogromov@inbox.ru,

Понуровский Иван Сергеевич – аспирант кафедры «Математическое обеспечение вычислительных систем» Пермского государственного национального исследовательского университета, Пермь, Россия, e-mail: iponurovskiy@gmail.com.