ISSN No:-2456-2165

Implementation of Cryptography Algorithms in Field Programmable Gate Array

Michal Hulič
Dept. of Computers and Informatics, FEEI TU of Košice,
Letná 9, Košice, Slovak Republic

Norbert Ádám
Dept. of Computers and Informatics, FEEI TU of Košice,
Letná 9, Košice, Slovak Republic

Abstract:- This paper is handling the problematics of implementation of selected cryptography algorithms in symmetric and asymmetric category, specifically RSA in symmetric and DES in asymmetric field. The implementation is focused on representation of computer unit, which is decelerated as hardware accelerator. Chapters are divided from the introduction to problematics to description of algorithms DES and RSA, that are the part of computer security field, the tested FPGA device is described in its own chapter, the design and completed solution and the results of systematic tests has presence at the end of this document.

Keywords:- *RSA; Computer Security; DES; Hardware Description Language; FPGA; VHDL.*

I. INTRODUCTION

This document in general is handling mostly with two parts, cryptography and hardware computing. The first part is defined according the literature, that cryptography has characteristic feature. The cryptography should be based on unidirectional mathematical function that secures the process called transformation of message or raw data to encrypted set of data. This transformation called encryption process should be fast and as secure as possible that guarantees that input data in process of encryption will be in best cases computationally impossible to decrypt using mathematical feature of unidirectional functions. The computational difficulty of decryption process should be as high as possible to hide the information of data. For our case we have chosen two encryption algorithms, specifically the asymmetric RSA and the symmetric DES encryption algorithm.

The principle of symmetric ciphers is pretty simple. The symmetric encryption uses only one key that shares both sender and receiver of message. This encryption is find as less secure than asymmetric encryption algorithms due to simplicity of computational process and the possibility of unwanted share of the key. The DES algorithm was declared as broken in 1993 [1], but we can also find the application of this algorithms these days [2][3], also in hardware implementations [4][5].

The principle of asymmetric ciphers is far more difficult. The RSA algorithm is among the most popular encryption algorithms nowadays [6]. The abbreviation is derivated from the surnames of its inventors Rivest, Shamir and Adleman, who created this algorithm in 1977 [6]. The RSA algorithm is widely spared in application for data encryption and for securing digital signatures [7]. The process of computation is presented in chapter II. The key which has less than 1024 bits is now considered as not safe anymore.

There are many applications of RSA implementation, e.g. RSA implementation in securing electronic passport and many others that are present in following literature [7-14].

This document contains our implementation of DES and RSA algorithm using specific FPGA device. The evaluation of the proposed modules are shown at the end of this publication.

II. THE RSA ALGORITHM

The RSA algorithm belongs to group of asymmetric cipher algorithms [7]. For RSA algorithm is process of encryption as follows. If person X wants to send a private message M to person Y, first person X encrypts the message by public key of person Y, and for the person Y is possible to decrypt the message using its own private key.

A. Creating public and private keys is presented as follows

- *Two big different prime numbers p, q are chosen from random generated numbers*
- Compute

$$n = p \times q \tag{1}$$

- Compute

$$\phi(n) = (p-1)(q-1) \tag{2}$$

- Next step represent selection of random integer

$$1 < e < \phi(n) \text{ where} GCD(e, \phi(n)) = 1 \tag{3}$$

- Compute

$$1 < d < \phi(n) \text{ where} ed \equiv 1 (\text{mod} \phi(n)) \tag{4}$$

- The (n, e) pair represents the public key and the (n, d) pair designates the private key.

It is crucially important, mainly for safety reasons, that integers p, q are randomly generated. Numbers should be random, different and sufficiently big prime numbers for the attempts of subsequent factoring [12]. It is also necessary to perform the test if the numbers are prime numbers according to the respective algorithm. In the literature [12], Solovay-Strassen or Fermatov, Miller-Rabin, Frobenius tests are usually in use to persuade if the numbers are prime as the primary numbers tests. The encryption/decryption process is presented below.

ISSN No:-2456-2165

According the equation (4), as follows the private key exponent is computed.

- Message M need to be selected and encrypted.
- Next step is to find public key, the person N, by using which we are going to encrypt the message M and the (*n*, *e*) key is representation of it.
- The message is representation as integer among 0 to n-1. Blocks of messages are used if the message is too large. Each block is also representation as integer as the same range [0, (n-1)].
- The final computation called encryption is represented as follows

$$C = M^e \pmod{n} \quad (5)$$

- Using public key (*n*,*e*) of person to who we are going to send the encrypted data are send using secure channel.
- Using the private key (n, d) we are going to perform a decryption process which belongs to the first person. The decryption process is computed as follows

$$M = C^d \pmod{n} \quad (6)$$

III. THE DES ALGORITHM

DES algorithm is abbreviation of Data Encryption Standard. This algorithm belongs to group of symmetric ciphers [19].

DES algorithm also belongs to group of archetypal block ciphers. It is actually an algorithm with fixed length string (Fig. 1) of plaintext bits and inputs in transformation through a series of complicated operations (Fig. 2) that result is cipher text bit string of the same length. DES algorithm has specifically length of block of data and also the key length is strictly limited to 64 bits. However, only 56 bits are used in encryption/decryption. The remaining 8 bits are used for checking parity that are thereafter discarded [14].

The encryption/decryption process is based on the Feistel function [15].

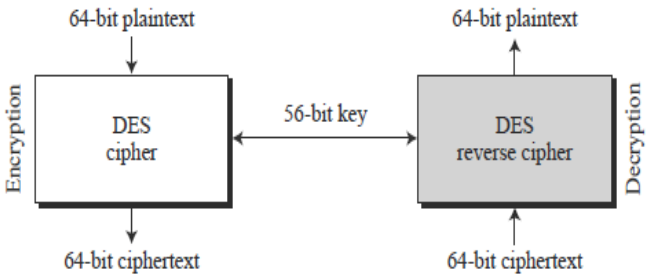

Fig 1:- Basic DES layout of encryption and decryption [13]

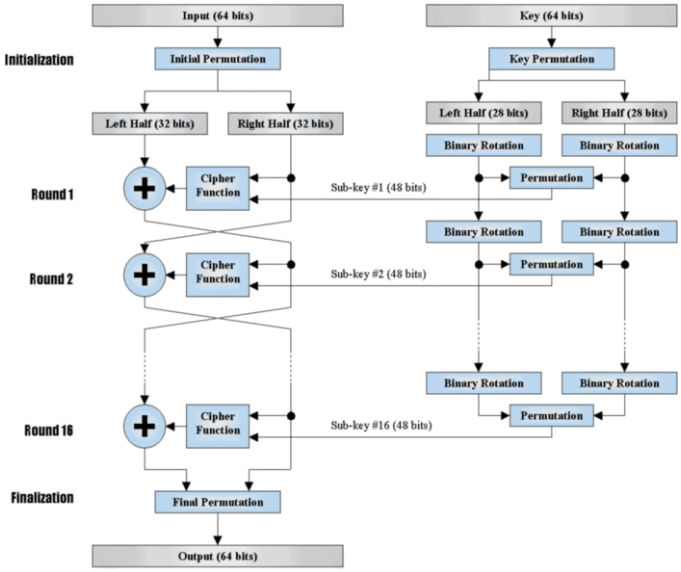

Fig 2:- Block of 16 rounds of DES algorithm [16]

IV. THE HARDWARE ENVIRONMENT FOR DEVELOPING AND EVALUATING DESIGNS

The two mentioned cryptography algorithm were implemented on Xilinx Kintex-7 KC705 development board (Fig. 3). The board provides a hardware environment for developing and evaluating the proposed DES and RSA modules. The key components of the board are XC7K325T-2FFG900C FPGA chip (Tab. 1), 1 GB DDR3 SODIMM 800 MHz/1600 Mbps memory, USB JTAG connector, SD connector, PCI Express interface, SFP connector, ethernet connector, HDMI connector, I2C bus or GTX receiver and transmitter. The full description of all components of this board is presented in documents [21] [22].

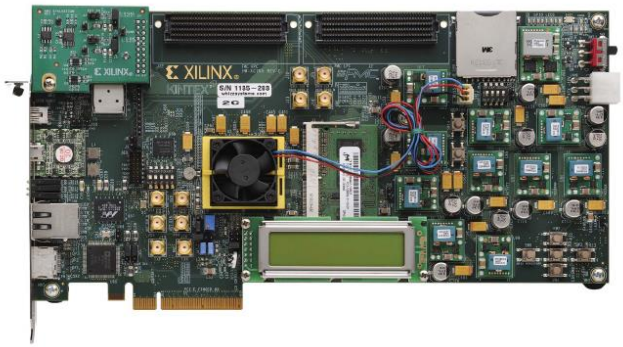

Fig 3:- The Kintex 7 KC705 board

	CLB Flip Flops	Logic Cells	DSP48 Slices
7K325T chip	407 600	326 080	840

	Slices	LUT	BRAM_36k
7K325T chip	50 950	203 800	445

Table 1. Ammout units present in tested board Part 1. and 2.

ISSN No:-2456-2165

The following chapters deal with our implementation of two mentioned cryptography algorithm.

V. IMPLEMENTATION OF DES ALGORITHM

The implementation of DES algorithm takes input of message M that has 64 bits, and the key K with the same size. The output from the process is the cipher C. The inverse process of encryption is decryption where the input is cipher C and the key K. The output is the original message M.

The Tab. 2 shows input and output ports of the circuit.

Name	Direction	Purpose
clk	IN	Clock signal for synchronization of components
reset	IN	Reset signal for reset of memory component
Key	IN	64 bit key
dIN	IN	Message input (n-bit wide input)
dOUT	OUT	Message output (n-bit wide output)
encryptFlag	IN	Flag for setup circuit for encryption
decryptFlag	IN	Flag for setup circuit for decryption

Table 2. FPGA tested device features

The two main steps in the process of encryption/decryption are:

- Create 16 sub keys, each of which is 48-bits long.
- Encode each 64-bit block of data.

These steps in our design of the DES module are based on the implementation of these sub-modules:

Key sub-module. This sub-module contains two logic circuits PC1 and PC2. These circuits are responsible for permutation of bits of the encryption/decryption key. This sub-module produces 16 sub keys.

Initializing sub-module. This sub-module is responsible for applying the initial permutation to the block of data.

Shift sub-module. This sub-module is responsible for performing 16 iterations. The first stage of this sub-module generates a data block of 48 bits from 32 bits of data. The next stage produces new data by using the eight S-box functions that are actually implemented as look-up-tables.

Reverse sub-module. The last sub-module is responsible for transfer the block of data to the original order by using XOR function.

The DES module was created in Vivado Design Suite - HLx of Editions 2016.3 development software by using C and VHDL languages.

VI. EVALUATION OF DES MODULE

For the purpose of DES module evaluation, we used DES modules with different configuration based on the message length. The table 3. shows dependency between the size of the message and the module computation time (response time). This dependency is also shown in Fig. 4. Observation showed that the computation time is increasing by using a bigger message. However, using a module for 1024-bit length message the encryption takes only approx. 0.22 seconds opposite to approx. 14 seconds achieved by chaining 128 modules configured for encryption of 8-bit length messages (128 x 0.106583 + delay).

Size of message	Computation time
8 bit	0.106583 s
64 bit	0.152119 s
128 bit	0.163629 s
256 bit	0.206162 s
512 bit	0.216516 s
1024 bit	0.224175 s

Table 3. Measured Times

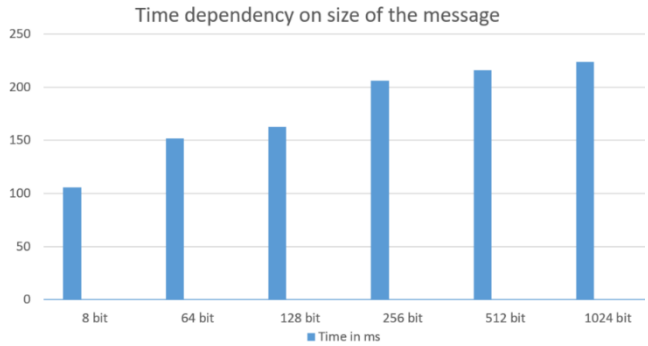

Fig 4:- Time dependency

VII. IMPLEMENTATION OF RSA ALGORITHM

Two version of RSA modules were designed. The first version of the proposed RSA module uses spatial parallelism [19], multiple copies of the same hardware components (implementing RSA steps) implemented in the look-up-table (LUT) form (RSA_LUT). The second solution uses pipelining (RSA_mul16s) [18]. The steps of the RSA algorithm were implemented as pipeline stages. The stages are connected one to the next to form a pipe – the message data block enter at one end, progress through the stages, and exit at the other end. The computations in stages are overlapped in execution. Although each message (message blocks) must pass through all stages, a different message (message block) will be in each stage [20].

The RSA modules were designed in Vivado Design Suite - HLx of Editions 2016.3 development software by using C and VHDL languages. The proposed RSA Cryptographic Accelerator is presented on Fig. 5 and consists of following base components.

ISSN No:-2456-2165

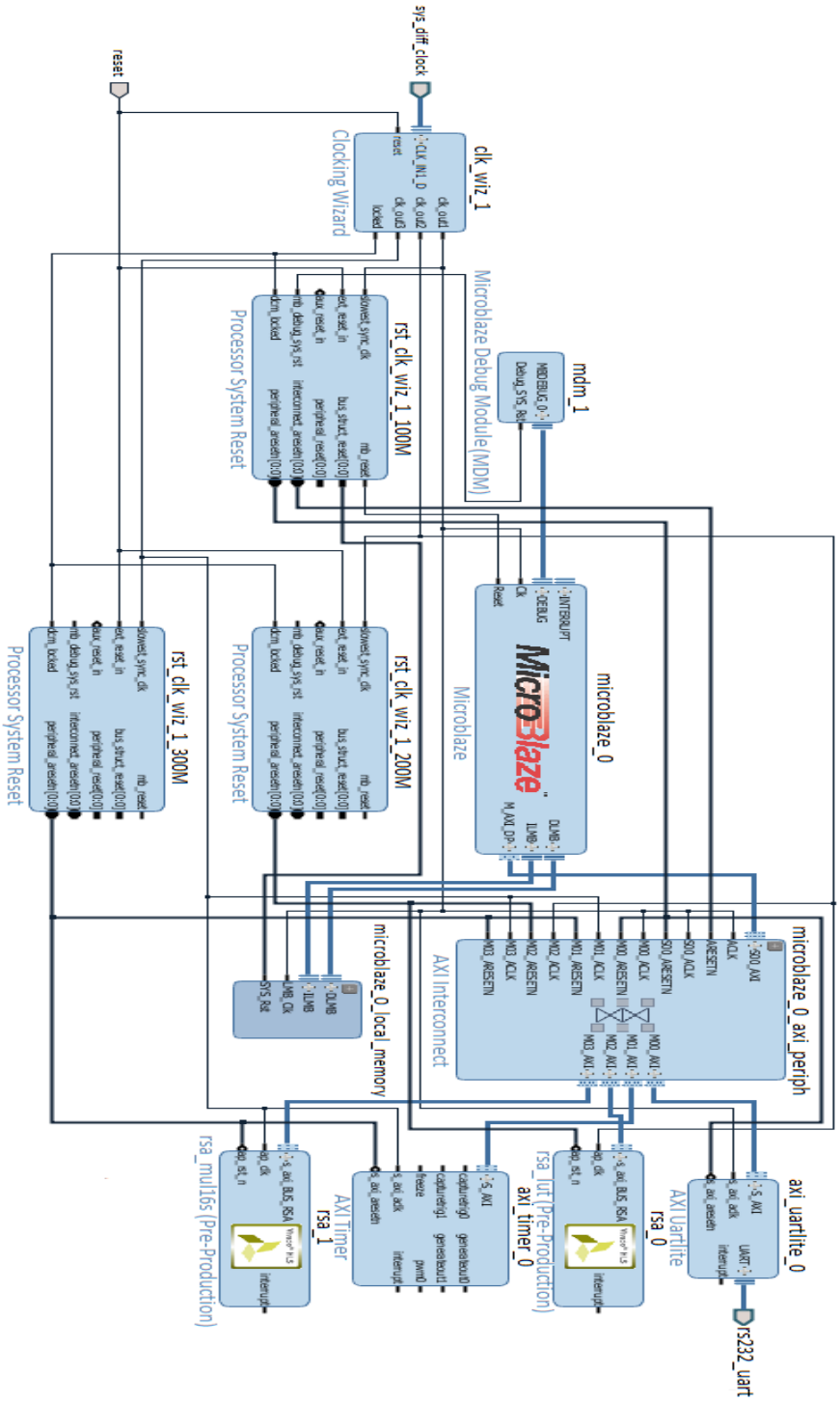

Fig 5:- RSA Cryptographic Accelerator

Microblaze – soft-processor that control the communication between the user and the designed modules.

RSA_LUT – look-up-table (LUT) based RSA module.

RSA_mul16s – RSA module based on pipelining.

Axi Interconnect – module that interconnect another modules with the soft-processor.

Axi Timer – this module is responsible for measure and record the RSA modules performance. It returns the modules response times.

Clocking Wizard – it is responsible for generating synchronous clock signal for the architecture.

Processor System Reset – it provides the reset of components in the proposed architecture.

VIII. EVALUATION OF RSA MODULES

During the testing we had an experience with memory limitation and the final solution is limited to 1024 bits of the key and the message size per one computation cycle using the

selected FPGA device on our designed architectural solution.

Time dependency is shown in Tab. 4. We determined that the bigger the message and key length is, the longer it takes to get cipher. This results is also present in Fig. 6.

Message length/Key length	128 bit	256 bit	512 bit	1024 bit
32 bit	48 527 ms	501 394 ms	500 202 ms	1000404 ms
64 bit	49 710 ms	501 654 ms	500 441 ms	999689 ms
128 bit	501 156 ms	501 156 ms	498 533 ms	1 512 527 ms
265 bit	499 725 ms	1 000 643 ms	2 002 239 ms	3 502 131 ms
512 bit	2 002 001 ms	2 401 582 ms	2 502 441 ms	4 003 048 ms
1024 bit	3 001 928 ms	5 025 854 ms	8 005 381 ms	8 006 811 ms

Table 4. Measured Times – In a row is representation of key

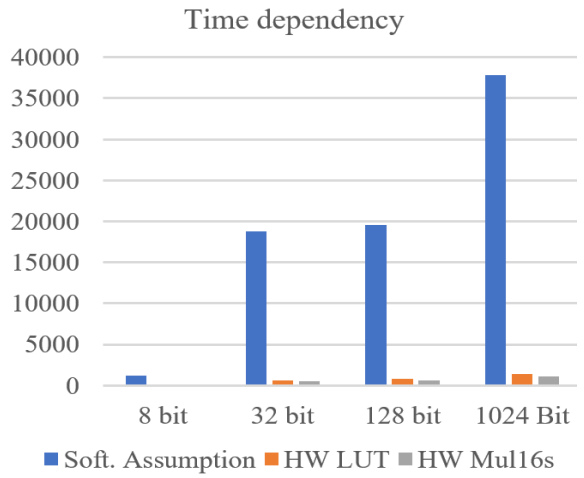

Fig 6:- Time dependency

The highest operation clock frequency that we achieved was 254.46 MHz at RSA_LUT module and 152.77 MHz at RSA_mul16s module.

The maximal usage of CLB units was at our tested FPGA device at maximum rate of three percentages that represents really low number. Our tested architecture is the demonstration of the higher performance then other compared solutions [5-10].

IX. CONCLUSION

The final RSA algorithm has implementation in C and VHDL language. The software development tool we used was the Vivado HLS 2016.3. We applied two approaches to build RSA computing unit. LUT was set as first approach, and RSA_LUT synthesis unit was created. Pipeline processing was second approach to synthetize RSA_mul16 unit. We gave its name based on 16bit logical-arithmetic unit that was used for calculation of the numbers in a field of unsigned numbers. As comparison speed of encryption was created FPGA chip to with rate of management comparing to usage of CPU algorithm. Measured data showed that assumption was fulfilled, and multiple acceleration was present. Also, both RSA modules were compared. The solution of RSA looks up table showed higher working frequency not with higher consumption of CLB. However, this consumption was less than 3%.

This document also shows the possibility of acceleration of another encryption algorithm, DES. This solution was implemented also in C and VHDL. The results shown in Tab III and Fig. 4.

X. ACKNOWLEDGMENT

This work was supported by KEGA Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic under Grant No. 077TUKE-4/2015 „Promoting the interconnection of Computer and Software Engineering using the KPIkit“, Grant No. 003TUKE-4/2017 „Implementation of Modern Methods and Education Forms in the Area of Security of Information and Communication Technologies towards Requirements of Labour Market“. This support is very gratefully acknowledged.

REFERENCES

[1] Anthes, Gary H. "Standard Encryption Vulnerable to Attack." Computerworld 12 February 1996. 25 June 2001. http://www.computerworld.com/cwi/story/0,1199,NAV47_STO13904,00.html.

[2] P. Vishwanath, R. C. Joshi, A. K. Saxena, “FPGA IMPLEMENTATION OF DES USING PIPELINING CONCEPT WITH SKEW CORE KEY-SCHEDULING”. Journal of Theoretical and Applied Information Technology, 2009.

[3] Máire McLooneJohn V. McCannyDes.: DES Algorithm Architectures and Implementations. Springer Science+Business Media New York 2003.

[4] G. Rouvroy, F. X. Standaert, J. J. Quisquater, J. D. Legat, “Efficient Uses of FPGAs for Implementations of DES and Its Experimental Linear Cryptanalysis,“ IEEE TRANSACTIONS ON COMPUTERS, VOL. 52, NO. 4, APRIL 2003.

[5] E. Pietriková, S. Chodarev, “Towards Programmer Knowledge Profile Generation“ Acta Electrotechnica et Informatica. Roč. 16, č. 1 (2016), s. 15-19. - ISSN 1335-8243.

[6] A. H. Ansari, A. R. Landge, “RSA algorithm realization on FPGA”, International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 2, Issue 7, July 2013.

[7] Menezes, Alfred; van Oorschot, Paul C.; Vanstone, Scott A. (October 1996). Handbook of Applied Cryptography. CRC Press. ISBN 0-8493-8523-7.

[8] M. Prerna, A. Sachdeva, “A Study of Encryption Algorithms AES, DES and RSA for Security”. Volume 13 Issue 15 Version 1.0 Year 2013. Double Blind Peer Reviewed International Research Journal. : Global

Journals Inc. (USA) Online ISSN: 0975-4172 & Print ISSN: 0975-4350.

[9] S. Khaled, H. Hussien, S. Yehia, "FPGA Implementation of RSA Encryption Algorithm for E-Passport Application," World Academy of Science, Engineering and Technology International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol:8, No:1, 2014.

[10] A. C. Shantilal, "A Faster Hardware Implementation of RSA Algorithm" Department of Electrical & Computer Engineering, Oregon State University, Corvallis, Oregon 97331 USA.

[11] A. H. Ansari, A. R. Landge, "RSA algorithm realization on FPGA", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 2, Issue 7, July 2013.

[12] A. Shashank, "FPGA Implementation of RSA Encryption and CRT based Decryption using Parallel Architecture," Journal of Innovation in Electronics and Communication

[13] L. Vokorokos, E. Chovancová, "Viacjadrová architektúra zameraná na akceleráciu výpočtov, " Acta Informatica Pragensia. Vol. 2, no. 1 (2013), p. 79-90. - ISSN 1805-4951.

[14] Calderbank, Michael "The RSA Cryptosystem: History, Algorithm, Primes". 2007.

[15] E. Chovancová, N. Ádám, A. Baláž, E. Pietríková, P. Feciľak, S. Šimoňák, M. Chovanec, "Securing distributed computer systems using an advanced sophisticated hybrid honeypot technology," Computing and Informatics. Roč. 36, č. 1 (2017), s. 113-139. - ISSN 1335-9150.

[16] Biham, Eli and Shamir, Adi "Differential Cryptanalysis of DES-like Cryptosystems". Journal of Cryptology. (1991). 4 (1): 3–72. doi:10.1007/BF00630563.

[17] W. Stallings, "Cryptography and Network Security Principles and Practices," Prentice Hall. 2005 0-17-187316-4.

[18] L. Vokorokos, A. Baláž, B. Madoš, "Anomaly and Misuse Intrusions Variability Detection," Acta Electrotechnica et Informatica. Roč. 2010, Č. 4 (2010), S. 5-9. - ISSN 1335-8243.

[19] Xilinx Inc. KC705 Evaluation Board for the Kintex-7 FPGA. 2016 Available web source: https://www.xilinx.com/support/documentation/boards_and_kits/kc705/ug810_KC705_Eval_Bd.pdf

[20] Xilinx. User Guide UG474 (v1.8) September 27, 2016. "7 Series FPGAs Configurable Logic Block," Available web source: https://www.xilinx.com/support/documentation/user_guides/ug474_7Series_CLB.pdf.