computing@computingonline.net
www.computingonline.net

Print ISSN 1727-6209
On-line ISSN 2312-5381
International Journal of Computing

DETECTION METHOD OF THE PROBABLE INTEGRITY VIOLATION AREAS IN FPGA-BASED SAFETY-CRITICAL SYSTEMS

Kostiantyn Zashcholkin [1], Oleksandr Drozd [1], Yulian Sulima [2], Olena Ivanova [1], Ihor Perebeinos [1]

[1] Odessa National Polytechnic University, 1, Shevchenko Ave., 65044, Odessa, Ukraine
const-z@te.net.ua, drozd@ukr.net, en.ivanova.ua@gmail.com, perebeinos92@gmail.com
[2] Odessa Technical College of the Odessa National Academy of Food Technologies, 54, Balkovskaya str., 65006, Odessa, Ukraine, mr_lemur@ukr.net

Paper history:
Received 10 January 2019
Received in revised form 17 December 2019
Accepted 10 February 2020
Available online 14 June 2020

Keywords:
LUT-oriented architecture;
FPGA;
hardware Trojans;
safety-critical systems;
integrity monitoring.

Abstract: The features of integrity monitoring of FPGA-based safety-critical systems are considered. Hardware Trojans are distinguished as one of the most dangerous types of malicious integrity violation of FPGA-based systems. The study has proved that Hardware Trojans can be implanted into the system (or system project) during its planned modification. In particular, it happens when the integrity monitoring, based on the hash sum usage, does not operate. Before running the integrity monitoring, one should ensure that Hardware Trojans were not implanted. Authors proposed the method for detecting the hardware Trojans location in the space of FPGA-based components of safety-critical systems. The method is based on the analysis of addressing to the values of calculated LUT units for these components in the normal and emergency modes of system operation. The hardware module for addressing the registration in accordance with the proposed method is implemented.

1. INTRODUCTION

The computer hardware is increasingly being developed owing to the computer-aided design (CAD). The modern CAD systems, which provide the computer hardware design, include high-performance subsystems of simulation. At present, however the designed computer hardware complexity has achieved the level, when the used simulation subsystems are losing their efficiency. The traditional method of simulation using the software models for computer hardware components has following disadvantages: a) the excessive complexity of created models; b) large computational complexity of simulation; c) big durability of simulation process.

Taking into consideration all these arguments, we have concluded that increasing the paradigm of traditional software models by developing the hardware models on the basis of high operational efficiency of the modern CAD systems is promising today. For FPGA a hardware model is a subcircuit placed on a chip with target project. Such kind of a subcircuit can: a) obtain the information from necessary points of the main circuit; b) perform the processing of this information according to the simulation purposes; c) support the process of transferring the simulation results in a CAD system. Such approach performance is achieved because both the model and simulation object are functioning in the similar FPGA chip environment, which is native for both of them.

The hardware model is a project different from a target one but identical with it in implemented functions, and this permits to analyze a process of their (implemented functions) calculation. For a number of applications, the hardware models have some advantages in both designing and using as compared to software models.

In the given paper, a method of creating a hardware model is proposed. It provides the functional analysis of FPGA-project for detecting the hardware Trojans location in FPGA-project.

The chips FPGA are greatly used as a base for building computer systems that control high-risk

technical objects. The computer systems of such kind are customary called safety-critical ones [1], [2]. The selection of FPGA for safety-critical systems building is argued by the following factors [3-5]: a) the possibility to modify the system functions by reprogramming; b) the higher productive indicators. The first of the factors enable performing the functional system optimization without long-term switching off [6]. This simplifies the processes: updating the system functions; repairing the faults detected during the process of exploitation; optimization of individual system functions (or modules).

One of the important primary attributes of the safety-critical systems dependability is an integrity – an ability of the system to exclude the unintended modification of system and services provided by it [7, 8]. Since the chips FPGA are program-driven devices, their operation modification is made by changing their program code. That is why the program code (at the stage of system operation) or project descriptions translated in the program code (at the stage of system design) are the basic system integrity medium. Thus, program code integrity of FPGA-based components ensures reliability of systems consisting of different components.

2. RELATED WORKS

For safety-critical systems a hidden malicious implantation of hardware Trojans into them (systems) is the most dangerous type of integrity violation [9, 10]. For FPGA-based systems the hardware Trojans are the fragments of malicious program code secretly implanted into the system. These fragments create a subcircuit, which runs the Trojan and provides its malicious functioning in FPGA space. The hardware Trojans functioning is characterized by the activation engine (trigger) and malicious function (payload). The program code of hardware Trojan consists of two parts [11]: a part that programs the activation subcircuit and the one that sets a malicious Trojan function.

The activation subcircuit can potentially analyze the input system signals and the ones, which are present in the internal system points. The basic purpose of the subcircuit activation is to alert about running of the malicious Trojan function when a certain event occurs. The activation event can possess both a combinational nature (some set combination of input and internal signals of the system) and a sequential one (the fact of passing the sequence of set states). The complexity of the hardware Trojan detection in the system is caused by the low probability of its (Trojan) activation in testing the system [12-14]. A subject implanting the hardware Trojan into the system is interested in the absence of any Trojan manifestation up to the moment of its activation.

The subcircuit of malicious Trojan function provision can lead to a fault in operating the system and organize the leakage of confidential information that is processed by the system [15].

The hardware Trojan implantation into the FPGA-based system can occur both at the stage of the system exploitation and design. Yet at the stage of system design the Trojan is implanted into a project-fragment of high-level description (HDL and/or circuit-base description), which is translated into the FPGA program code in the end [16].

The integrity of the project of the FPGA-base system is usually provided by obtaining the hash sum [17, 18] for individual project files or the entire project [19]. Herein the hash sums (helping to carry out the integrity monitoring) are attached to the corresponding project files and as well as embedded into the project structure [20]. At the stage of system exploitation, the program code integrity of FPGA-based system can be provided either with the individual hash-sum file or by embedding hash immediately in the program code in the form of digital watermark [21, 22]. The hardware Trojan implantation into the FPGA-based system project violates this project integrity. The Trojan implantation into the functioning system program code violates similarly its integrity. The Trojan implantation into the project (system) which is under monitoring violates the integrity and consequently leads to the detection of the facts of implantation.

The possible way of evasion of the integrity monitoring (realized with the help of hash sum or checksum for sof or pof configuration files of FPGA) is the usage of the life cycle vulnerability of FPGA-based components of safety-critical systems. To analyze the life cycle of systems of such kinds a great number of models have been developed, for example, V-model of the life cycle of safety-related systems (Standard IEC 61508) [23] and the adaptation of this model to FPGA-systems [24]. The model, which determines the links between FPGA design stages and FPGA-project verification stages (and the products of development participate in the forming of these links) is also known [25]. In [26] a model of the integrity life cycle of FPGA-based systems is offered. The analysis of these models shows that there are two types of stages in the FPGA-based system life cycle:

a) stable stages during which the system (project) modification is not performed and the system (project) itself is under the integrity monitoring;

b) system (project) modification stages during which the legal system (project) modification is carried out wherefore the integrity monitoring is temporarily suspended.

In Fig. 1 the indicative stages of the life cycle are shown: stable stages are alternated with modification stages. The transition into modification stage requires the pause of the integrity monitoring process (time moments M_{stop}). The next transition into the stable stage requires hash-sum recalculation and restarting the integrity monitoring (time moments M_{start}). At the time moments when the integrity monitoring is not realized the illegal modifications of the system (e.g., hardware Trojans implantation) are possible along with the legal ones. In order to prevent the illegal modification before monitoring the system (project), validation should be performed [27-29] (time moments V), which guarantees that unintended modification has not been included.

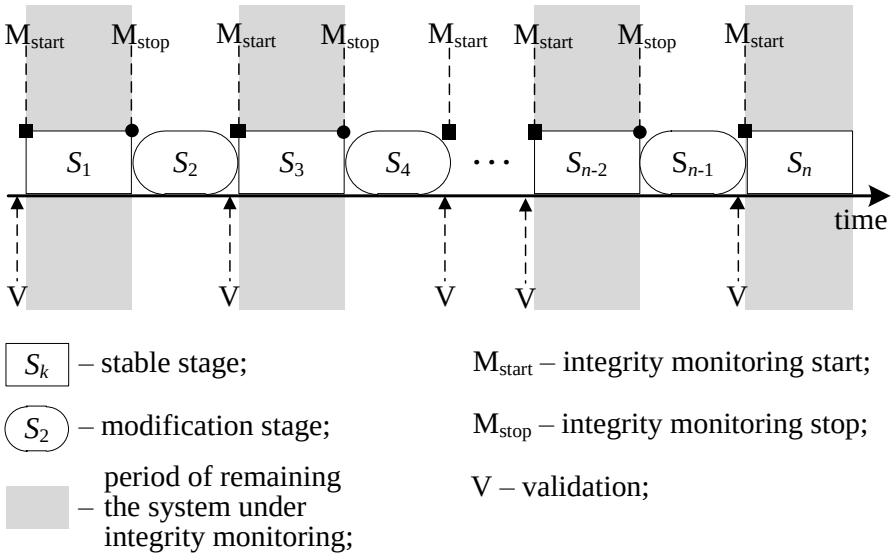

Figure 1 – Alteration of the life cycle stages of FPGA-based system

The process of the hardware Trojan detection (during validation) is complicated due to the following features. First, hardware Trojan is disguised as hardware resources providing the basic function of the system. Secondly, Trojan is created to complicate its detection in the process of testing the system. Lastly, Trojan does not manifest itself in the process of the system exploitation up to the moment of activation.

The goal of the given paper is to improve the process of the probable area detection of hardware Trojan location (performed in validating the system before starting the integrity monitoring) in the space of FPGA-based components of safety-critical systems using the analysis of addressing to the LUT unit values of these components.

3. METHOD OF ANALYSIS OF ADDRESSING TO THE LUT UNIT VALUES

The offered method we consider to be the one, which performs the preliminary processing of information of FPGA-based system functioning. The method is based on embedding a circuit, which registers addressing to LUT unit (elementary calculating unit of FPGA) [30, 31] values, in the space of chip FPGA. This circuit allows to register and extract the information of addressing to addresses of calculating LUT units from FPGA during the system functioning.

We suppose that the information of addressing to the LUT unit addresses can be used to detect the probable hardware Trojans location areas in chip FPGA space. We proceed from the following considerations. The safety-critical systems are designed for functioning in the two modes: normal and emergency. However there is a contradiction: on the one hand the main function of these systems is to provide the safety in emergency mode, on the other hand these systems function in the normal mode the most part of their life cycle [32, 33]. Wherein the sets of input codewords for safety-critical systems are essentially different both in normal and emergency modes. This can be explained with the peculiarities of informational interaction between high-risk objects and safety-critical systems serving them. Hardware Trojans can potentially manifest their malicious functions both in the normal mode and emergency one. However, we consider the scenario of attack on the system when Trojan manifests itself only in the emergency mode of system operation to be probable and the most dangerous. Within the framework of this scenario the Trojan preserves the visibility of system integrity and does not disturb its correct functioning in the

normal mode (most of the time of safety-critical system functioning). However, in extremely dangerous conditions Trojan behaves in the way which obstructs the system to function correct. This Trojan functioning scenario we suppose to be the most advantageous for an initiator of attack on the safety-critical system.

The matter is that the input data nature is sufficiently different for the normal and emergency modes [34, 35]. Under these conditions the presence of calculating LUT unit activeness statistics gives the possibility to analyze changes in dynamics of participation of these units in computational process (in each of the modes of FPGA-based safety-critical system operation) [26]. The statistics of addressing to the individual LUT unit addresses gives more detailed information of the unit functioning in the two modes in different sets of codewords. The information about which LUT units are active in each of the modes and which addresses of these units are used in each of the modes, is sufficient for the hardware Trojans detection methods. *The basic idea of the offered method* is the research of addressing to the LUT unit addresses on the input data distinct for the normal mode. Herein the LUT unit addresses, for which such addressing is not made, form set of addresses referred to the system functioning in the emergency mode. We consider such receiving the sets of addresses providing the system functioning in different modes of its operation as useful for future methods under more exact hardware Trojans location determination.

The addressing registration circuit is embedded in the researched FPGA-project (Fig. 2) within the framework of the proposed method as well as it is used for addressing the analysis to the LUT unit addresses. The circuit consists of similar fragments connected to inputs of each of the analyzed LUT units of the project. Each of these fragments is assigned to the individual LUT unit and designed for addressing registration to this unit addresses. A fragment consists of a decoder and shift register, which is able to input the data in parallel format. The register transfers in the mode of parallel data input at value 1 of signal L and in the shift mode at value 0 of signal L. The register feature is as follows – each of the bits of this register obtaining value 1 does not change it any more.

The addressing registration to the LUT unit addresses occurs at value 1 of the input signal Load/Shift. In addressing to some address of LUT unit $Adr_i = a_3a_2a_1a_0$ this address is sent to the decoder input. The decoder produces value 1 at its input i, which corresponds to input address Adr_i. The register on rising edge CLK receives value 1 from the decoder to put in bit i. With the help of this action the registration of addressing to LUT unit with address i is performed. In keeping the addressing registration procedure, the bit i of the register does not change and remains in value 1. On finishing the register procedure, the input signal Load/Shift is set in value 0. This leads to transferring the register to the shift mode. On each rising edge of the clock CLK the register data shift to the most significant bits occurs. Herein the next most significant bit shifts out of output SO.

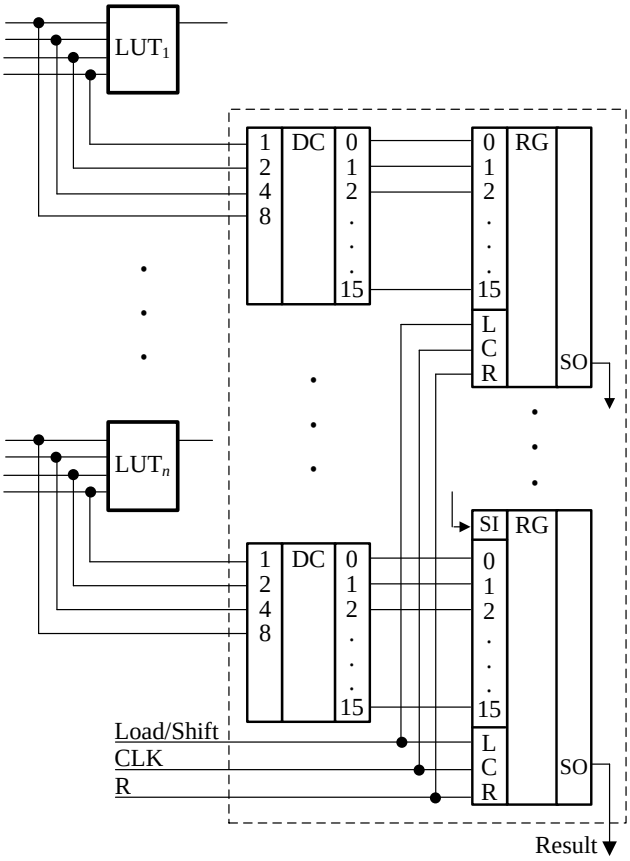

Figure 2 – Circuit of addressing registration to the LUT unit addresses

A circuit of the register, which provides the functioning necessary for addressing registration, is proposed (Fig. 3). Each of the register bits is a synchronous flip-flop. There is a two-address multiplexor and OR gate at each of the flip-flop input. At input signal value L=1 the flip-flops are connected in series and as a result the register transfers into the shift mode. At input signal value *L=1* the register transfers to the addressing registration mode (the flip-flops become unconnected to each other). In this mode the values D_i?Q_i (where D_i is the input signal of i-bit of the register; Q_i is the state of i-flip-flop; $i = 0…15$) are present at the flip-flops inputs. In such connection if a flip-flop transfers to state 1 (that is the addressing was fixed by this flip-flop) it cannot change this state by input data.

The circuit fragments registering the addressing to the LUT unit addresses have the common clock signal, reset signal as well as the signal of transferring to the registration and shift modes. The fragments are linked between each other by connecting the serial outputs SO to serial inputs SI in the shift registers. Such kind of connection enable to combine all fragments registers in the common shift register and extract the information about registered addressing through a single output Result.

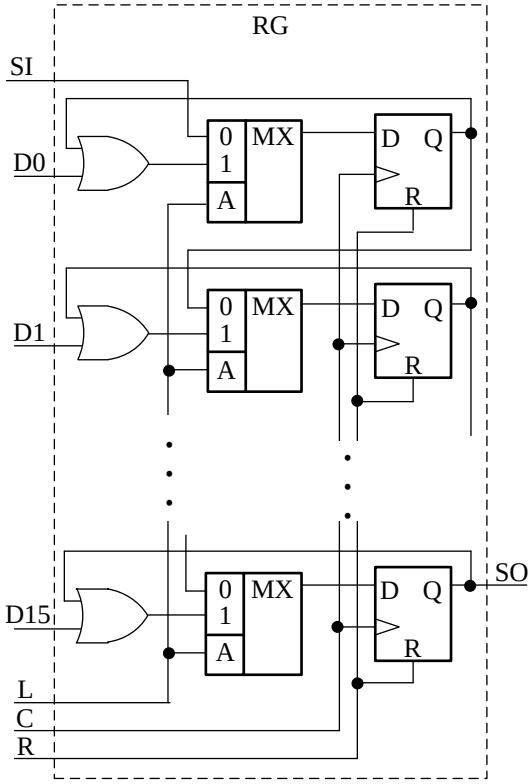

Figure 3 – Register subcircuit for registration circuits of addressing to LUT units

The proposed method is a sequence of stages leading to receiving the information about addressing to the LUT unit addresses during the researched system functioning.

Stage 1. The number *N* of LUT units used in FPGA-based system is indicated.

Stage 2. HDL-description of the registration circuit of addressing (Fig. 2) to the LUT unit addresses is formed. Herein the circuit contains *N* fragments of addressing registration.

Stage 3. By means of CAD the synthesis of circuit formed at stage 2 is carried out.

Stage 4. The amount of hardware resources of chip FPGA, which are not unincluded in the analyzed circuit, is indicated.

Stage 5. If the amount of free resources is not enough for placement of the circuit formed at stage 3 the number of circuit fragments decreases (in this case the method is not applied to the FPGA-system as a whole but gradually to individual subsets of its LUT units) and the return to stage 2 occurs. Otherwise the returning to stage 6 is executed.

Stage 6. Placement and routing of the circuit formed at stage 3 are carried out.

Stage 7. Connection of the placed circuit to the LUT unit inputs and the chip FPGA outputs is carried out (manually or with the help of special software).

Stage 8. Configurating the chip or preparing the project model for simulation is carried out.

Stage 9. The system is transferred in the mode of addressing registration to LUT units. The input data distinct for the normal system mode are sent to the system inputs (the real one or in the simulation mode). Herein the addressing registration circuit performs the accumulation of information about addressing to the specific LUT unit addresses in its registers.

Stage 10. On completing the addressing registration process the system is transferred in information extraction mode. In binary codeword extracted out of the system each of the bits is assigned to the specific address of the specific analyzed LUT unit. The presence of value 1 in a bit demonstrates addressing to the address corresponding to this bit.

4. CASE STUDIES

The proposed method was implemented in the form of software application. It (application) is an add-on for CAD Intel (Altera) Quartus [36]. This software application performs the following basic actions: a) extracts from FPGA-project the detailed information of placement of its elementary units, program codes of these units and the links between units; b) places the addressing registration circuit in FPGA-project; c) prepares the obtained FPGA-project model and transfers it to simulation system / performs the chip FPGA configurating; d) extracts the information about registered addressing out of simulation system.

By means of the developed software application an experiment has been carried out. This experiment was performed to obtain statistics of addressing to the individual LUT unit addresses. As an experimental material the five FPGA-projects were used. Target chips for these projects were FPGA Intel (Altera) Cyclone II–IV [37].

Each of these projects contains in its content a safety-critical system, in which a Trojan circuit was implanted. Each of the experimental safety-critical systems is a control system for a certain hypothetical high-risk technical object. Moreover, each of the systems has different complexity algorithms for forming control signals depending on the input

signals of the system. Due to the difference in these algorithms, the total number of LUT units of a project is different (from 196 to 843). Trojan circuits, which were implanted into experimental projects, used combinations (both parallel and sequential) of input values typical of emergency mode as an activating event. The action of the implanted Trojans was to block critical links in the circuits of experimental systems.

The data typical for the normal system mode is sent at the input of systems, which participate in the experiment. Wherein the activeness of addressing to the LUT unit addresses is registered. In the course of the experiment the following is detected (Table 1): N_{LUT} – the total number of LUT units in the project; N_{Adr1} – the number of LUT units at the inputs of which the only address takes place during the experiment; N_{AdrM} – the number of LUT units at the inputs if which more than one address took place during the experiment. Column *Result* contains information of correlation of the implanted Trojan circuit and LUT unit set N_{Adr1} (the fact demonstrating that LUT unit set N_{Adr1} covers the Trojan circuit completely or partially.

Table 1. Experiment results

Project No	N_{LUT}	N_{Adr1}	N_{AdrM}	Result
1	196	14	182	completely
2	221	15	206	partially
3	393	29	364	completely
4	655	32	623	completely
5	843	51	792	partially

In Table 1 one can see that in all the projects a hardware Trojan circuit completely or partially consists of LUT units including set N_{Adr1}. A number of LUT units in set N_{Adr1} (potentially probable area of Trojan location) is significantly less that the total number of LUT units. The information of such type (Table 1) adds the data of the LUT unit activeness obtained in accordance with a method which is offered in [26]. In total the information of the LUT unit activeness and statistics of addressing to their addresses gives the possibility to increase the Trojan search efficiency in starting the integrity monitoring on account of sufficient reduction of search area. With all the variety of the cases, the experiments confirmed the possibility of using the proposed method to reduce the search area in the Trojans detecting process.

5. CONCLUSION

An approach offered in the paper we take as the one of hardware model creation for analyzing the FPGA-project functioning for the purpose of probable area detection of hardware Trojans location in FPGA-project. The simulation problem, which is caused by the obtained hardware model, can be certainly solved with the help of traditional approaches due to software simulation. Software-based environment of target circuit functioning should be reproduced and the process of simulation be provided in this environment. This procedure is extremely difficult to be carried out and unformalized. But, the offered method is a strictly formalized sequence of operations which can be executed in automated mode.

We are positioning the proposed method as a base for hardware Trojans detection in the system which components are developed on FPGA. This method can be applied to the verification procedure before starting the system integrity monitoring. The method allows us to form two LUT unit address subsets: the subset providing the system functioning in the normal mode and the one providing its functioning in the emergency mode. Such formation of two subsets is based on the substantial difference of the input data nature intended for the two modes of safety-critical system functioning.

The offered method is not considered to be self-sufficient. It only performs the information preprocessing of FPGA-based system, namely registers addressing to the LUT unit addresses at input data distinct for the different system operation modes. The method presents the base information about location of possible integrity violation areas. This information is used in methods which can be run after the proposed method and perform more detailed procedure of localization. Therefore, we can conclude that there is the necessity to continue the research devoted to developing the methods, which use the results of the proposed method as initial data.

6. REFERENCES

[1] G. Xie, Y. Chen, R. Li, K. Li, "Hardware Cost Design Optimization for Functional Safety-Critical Parallel Applications on Heterogeneous Distributed Embedded Systems," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 6, pp. 2418-2431, 2018. DOI: 10.1109/TII.2017.2768075.

[2] C. Cho, W. Chung, S. Kuo, "Using Tree-Based Approaches to Analyze Dependability and Security on I&C Systems in Safety-Critical Systems," *IEEE Systems Journal*, vol. 12, no. 2, pp. 1118-1128, 2018. DOI: 10.1109/JSYST.2016.2635681

[3] C. Unsalan, B. Tar, *Digital System Design with FPGA*, McGraw-Hill, 2017, 402 p.

[4] F. Kastensmidt, P. Rech (Eds.), *FPGAs and Parallel Architectures for Aerospace Applications: Soft Errors and Fault-Tolerant Design*, Springer, Cham, Switzerland, 2016, 325 p.

[5] R. Woods, J. McAllister, G. Lightbody Y. Yi, *FPGA-based Implementation of Signal Processing Systems*, 2nd Edition, Wiley, Hoboken, USA, 2017, 356 p.

[6] J. Andina, *FPGAs: Fundamentals, Advanced Features, and Applications in Industrial Electronics*, CRC Press, Boca Raton, USA, 2017, 266 p.

[7] A. Avizienis, J. Laprie, B. Randell, C. Landwehr, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, issue 1, pp. 11-33, 2004.

[8] D. Maevsky, E. Maevskaya, L. Shapa, "Software reliability growth model's assumptions in context of the secondary faults," *CEUR Workshop Proceedings*, vol. 1844, pp. 645-6536 2017.

[9] N. Sklavos, R. Chaves, G. Natale, F. Regazzoni (Eds.), *Hardware Security and Trust: Design and Deployment of Integrated Circuits in a Threatened Environment*, Springer, Cham, Switzerland, 2017, 254 p.

[10] L. Bossuet, L. Torres (Eds.), *Foundations of Hardware IP Protection*, Springer, New-York, USA, 2018, 248 p.

[11] H. Salmani, M. Tehranipoor, "Analyzing circuit vulnerability to hardware Trojan insertion at the behavioral level," *Proceedings of the IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, New York, USA, 2013, pp. 190-195.

[12] D. Kleidermacher, M. Kleidermacher, *Embedded Systems Security: Practical Methods for Safe and Secure Software and Systems Development*, Newnes. Boston, USA, 2012, 416 p.

[13] M. Bishop, *Computer Security*, 2nd Edition, Addison-Wesley, Boston, USA, 2018, 1440 p.

[14] O. Kehret, A. Walz, A. Sikora, "Integration of Hardware Security Modules into a Deeply Embedded TLS Stack," *International Journal of Computing*, vol. 15, issue 1, pp. 22-30, 2016.

[15] M. Tehranipoor, H. Salmani, X. Zhang, *Integrated Circuit Authentication: Hardware Trojans and Counterfeit Detection*, Springer, Cham, 2013, 224 p.

[16] A. Adetomi, G. Enemali, T. Arslan, "Relocating Encrypted Partial Bitstreams by Advance Task Address Loading," *Proceedings of the IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, Napa, CA, 2017, pp. 188-191. DOI: 10.1109/FCCM.2017.50

[17] J. Katz, Y. Lindell, *Introduction to Modern Cryptography*, Second Edition, Chapman & Hall/CRC, 2014, 604 p.

[18] D. Grochol, L. Sekanina, "Fast reconfigurable hash functions for network flow hashing in FPGAs," *Proceedings of the NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, Edinburgh, 2018, pp. 257-263. DOI: 10.1109/AHS.2018.8541401.

[19] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th Edition, Pearson Education Limited, Harlow, United Kingdom, 2017, 768 p.

[20] J. Katz, *Digital Signatures. Advances in Information Security*, Springer, New York, USA, 2010, 192 p.

[21] K. Zashcholkin, O. Ivanova, "LUT-object integrity monitoring methods based on low impact embedding of digital watermark," in *Proceedings of the 14th International Conference* "*Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET-2018)*", Lviv-Slavske, Ukraine, 2018, pp. 519-523.

[22] F. Shih, *Digital Watermarking and Steganography: Fundamentals and Techniques, 2nd Edition*, CRC Press, Boca Raton, USA, 2017, 292 p.

[23] IEC 61508:2010 "Functional Safety of Electrical / Electronic / Programmable Electronic Safety-related Systems", 2010.

[24] J. Kim, E.-S. Kim, J. Yoo, Y. J.Lee, J.-G. Choi, "An integrated software testing framework for FPGA-based controllers in nuclear power plants," *Nuclear Engineering and Technology*, vol. 48, issue 2, pp. 470-481, 2016.

[25] E.-S. Kim, D.-A. Lee, S. Jung, J. Yoo, J.-G. Choi, J.-S. Lee, "NuDE 2.0: A formal method-based software development, verification and safety analysis environment for digital I&Cs in NPPs." *Journal of Computing Science and Engineering*, vol. 11, issue 1, pp. 9-23, 2017. DOI: 10.5626/JCSE.2017.11.1.9.

[26] K. Zashcholkin, O. Drozd, "The detection method of probable areas of hardware trojans location in FPGA-based components of safety-critical systems," *Proceedings of the IEEE 9th International Conference on Dependable Systems, Services and Technologies DESSERT-2018*, Kiev, Ukraine, 2018, pp. 220-225.

[27] R. Chakraborty, I. Saha, A. Palchaudhuri, G. Naik, "Hardware trojan insertion by direct modification of FPGA configuration bitstream," *IEEE Design & Test*, vol. 30, no. 2, pp. 45-54, 2013. DOI: 10.1109/MDT.2013.2247460.

[28] M. Komar, V. Golovko, A. Sachenko, S. Bezobrazov, "Development of neural network

immune detectors for computer attacks recognition and classification," *Proceedings of the 7th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications IDAACS'2019*, Berlin, Germany, 12-14 September 2013, pp. 665-668.

[29] T. Zhang and X. Wang, "High-reliable testing for FPGA software in space utilization engineering," *Proceedings of the International Conference on Dependable Systems and their Applications (DSA)*, Beijing, 2017, pp. 86-91. DOI: 10.1109/DSA.2017.22.

[30] H. Amano, *Principles and Structures of FPGAs*, Springer, 2018, 232 p.

[31] O. Drozd, M. Kuznietsov, O. Martynyuk, M. Drozd, "A method of the hidden faults elimination in FPGA projects for the critical applications," *Proceedings of the 9th IEEE International Conference on Dependable Systems, Services and Technologies (DESSERT'2018)*, Kyiv, Ukraine, 2018, pp. 231–234. DOI: 10.1109/DESSERT.2018.8409131.

[32] T. Xu, H. Wang, T. Yuan, M. Zhou, "BDD-Based synthesis of fail-safe supervisory controllers for safety-critical discrete event systems," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 9, pp. 2385-2394, 2016. DOI: 10.1109/TITS.2016.2515063

[33] V. Piterska, O. Kolesnikov, D. Lukianov, K. Kolesnikova, V. Gogunskii, T. Olekh, A. Shakhov, S. Rudenko, "Development of the Markovian model for the life cycle of a project's benefits," *Eastern-European Journal of Enterprise Technologies*, vol. 5, no. 4 (95), pp. 30-39, 2018. DOI:10.15587/1729-4061.2018.145252.

[34] A. Drozd, S. Antoshchuk, J. Drozd, K. Zashcholkin, M. Drozd, M. Kuznietsov, M. Al-Dhabi, V. Nikul, *Checkable FPGA Design: Energy Consumption, Throughput and Trustworthiness*, in: V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.), *Green IT Engineering: Social, Business and Industrial Applications, Studies in Systems, Decision and Control*, vol. 171, Springer International Publishing, Berlin, Heidelberg, 2019, pp. 73-94. DOI: 10.1007/978-3-030-00253-4_4.

[35] Y. Kondratenko, S. Encheva, E. Sidenko, "Synthesis of intelligent decision support systems for transport logistic," *Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS'2011*, Prague, Czech Republic, 2011, vol. 2, pp. 642-646. DOI: 10.1109/IDAACS.2011.6072847.

[36] Intel Quartus, [Online]. Available at: https://www.intel.com/content/www/us/en/software/programmable/quartus-prime/overview.html.

[37] Intel Cyclone FPGA series, [Online]. Available: https://www.intel.com/content/www/us/en/products/programmable/cyclone-series.html.

***Kostiantyn Zashcholkin**, PhD, Associate Professor of the Department of Computer Intelligent Systems and Networks of the Odessa National Polytechnic University. Area of scientific interests: FPGA-based systems, digital watermarking, digital steganography.*

***Oleksandr Drozd**, Dr. Sci Eng., Professor of the Department of Computer Intelligent Systems and Networks of the Odessa National Polytechnic University. Area of scientific interests: on-line testing and checkability of the digital components, safety-critical and FPGA-based systems.*

***Yulian Sulima,** Ph.D, Head of the Computer Systems Department of the Odessa Technical College in Odessa National Academy of Food Technologies**.** Area of scientific interests: FPGA-based systems, computing education.*

***Olena Ivanova**, Senior Lecturer of the Department of Computer Systems of the Odessa National Polytechnic University. Area of scientific interests: FPGA-based systems, digital watermarking, digital steganography.*

***Ihor Perebeinos**, Master, Postgraduate student of the Department of Computer Intelligent Systems and Networks of the Odessa National Polytechnic University. Area of scientific interests: on-line testing and checkability of the digital components, safety-critical systems, FPGA-based systems.*