Plenary Section

UDC 004.383

Anatoliy Sergiyenko, Yuriy Vinogradov,
Olexiy Molchanov, Chojadurdy Jepbarov

MALICIOUS HARDWARE IN FPGA

Сергієнко Анатолій, Виноградов Юрій,
Молчанов Олексій, Джепбаров Ходжадурди

ЗЛОНАМІРЕНО СТВОРЕНЕ АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ У ПЛІС

A survey of malicious hardware in FPGA is considered. The methods for malicious hardware searching and preventing its loading into FPGA are highlighted as well. A conclusion is made that FPGA is the most-safe device against malicious hardware loading. A stack processor structure is proposed which can be used for monitoring the malicious hardware.

Keywords: FPGA, stack processor, side channel.
Fig.: 0. Tabl.: 1. Bibl.: 7.

У статті оглядаються можливості впровадження злонамірено створеного апаратного забезпечення (ЗСАЗ) у програмованих логічних інтегральних схемах (ПЛІС), способів їх виявлення та перешкоджання їх впровадженню. Робиться висновок про те, що ПЛІС є найбільш захищені від впровадження в них ЗСАЗ. Запропоновано блок моніторінгу ЗСАЗ на базі стекового процесора.

Ключові слова: ПЛІС, стековий процесор, побічний канал.
Рис.: 0. Табл.: 1. Бібл.: 7.

Introduction. Malicious hardware is the hardware, which is designed and implemented to violate the security and reliability of the computer systems. Many issues of malicious hardware in ASICs are described in [1]. Recently, more attention is paid to malicious hardware which is implemented in the field programmable gate arrays (FPGAs) [2,3]. The development and implementation of projects for FPGA has the same stages as the creation of the software: compilation, and debugging of the programs written in a high-level programming language, loading of the codes into the equipment. Therefore, the well-known malicious software and malicious hardware have much in common. But the second one needs more investigations.

Types of malicious hardware. The classification of malicious hardware comes from the classification of malicious software. A *backdoor* or a *trapdoor* allows the access to a computer system that is not subject to its specifications. It can be implemented during the development of the computer system or during its upgrade.

A *kill switch* is a malicious artificial object that allows the attackers to interfere with the correct functioning of the hardware. The kill switch causes a failure in the form of an inability to perform the necessary functions. It can be installed as part of the FPGA hardware or the FPGA configuration.

An *FPGA virus* is part of the FPGA configuration, which can cause the short-circuiting of the outputs of the internal gates and, as a result, the overheating and failure of the chip [4].

Trojan hardware is a wide variety of malicious hardware. Like the software Trojans, this malicious hardware is implemented by third parties and is intended to make a violation of the normal device operation, such as slowing down, or transmitting or receiving important information through a hidden channel [5].

By the size and number of involved inputs-outputs, the malicious hardware blocks are divided into *small* and *large* ones. A large block is much easier to activate and detect than a small one. But it has a functionality that provides the solutions that cause great losses.

Malicious hardware can be activated from the outside *through a hidden channel* or *from inside*. The latest malicious hardware is divided into *always-on* and *on-demand* activated hardware. The always-on malicious hardware is permanently activated and may interfere the functioning of the computer system at any time. The on-demand malicious hardware is inactive until a certain condition is met. This may be the fixation of a given internal logical state or state of input signals or overflow of an internal counter. Such malicious hardware cannot be detected until its activation.

The *information transmitting* malicious hardware is designed to send secret information from a protected area to the outside. For example, a certain change in the network parameters of the encryption block allows the spy to determine the encryption key through the analysis of power consumption of the computer system.

Ability to implement malicious hardware in FPGA. Theoretically, malicious hardware can be embedded in FPGA during crystal manufacturing. But due to the fact that at the production stage there is no thought about where and how the FPGA would be used, what kind of firmware would be implemented, the implementation of malicious hardware is too difficult [6].

The FPGA firmware can always be designed in a state of secrecy. Therefore, a project for FPGA is protected, if it is not acquainted with an intruder. Since the operation of FPGA is based on the information in the configuration file, the standard methods of the information protection are applied to it.

The attacks on the hardware, which are used for the conventional chips, are unacceptable for FPGAs because they lead to the configuration destruction. In order to inject the malicious hardware into FPGA, the malefactor must have the initial or the precisely reconstructed project.

Nevertheless, the additional protection measures are being taken in responsible applications. The simplest measure is the one-time programming, when the full access

to the configuration is not possible. During the last two decades, FPGA encrypts the configuration file, which allows it to be loaded into FPGA multiple times after shutting down and turning on the power without any unauthorized copying. This additionally makes it impossible to reconstruct the project.

In addition, the partial FPGA reconfiguration, which allows the intruder to attach the parts of the project, is prohibited in the encrypted mode. Also, the requirement to turn off the power before downloading a new configuration destroys the previous configuration. The configuration file key is stored in battery-operated RAM of FPGA, which is automatically erased when it is turned off. The known attack methods on this key disable this power and break the cipher since the key is stored at the depth of the crystal under several layers of dielectrics and metallization.

Thus, perhaps the only opportunity to insert the malicious hardware into the FPGA is its attachment during the design process. For example, the designer can insert an IP core from a third-party vendor with the malicious hardware hidden in it. Such an IP core can be embedded in an automated software program generator that invisibly invades this malicious hardware to the project [6].

Ways to detect malicious hardware. There are three basic approaches to detect malicious hardware in FPGA. They are an analysis of project files, testing with the automatic test generation and analysis of signals from the side channels [5]. But without knowledge of the logical network, its location, and activation method of the malicious hardware, it is virtually impossible to find a test sequence that detects the presence of this malicious hardware.

The analysis of the information leakage in the side channels allows the investigator to detect the presence of malicious hardware. For example, the feedback signals such as power consumption, infrared radiation, and radio frequency can be analyzed. The efficiency of this method is increased if the special modules which are introduced in the chip that contribute to this analysis. These may be the voltage and temperature sensors, delay measurement modules placed at different points of the chip, as well as the built-in self-testing circuit [2].

In contrast to the complex analysis of ASIC, the detection of malicious hardware in FPGA is trivial. Any correction of the configuration file changes its control code. In addition, the existing software tools make it possible to compare the project versions at different design stages to distinguish the unwanted logic networks [6].

Prevention of malicious hardware introduction. The harmful effect of the malicious hardware can be made impossible by the special measures of the structural design. So, a redundant method can counteract the kill switches, and the introduction of the infrastructure modules like the parametric sensors help to detect the malicious hardware.

The most of FPGA projects are built around the general purpose processor which runs some user programs under some RTOS. This processor and its memory remain the target for the attacks from both malicious hardware and software. The most

effective way is the spatial and logical isolation of the responsible blocks like this processor from the rest of the computer system. This is the essence of the moats and drawbridges method. This isolation greatly complicates the configuration of the hidden channel and simplifies its detection.

Block of monitoring the malicious hardware. To remove a side channel through a module with shared access, for example, the shared memory, the mechanism for access confirmation is usually used. Then, the request monitoring block denies access to the shared module from other modules that do not have this right, including the malicious hardware [2].

A small 8-bit configured microcontroller, such as SM8, described in [7] is proposed to serve as a block of monitoring the malicious hardware. Its features are very small hardware volume, short instructions (8-bit and16-bit instructions), which are implemented for one or two clock cycles. Table 1 shows the parameters of the SM8 core configured in FPGAs of different series.

The microcontroller core has a very small share in the total hardware volume and moderate speed, which is enough for its functioning. It could not seriously infer the computer system project configured in FPGA. From this point of view, this core can serve as the malicious hardware as well. Therefore, its investigations can help to investigate the problem of malicious hardware revealing.

Table 1

Parameters of the SM8 microcontroller core

FPGA series	***Hardware volume***	***Share of total volume***	***Maximum clock frequency, MHz***
Xilinx XC7K480T	181 LC	0.038%	250
Intel 10M50	1164 LE	2.3%	150
Intel 10CX220	210 ALM	0.095%	230

Conclusions. In recent times, a lot of attention is being paid to the malicious hardware problem due to the fact that it poses a serious threat to the security of the state and people. The detecting or removing the malicious hardware embedded in FPGA is a very complicated task. However, the implementation of malicious hardware in the FPGA-based computing systems is considerably complicated compared to ASICs. The cheapest and most reliable way to prevent the implementation of malicious hardware in FPGA is to perform the appropriate security measures when designing a configuration file for the project. The structural and logical measures that prevent the introduction of malicious hardware are redundant design, spatial and logical isolation of critical units, access control to shared resources, minimization of noise switching, implementation of infrastructure modules to counteract malicious hardware. The development, detection, counteraction, and prevention of malicious hardware is a range of tasks that require further multilateral scientific and technical research.

References

1. Сергієнко А. М. Злонамірено створене апаратне забезпечення / А. М. Сергієнко // Information Technology and Security. – 2012. – № 1. – С. 93 – 100.

2. Huffmire T., Irvine C., Nguyen T.D., Levin T., Kastner R., Sherwood T. Handbook of FPGA Design Security. – Springer. – 2010. – 177 p.

3. Security Trends for FPGAs. From Secured to Secure Reconfigurable Systems / Editors: B. Badrignans, G. Gogniat, J. L. Danger, L. Torres, V. Fischer. – Springer, 2011. – 196 p.

4. Hadzic I., Udani S., Smith J. FPGA viruses // Proceedings of the 9-th Int. Workshop on Field-Programmable Logic and Applications, FPL'99, Glasgow, UK. – 1999. – P. 291 – 300.

5. Wang X., Tehranipoor M., Plusquellic J. Detecting malicious inclusions in secure hardware: challenges and solutions // Proc. IEEE Workshop on Hardware Oriented Security and Trust, HOST, Anaheim, CA, June, 2008, –2008. –P.15-19.

6. Trimberger S. Trusted design in FPGAs // Proceedings of the 44-th Design Automation Conference, San Diego, CA, USA, DAC 2007, June 4 – 8, 2007, – P. 1.5 – 1.8.

7. Molchanov O. Microprocessor Architecture for Serial Port Communications / Sergiyenko A., Orlova M. // Advances in Computer Science for Engineering and Education II. – Springer. – 2019. – P. 238 – 246.

Authors

Anatoliy Sergiyenko – professor, Department of Computer Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

E-mail: aser@comsys.kpi.ua

Сергієнко Анатолій Михайлович – професор, кафедра обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

Juriy Vinogradov – assistant, Department of Computer Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

E-mail: vinograd@comsys.ntu-kpi.kiev.ua

Виноградов Юрій Миколайович – асистент, кафедра обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

Oleksiy Molchanov – post-graduate student, Application Specific System Department, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

E-mail: oleksii.molchanov@gmail.com

Молчанов Олексій Андрійович – аспірант, кафедра спеціалізованих комп'ютерних систем, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

Chojadurdy Jepbarov – student, Department of Computer Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

Джепбаров Ходжадурди – студент, кафедра обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

РОЗШИРЕНІ АНОТАЦІЇ

Сергієнко Анатолій, Виноградов Юрій,
Молчанов Олексій, Джепбаров Ходжадурди

ЗЛОНАМІРЕНО СТВОРЕНЕ АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ У ПЛІС

Актуальність теми дослідження. На відміну від злонамірено створенного програмного забезпечення, злонамірено створеному апаратному забезпеченню (ЗСАЗ) присвячується недостатньо уваги. В той же час ЗСАЗ може нанести не менше шкоди, ніж його програмний аналог, а методи і заходи для протидії ЗСАЗ розвинуті недостатньо.

Аналіз останніх досліджень і публікацій. Протягом останніх років з'являється все більше статей присвячених класифікації та заходам виявлення та протидії ЗСАЗ особливо у замовлених НВІС. Але проблемам протидії ЗСАЗ в програмованих логічних інтегральних схемах (ПЛІС) присвячено мало наукових робіт.

Виділення недосліджених частин загальної проблеми. Дана стаття присвячена огляду, вивченню та аналізу проблем, які пов'язані з впровадженням ЗСАЗ у ПЛІС.

Постановка завдання. Завданням є пошук перспективного напрямку дослідження ЗСАЗ у ПЛІС та створення блоку моніторингу наявності ЗСАЗ та протидії його впливу на ресурси, які розділяються.

Викладення основного матеріалу. Розглянуто класифікацію ЗСАЗ, які впроваджуються у ПЛІС, шляхи, ймовірність та ефективність їх впровадження та заходи для їх виявлення і знешкодження. Запропоновано блок моніторингу наявності ЗСАЗ та протидії його впливу на ресурси, які розділяються. Блок виконано у вигляді ядра мікроконтролера зі стековою архітектурою, який має як мінімізовані апаратні витрати, так і невеликий об'єм вбудованого програмного забезпечення.

Висновки. Встановлено, що ПЛІС є достатньо захищеною мікросхемою від впровадження ЗСАЗ. Найбільш ймовірним джерелом ЗСАЗ у ПЛІС є віртуальні модулі, які поставляються сторонніми компаніями. Для протидії таким ЗСАЗ запропоновано блок моніторингу наявності ЗСАЗ та протидії його впливу на ресурси, які розділяються.

Ключові слова: ПЛІС, стековий процесор, побічний канал.