Influence of Passive Hardware Redundancy on Differential Power Analysis Resistance of AES Cipher implemented in FPGA

Vojtěch Miškovský, Hana Kubátová, Martin Novotný
Czech Technical University in Prague
Faculty of Information Technology
Department of Digital Design
{miskovoj, kubatova, novotnym}@fit.cvut.cz

Abstract Many electronic systems has to fulfill strict dependability properties, especially both fault tolerance and attack resistance. Intuitively, these requirements may seem to contradict each other. A study and an experiment description of the possible methods how to measure these impacts as well as result of first experiments are presented in this paper. Specifically, how basic passive hardware redundancy design methods affects resistance against differential power analysis attack and how the whole design can be modified to increase attack resistance will be discussed.

Keywords fault tolerance; attack resistance; security; AES; differential power analysis; FPGA

1 Introduction

One of the common digital design requirements is to be fault-tolerant. There are many methods of digital design to achieve fault tolerance in FPGA. They can be based on replication [1], off-line BIST [2], reconfiguration [3] and so on [4].

Another common requirement is attack resistance. This property is important especially for cryptographic devices and it means that the device is protected against disclosure of confidential information, usually the cipher key. In the context of digital design, only the side channel attacks are relevant, because they target the physical implementation of the cryptographic

device rather than the cryptographic properties of the cipher. We focus on resistance against differential power analysis for which also exist many digital design methods like for example masking [5],[6], dual-rail logic [7] and threshold implementation [8].

The target of our research is to investigate the influence of fault-tolerant design methods on attack resistance and vice versa considering the final design of real circuits, especially the ones used in mission-critical applications. New methods or conformation of the present ones to achieve both fault tolerance and attack resistance at the same time with lower overhead will be the main aim of this research.

We will focus on passive hardware redundancy methods in this article. These methods cause design size and power consumption increase. When some kind of security module is designed using these fault tolerance increasing methods, the increased power consumption may undesirably affect its resistance against side channel attacks like differential power analysis.

The important question is, whether this influence is positive or negative. This question cannot be easily answered because more identical modules could increase the signal to noise ratio making the attack easier, but each module may also introduce some extra noise (independent for each module).

This article proposes methods for measuring this influence in order to propose some ways to achieve both fault tolerance and DPA resistance. Also the results of first experiments are presented.

This article contains a description of chosen fault-tolerant methods, AES cipher and differential power analysis in section 2. We will also mention how we have implemented AES and its fault-tolerant variants in section 3. Section 4 describes measuring platform and a control application we developed. Finally, we will present the results, summarize our progress and look at our future plans (sections 5, 6 and 7).

2 State of the Art

There is are multiple ways to make FPGA design about fault tolerant, for example replication [1],[9], off-line BIST [2],[10], reconfiguration [3],[11]. We also know how to increase and attack resistance like masking [5],[6], dual-rail logic [7],[12],[13], frequency switching [14], threshold implementation [8],[15] and so on, but we are not familiar with any work related to the mutual influence of them.

The current research is based on passive hardware redundancy fault-tolerant design methods [16], AES cipher [17] and differential power analysis

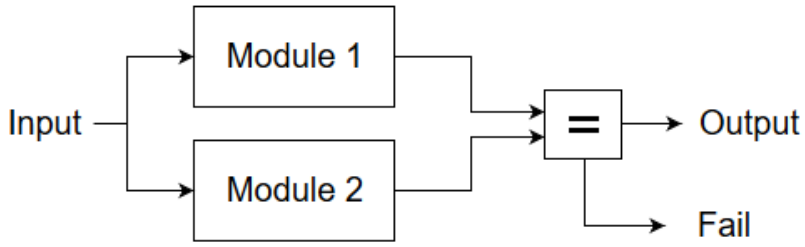

Figure 1: Schema of duplex

[18].

2.1 Fault-tolerant design methods

We decided to start with passive hardware redundancy methods [16]. These methods are based on multiplication of the whole logic module and they are usable and easily implemented in FPGA [1]. They have high impact on power consumption and they are commonly used.

2.1.1 Duplex

Duplex consists of two modules. It compares outputs of modules and signals a failure when they differs. It can detect a failure of one module. A schema of duplex is shown in Figure 1.

2.1.2 Triple Modular Redundancy (TMR)

TMR is based on three modules and a voter. The voter provides a majority vote of outputs. When a single module fails the system remains fully operational. When two modules fail the whole system fails. A schema of TMR is shown in Figure 2.

2.1.3 N-Modular Redundancy (NMR)

NMR is a generalization of TMR. It uses $N = n \times 2 + 1$ modules and a voter. The n is the number of modules which failure is tolerated by NMR. Therefore an odd number of modules should be used to secure the existence of a majority.

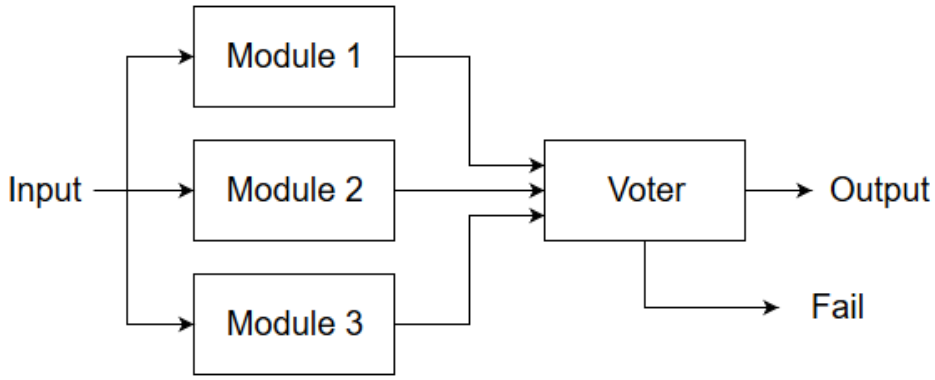

Figure 2: Schema of TMR

2.2 AES

AES (advanced encryption standard) is a symmetric block cipher, which is one of the most common ciphers used for example for securing wireless networks. It uses fixed block size (128 bits) and one of three key sizes (128, 192 or 256 bits) [17]. We use the 128bit version.

AES algorithm for 128bit key consists of ten rounds and initial transformation. At each round different key derived from the initial one is used.

A schema of AES cipher is shown in Figure 3. For more detailed information see the AES specification [17].

2.3 Differential Power Analysis

DPA belongs to a group of attacks known as the side-channel attacks. It is based on measuring power consumption traces of a device and a calculation of the secret information (key) by an application of statistical functions on obtained power traces. DPA requires plain text or cipher text to be known for each power consumption waveform [18].

We will use an advance type of DPA using correlation coefficients for calculations (sometimes called correlation power analysis (CPA) or Multi-Bit DPA) [19]. We can divide it into three phases: the measurement phase, the hypothesis phase and the calculation phase.

2.3.1 Measurement phase

We have to measure power consumption of device during the encryption in this phase. We have a list of n plain texts/cipher texts and a matrix of

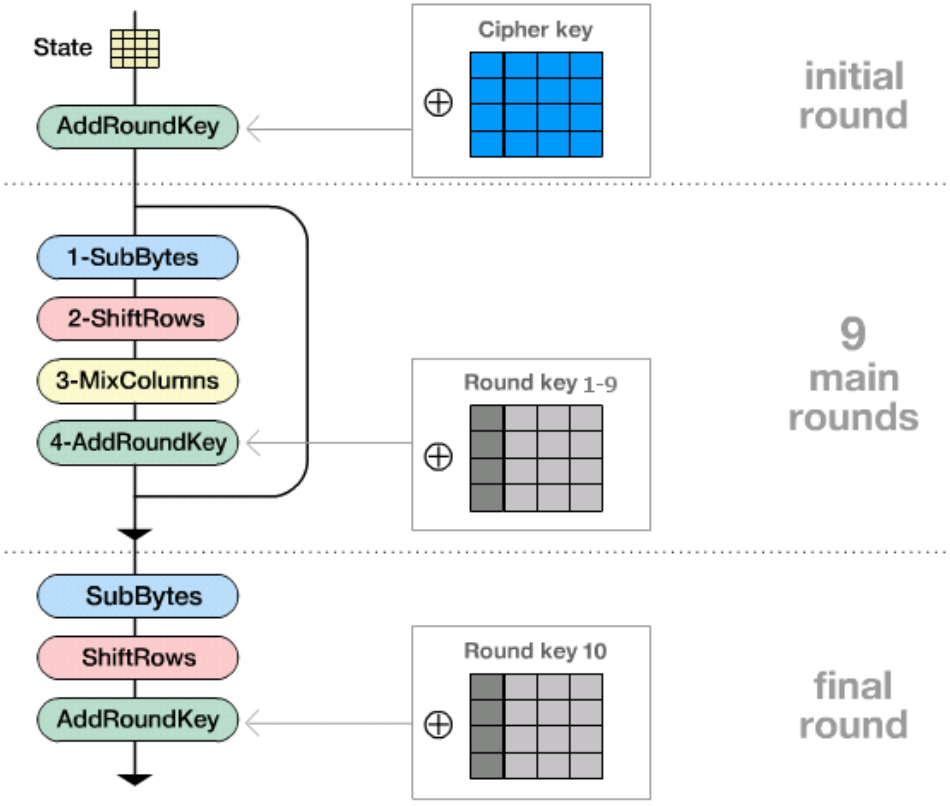

Figure 3: Schema of AES

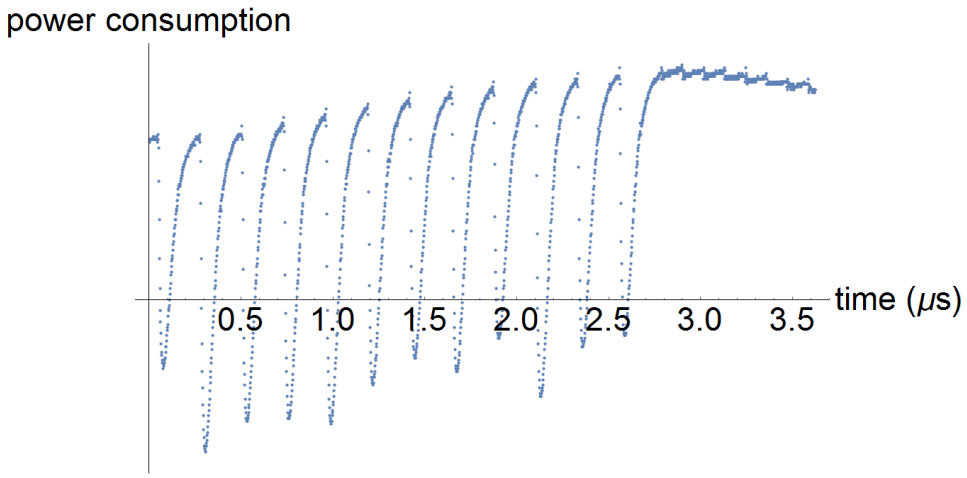

Figure 4: Example of samples of single power trace

$t \times n$ elements as a result of this phase, where n means number of measures (waveforms) and t means count of power consumption values (traces) for each measure. An example of a waveform is shown in Figure 4.

2.3.2 Hypothesis phase

We need to split the key into parts (for example bytes) and make some prediction about power consumption for each possible value of each part of key (e.g 256 possible values for one byte). Then we need to choose a power model which must be a function dependent on plain/cipher text and the key, it should include some non-linear transformation and it should represent some inner state of the cipher. For example when we are analyzing the AES cipher we can split the key into bytes, then for each byte XOR is applied on each possible value (0-255) with corresponding byte of plain text, then SubBytes function is applied [17]. It corresponds with the AES initial transformation. Now we can use Hamming weight of the result as our hypothetical value.

During the hypothesis phase we will produce $k \times n$ values for each part of the key, where k means a number of possible values of the selected part of the key.

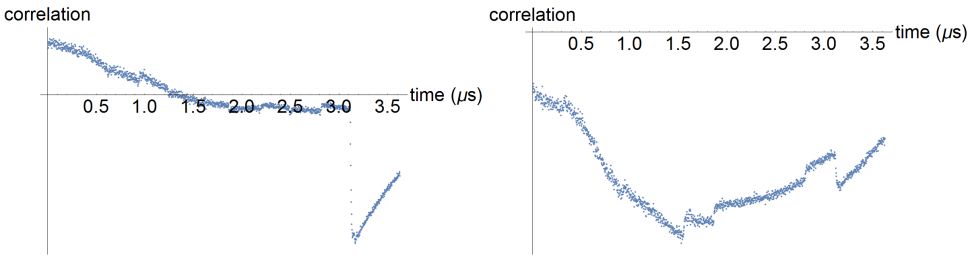

Figure 5: Example of correlation plot in time for correct key (left) and wrong key (right)

2.3.3 Calculation phase

Now we need to evaluate, which one of the possible key values is the correct one.

We have the hypothetical value matrices for each part of the key and the power consumption matrix. Now we need to apply a vector correlation function on each row of hypothetical value matrix with each row of power consumption matrix. This results into $t \times k$ correlation matrix for each part of key. Now if we have used enough power traces the key can be revealed. We can find maximal absolute value of the correlations, some statistical function or just eye exploration to guess the key. An example of correlation plot for correct and incorrect key is shown in Figure 5.

3 Implementation

In this section we will describe the implementation of the AES cipher and it's fault-tolerant variants.

3.1 AES

AES encryption was implemented to take ten clock cycles (one clock cycle for each round). The round itself is implemented as a combination logic divided into entities corresponding to the transformation functions of each round [17].

Each round key is generated in the same clock cycle as the round using it is processed.

Due to this we have one set of registers containing the current round key and one set of registers containing the current cipher state. The presumed

content of the current state register will be used for calculating hypothetical values in differential power analysis.

One pin on the Evariste module is dedicated to indicate an active encryption and it will serve as a trigger signal for the oscilloscope measurement.

3.2 Fault-tolerant variants

We have implemented only simple realizations of fault tolerant variants using multiple of the same copies of AES module so far.

Duplex is implemented as two copies of AES module, which outputs are compared and if they are not equal the fail signal is set.

NMR is implemented as a parametric entity with parameter N representing number of modules. It is realized as N copies of AES module and a voter. The voter represents a set of comparators comparing all k-combinations of N outputs, where k is the lowest majority $k = (N-1)/2$, and a multiplexer deciding which output is the correct one or whether fail signal should be set. It is realized by a recursive function generating both the comparators and the multiplexer.

We also implemented non-parametric TMR to simplify synthesis when NMR with only three AES modules is required.

4 Measurement

In this chapter we will describe the FPGA platform and the hardware architecture of measurement, we will present software for fast DPA calculations we developed and also the actual experiment will be subscribed.

Measurement methods were tested on a simple AES implementation. The Hamming distance between the state register value at the ninth and the tenth round was chosen as the power model used for the hypothesis phase of DPA. According to [20] it is the best choice to be used for DPA against FPGA.

4.1 FPGA platform

Evariste III [21] was chosen as an implementation platform. It provides USB communication and FPGA module with connectors for power consumption measurement. VHDL examples of USB communication are provided, too. Our FPGA module contains Altera Cyclone III. This device's typical clock frequency is 48MHz but for the DPA measurement we set it to 6Mhz. The whole platform is shown in Figure 6.

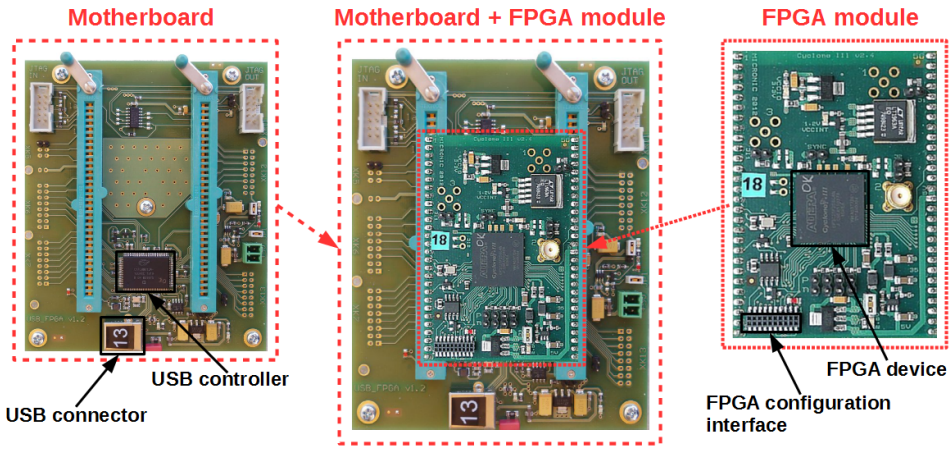

Figure 6: Evariste platform setup

We adjusted VHDL USB communication examples provided along with Evariste module to be able to handle changing cipher key, receiving the plain text and sending the cipher text back to the PC.

4.2 Hardware architecture

We chose an oscilloscope PicoScope 6404D for the measurement. It provides sampling rate 5GS/s (2.5GS/s when two channels are used). We use 500MS/s sampling rate for the measurement.

The whole hardware architecture is shown in Figure 7.

4.3 Software

Wolfram Mathematica for correlation calculations, USB communication script provided by Evariste authors for communication and PicoScope software for power measurement proved to be too slow to make serious measurements of DPA against FPGA implementation. Compared to SmartCards or other MCU implementations, FPGAs and ASICs demand much more power traces to be obtained. Therefore we decided to develop our own measurement software.

Our software is responsible for the generation of the data, the communication with the encryptor, the oscilloscope measurement and the power analysis. We chose C programming language, since it's really performance-efficient and all necessary APIs are provided in C.

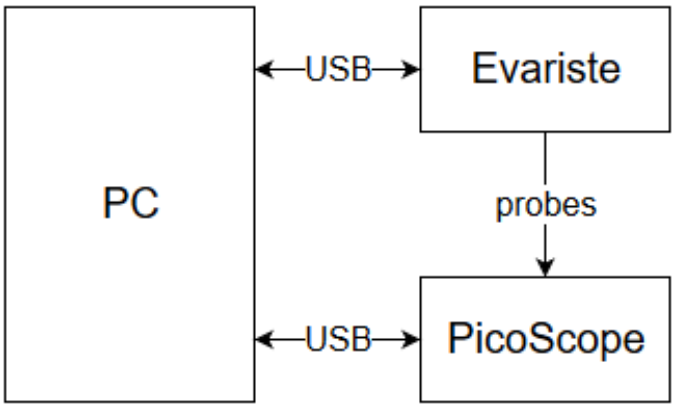

Figure 7: Measurement hardware architecture

Table 1: Duration of measurement and calculation in seconds for various counts of power traces (500 samples per trace)

	100	**1,000**	**10,000**	**100,000**	**1,000,000**
Mathematica	4	37	347	n/a	n/a
Our software - 1 core	<1	2	19	189	1,890
Our software - 4 cores	<1	<1	6	58	599

We replaced proprietary USB driver of Evariste device by WinUSB driver [22], so we were able to use libusb library [23] for communication with the encryptor. The communication protocol remains the same as the original one used by Evariste program.

Due to the oscilloscope C API it was relatively simple to achieve about a hundred waveforms triggered per second and there is still some reserve for improvements.

This application implements a highly optimized and fully parallelized correlation function running much faster than our original script written in Mathematica. It also allows incremental calculations, therefore if we are not able to reveal the key, we can measure more traces, and continue the calculation at the point we stopped.

The whole software architecture of the advanced method is shown in Figure 8. This method accelerated the measurement dozens of times as is shown in Table 1. It is even more obvious for the higher counts of waveforms.

More detailed information about this software is about to be published.

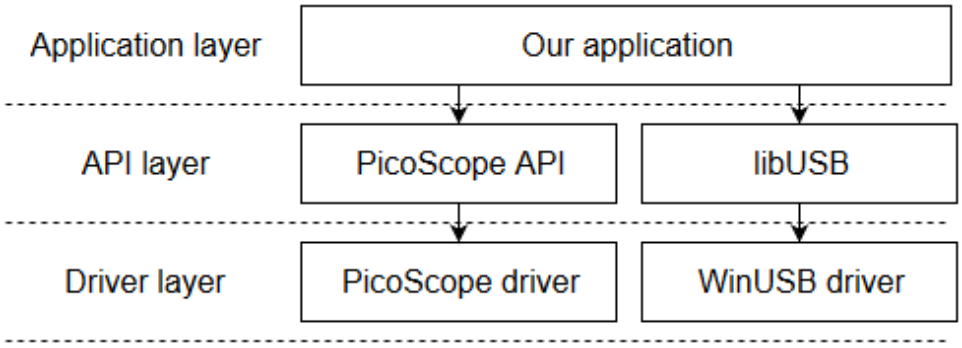

Figure 8: Measurement software architecture

4.4 Experiment

We made the experiment on single module implementation, duplex implementation and TMR implementation of AES. For each implementation we measured 50 different sets of power traces and then for each set we observed how many traces are enough to reveal the correct key, so for each implementation we got 50 counts of needed traces (hereinafter referred to as *minTraces*).

5 Results

As mentioned in section 4.4, we measured 50 *minTraces* for each implementation differing in the number of modules used. We made a histogram of the collected *minTraces* which can be seen in Figure 9. Averages and variances of the data can be seen in Table 2. According to these it seems that this basic hardware redundancy design methods have negative influence on resistance against DPA, because the value of *minTraces* is lower when multiple modules are used. Nevertheless this influence is quite small. Higher influence could have been expected, because multiple same modules with similar power consumption should increase the signal to noise ratio of the power consumption, but in fact the higher overall power consumption could also increase the noise level. This explanation also corresponds with the variance of *minTraces* which is increased when more modules are used, as can be also seen in Table 2 and in the histogram in Figure 9.

According to these results, passive hardware redundancy techniques do not compromise resistance against differential power analysis of the design. Therefore we expect that we can use these design techniques to make attack

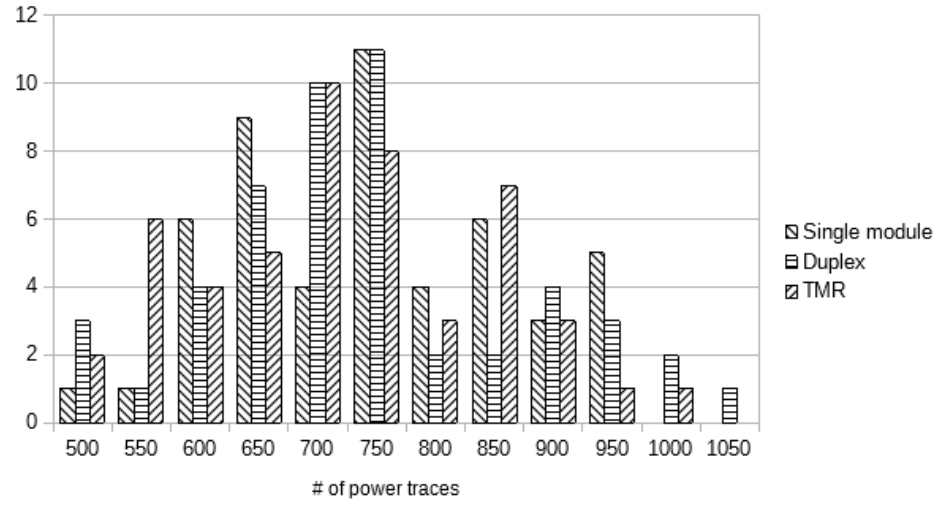

Figure 9: Histogram of *minTraces*

Table 2: Comparison of averages and variances of minimal numbers of power traces needed to reveal the correct key

	Single module	**Duplex**	**TMR**
Avg. of *minTraces*	745.28	741.74	719.22
Var. of *minTraces*	13,281.59	18,222.03	14,802.38

resistant digital design also fault tolerant at the same time.

6 Conclusions

We have implemented an AES encryptor and basic passive hardware redundancy methods in VHDL considering the fact that we need to measure its power consumption. We chose suitable device for the measurement.

Also we have programmed an application serving the whole measuring and calculation process of differential power analysis, which is fast and easy to use.

We attacked our fault-tolerant variants of AES implemented in FPGA. The influence of passive hardware redundancy on DPA resistance seems to be negative but very small. It shows that the signal to noise ratio increased by identical modules is almost in balance with the extra noise introduced by

each module (independent of the other modules). The difference of power traces needed is so small that we can imagine the influence could be even positive with some other implementations.

Considering these facts we did not reject these design methods as candidates for being part of the future design methods combining fault tolerance and attack resistance.

7 Future work

One of the next steps will be an implementation of fault-tolerant variants using divergent AES modules. For example duplex with AES modules mutually masking power consumption should considerably increase attack resistance. Also another fault-tolerant design methods will be considered.

Other possible continuation of the research is to investigate the influence of countermeasures against DPA or other side-channel attack on reliability of the design.

Another important approach is to verify these experiments on mathematical base. This would entail the need of deep knowledges of FPGA chips design and behavior, which is not well documented and is really hard to determine, but we plan to try to research this way too.

Having collected information from these parts of research we could be able to propose some new or improved methods for combining both fault tolerance and attack resistance and make the mission-critical security devices more efficient.

Acknowledgment

This research has been partially supported by the grant GA16-05179S of the Czech Grant Agency, "Fault-Tolerant and Attack-Resistant Architectures Based on Programmable Devices: Research of Interplay and Common Features" (2016-2018) and CTU project SGS17/017/OHK3/1T/18.

References

[1] L. Anghel, D. Alexandrescu, and M. Nicolaidis, "Evaluation of a soft error tolerance technique based on time and/or space redundancy," in *Integrated Circuits and Systems Design, 2000. Proceedings. 13th Symposium on*, pp. 237–242, IEEE, 2000.

[2] J. Smith, T. Xia, and C. Stroud, "An automated BIST architecture for testing and diagnosing FPGA interconnect faults," *Journal of electronic testing*, vol. 22, no. 3, pp. 239–253, 2006.

[3] J. Lach, W. H. Mangione-Smith, and M. Potkonjak, "Low overhead fault-tolerant FPGA systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 6, no. 2, pp. 212–221, 1998.

[4] E. Stott, P. Sedcole, and P. Y. Cheung, "Fault tolerant methods for reliability in FPGAs," in *2008 International Conference on Field Programmable Logic and Applications*, pp. 415–420, IEEE, 2008.

[5] T. S. Messerges, "Securing the AES finalists against power analysis attacks," in *International Workshop on Fast Software Encryption*, pp. 150–164, Springer, 2000.

[6] J. Blömer, J. Guajardo, and V. Krummel, "Provably secure masking of AES," in *International Workshop on Selected Areas in Cryptography*, pp. 69–83, Springer, 2004.

[7] D. Suzuki and M. Saeki, "Security evaluation of DPA countermeasures using dual-rail pre-charge logic style," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 255–269, Springer, 2006.

[8] S. Nikova, C. Rechberger, and V. Rijmen, "Threshold implementations against side-channel attacks and glitches," in *International Conference on Information and Communications Security*, pp. 529–545, Springer, 2006.

[9] F. G. de Lima Kastensmidt, G. Neuberger, R. F. Hentschke, L. Carro, and R. Reis, "Designing fault-tolerant techniques for SRAM-based FPGAs," *IEEE design and test of computers*, vol. 21, no. 6, pp. 552–562, 2004.

[10] A. Alaghi, M. S. Yarandi, and Z. Navabi, "An optimum ORA BIST for multiple fault FPGA look-up table testing," in *2006 15th Asian Test Symposium*, pp. 293–298, IEEE, 2006.

[11] J. Emmert, C. Stroud, B. Skaggs, and M. Abramovici, "Dynamic fault tolerance in FPGAs via partial reconfiguration," in *Field-Programmable Custom Computing Machines, 2000 IEEE Symposium on*, pp. 165–174, IEEE, 2000.

[12] Z. Chen and Y. Zhou, "Dual-rail random switching logic: a countermeasure to reduce side channel leakage," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 242–254, Springer, 2006.

[13] J.-L. Danger, S. Guilley, S. Bhasin, and M. Nassar, "Overview of dual rail with precharge logic styles to thwart implementation-level attacks on hardware cryptoprocessors," in *Signals, Circuits and Systems (SCS), 2009 3rd International Conference on*, pp. 1–8, IEEE, 2009.

[14] S. Yang, W. Wolf, N. Vijaykrishnan, D. N. Serpanos, and Y. Xie, "Power attack resistant cryptosystem design: a dynamic voltage and frequency switching approach," in *Proceedings of the conference on Design, Automation and Test in Europe-Volume 3*, pp. 64–69, IEEE Computer Society, 2005.

[15] A. Moradi, A. Poschmann, S. Ling, C. Paar, and H. Wang, "Pushing the limits: a very compact and a threshold implementation of AES," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 69–88, Springer, 2011.

[16] D. Pradhan, *Fault-tolerant computer system design*. Upper Saddle River, N.J: Prentice Hall PTR, 1996.

[17] N. F. Pub, "197: Advanced encryption standard (AES)," *Federal Information Processing Standards Publication*, vol. 197, pp. 441–0311, 2001.

[18] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Annual International Cryptology Conference*, pp. 388–397, Springer, 1999.

[19] M.-L. Akkar, R. Bevan, P. Dischamp, and D. Moyart, "Power analysis, what is now possible...," in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 489–502, Springer, 2000.

[20] L. T. McDaniel III, *An investigation of differential power analysis attacks on FPGA-based encryption systems*. PhD thesis, Virginia Polytechnic Institute and State University, 2003.

[21] V. Fischer, F. Bernard, and P. Haddad, "An open-source multi-FPGA modular system for fair benchmarking of true random number generators," in *Field Programmable Logic and Applications (FPL), 2013 23rd International Conference on*, pp. 1–4, IEEE, 2013.

[22] Microsoft, “Winusb (winusb.sys),” dec 2016.

[23] Libusb, “Libusb-1.0 api reference,” dec 2016.