# Analysis and Evaluation of PUF-Based SoC Designs for Security Applications

Alexandra Stanciu, Marcian Cirstea, *Senior Member, IEEE,* Florin Moldoveanu, *Member, IEEE*

***Abstract*—This paper presents a critical analysis and statistical evaluation of two categories of Physically Unclonable Functions (PUFs): ring oscillator PUF and a new proposed adapted latch based PUF. The main contribution is that of measuring the properties of PUF which provide the basic information for using them in security applications. The original method involved the conceptual design of adapted latch based PUFs and ring oscillator PUFs in combination with peripheral devices in order to create an environment for experimental analysis of PUF properties. Implementation, testing and analysis of results followed. This approach has applications on high level security.**

***Index Terms*—FPGA, security, physical unclonable function, temperature, ring oscillator, latch**

## I. Introduction

In the past two decades, advances in programmable device technologies, both hardware and software areas, have been extraordinary [7]. The original application to rapid prototyping in a large number of areas such as: medical apparatus, automotive industry, industrial control systems, remote sensing, data complex computations, or robotics has been complemented with a large number of new applications in the security domain [8 9 10]. New security methods are based on PUFs (Physically Unclonable Functions). A mutual authentication scheme between the FPGA chip manufacturers and the IP providers is presented in [13]. In [21] the authors described a method for Authentication via CRP (Challenge-Response Pairs). PUFs can be also used to extract chip-unique signature and volatile secret keys which can be used in cryptographic protocols or authentication schemes. An overview of the most important implementation details and experimental results for the PUFs is presented in [1, 12]. An important analysis regarding the techniques for design and implementation of an FPGA PUF circuit is presented in [11]. The present paper describes the implementation details and experimental analysis and results regarding a ring oscillator PUF(a delay based PUF) and an adapted Latch based PUF(a memory based PUF) to fulfill an FPGA practical approach.

Two PUF circuits suitable for an FPGA implementation, described in the literature, are: SRAM PUF and PDL PUF. Experiments regarding the SRAM PUF are presented in [1] but this PUF suffers from two main drawbacks: i) the SRAM PUF is not available on all mainstream FPGA platforms (because no uninitialized SRAM is available on most types) [23]; nowadays, most of the FPGA manufacturers reset the start-up state of the SRAM cells to a known value, rendering the SRAM-PUF difficult to be used [17, 18]; ii) the response of the SRAM PUF is generated only on the power-up state of the circuit, the output of the PUF circuit could not be re-generated while the circuit is operational [16]. The authors of [17] attempt to remove the fundamental obstacle Xilinx FPGAs have – from the SRAM-PUF point of view - by using a work-around that enables to disconnect uninstantiated BRAM from the power supply network. The complex mechanism is based on 3 partial reconfigurations and the results regarding the uniqueness are not successful. As an alternative to SRAM PUF, memory based PUFs that use the basic principle of SRAM PUF started to appear: SR Latch PUF, D Flip-Flop PUF, Butterfly PUF or Buskeeper PUF cell [1]. All of these require careful routing in order to be implemented on FPGA. For FPGA, programming is done through logic blocks and interconnects; the gate level structure cannot be accessed, in order to exploit layout design techniques.

Programmable delay lines (PDLs) are introduced in [19] in order to accurately equalize the signal arrival times to flip-flop. PUF circuits based on identical and symmetrical interconnections may use PDLs in order to adjust their values of propagation times. In [20], the authors use PDL to cancel out delay skews caused by asymmetries in routing and systematic variations for an Arbiter PUF. Their mechanism of adjusting delays is complex and requires a non-negligible effort to implement in practice. In the present work, the efforts were focused on trying to find out identical or slightly different interconnections. This paper presents the experimental results obtained by considering the interconnects inside PUFs slightly differently. The PDLs are a novel mechanism that may help constructing PUFs when the identical or slightly different interconnections could not be achieved. In this work, the correct timing was accomplished without PDLs.

According to [1], there are two delay based PUFs: the Arbiter PUF and the Ring Oscillator PUF. There are some differences between the presented implementation and testing approach and other experiments, which are described briefly below. The Ring Oscillator circuit is analyzed in this paper on two different FPGA families: Spartan 3E and Spartan 6. The Spartan 3E was released in 2004, on 90-nanometer copper process technology [15]. The Spartan 6 is built in 2009, on a 45-nanometer, 9-metal layer and dual-oxide process technology [14]. Besides other experiments, details of implementation and measurements of propagation times through gates and interconnections are presented for these two FPGA families. There are differences in FPGA architectures between families and the results presented in this paper show that PUF circuits may be implemented despite architecture differences; moreover, the uniqueness and reliability properties are not affected by those differences.

moreover, the uniqueness and reliability properties are not affected by those differences. The random level of the PUF responses, and by default the uniqueness property, is increased due to differences in the FPGA family architecture. Also, the properties of PUF circuits are analyzed along IP cores and digital circuits with different complexities because one of the main applications is to protect critical information acquired and processed in complex System-on-Chips implemented on FPGAs using the generated PUF based secret key. Even if the RO PUF is designed and implemented as a hard macro, and when duplicated all routing and logic resources will remain identical, the other digital components used to obtain the response of the PUF circuits (e.g. counters) are placed and routed by the synthesis tool. These changes may be reflected in the reliability of RO PUF circuits. The paper also explains how the digital circuit activity can influence the RO PUF responses. The internal activities of the circuits produce delay variations (nano-variations) through gates and interconnections between gates. The process variations also affect the electrical parameters of logical gates and interconnections and by default the delays are affected by the internal activities. The paper shows that the uniqueness property of PUF circuits is obtained through both types of variations and analyzes how the reliability property responds to both types of variations.

The latch based PUF is not widely researched. Due to its requirements of symmetry, it is difficult to implement it on FPGA devices. There are few results in the research literature regarding the latch based PUFs implemented on FPGAs. In [5], a design and implementation of a true random generator is presented, which exploits the metastability of the RS latch for an FPGA device. In [6], the authors introduced a novel structure with random latches for generating high-entropy responses using randomness. The article mentioned above [6] does not include implementation details; therefore, it is unclear how the identical and symmetrical interconnections are obtained. The aim of the present paper is to implement and evaluate another possible PUF circuit appropriate for different FPGA architectures. After an exhaustive analysis of PUF results presented in dedicated literature and after many attempts to obtain symmetrical and identical routes using the Xilinx FPGA Editor, it has been concluded that another good candidate for FPGA PUF is a circuit with minimum requirements of identical interconnects. The paper presents the possibilities to obtain interconnections between logic blocks, with related propagation times, and use them in order to construct a latch based PUF circuit.

PUFs are also susceptible to temporal variations such as changes in voltage, temperature and silicon aging, which makes it difficult to reliably produce the correct PUF output at all times. Circuit operation also induces power supply voltage variations, namely power supply noise [2]. When an input pattern signal is applied to a circuit, it will generate a large number of switches in the circuit. These will increase the dynamic power consumption, causing a voltage drop on power lines and a voltage increase on ground lines. This effect is known as power supply noise. Temperature distribution also depends on the location of the switches [3]. Even by applying the same signal patterns to the inputs of the same digital circuit implemented on different chips, it will induce different temperature and voltage drops.

This paper presents a conceptual design and a statistical analysis of two PUF circuits using 30 identical Spartan 3E Starter Boards and a Spartan 6 board. An experimental environment (test bed) is conceptually designed and implemented in order to analyze how much the PUF answer reproducibility is influenced by the running SoC (System on Chip) or by the complexity of the RTL (Register Transfer Level) design. In order to implement the protective method based on PUFs for FPGA applications, PUF circuits must be placed and routed on the unused hardware resources, near the complex logic which implements the entire system. Thus, this paper also focuses on how the digital circuit activity can influence the PUF responses, because one of the main applications is to protect critical information that is acquired and processed in complex SoCs implemented on FPGAs using the generated PUF based secret key.

The original contributions to knowledge presented in this paper can be summarized as consisting of the following: i) the method for obtaining identical and symmetrical routes in order to construct a PUF circuit on two different FPGA architectures without programmable delay lines; ii) a novel mechanism making the latch based PUF a good candidate for FPGA implementation; iii) an environment for the experimental analysis of PUF properties along with analysis results; iv) an original theoretical perspective demonstrating that the internal activity of the digital system implemented on FPGA contributes, along the process variations, to the uniqueness and randomness of the PUF responses.

## II. System concept

A new system concept for securing the FPGA System on Chip (SoC) is introduced. This is based on encrypted and authenticated communication between SoC peripherals. One system composed of two microprocessors and some peripherals may be used as example. The peripherals may be further classified in two domains: the first domain is composed of Microblaze1, RS232 controller, DRAM memory and cryptographic coprocessor and the second domain contains Microblaze2, Ethernet controller, DRAM memory and cryptographic coprocessor. The first domain is considered as the critical domain, in which data are sensitive and must be exchanged considering the encryption and authentication set of rules. Each domain has a secret key generated using PUF circuits. The peripherals from the same domain have access to the domain’s secret key and a wrapper with cryptographic algorithms in order to authenticate, encrypt or decrypt the messages send or received, based on the domain’s secret key. The method may be also used when there is more than one critical domain. For each domain a distinct key, based on PUF circuits, must be generated. The present article analyses the possibility of generating distinct PUF keys on the same FPGA device: one for each domain. The system described before (or another one with more critical domains) is implemented on a single FPGA device and each domain has a PUF secret key. So there may be more PUF secret keys on the same device. In order to use those secret keys in a security application, the correlation of these PUF secret keys must be analyzed. This is done by also measuring the uniqueness and reproducibility, by considering one board divided into 4 distinct parts or 4 distinct identical small devices. This concept may also be applied to a more complex system.

## III. SYSTEM DESIGN AND IMPLEMENTATION

The method followed for the design and implementation of the PUF circuits and the experimental environment consists of 3 steps:

### *A. PUF Design Method using manual placement and routing on FPGA resources*

Two ring oscillators generate periodical signals with frequencies freq1 and freq2, which are connected to the clock input of two counters. Ideally, the freq1 and freq2 frequencies are identical, and therefore the two counters should reach their maximum value at the same time. However one of the counters will arrive first at the maximum value and the comparator's output will be set to 1 or 0, depending on the process variations. Since producing a ring oscillator response generally involves a physical measurement, there are a number of unwanted physical side effects, produced by temperature or voltage variations, which could interfere. Due to the fact that adjacent FPGA locations are influenced by the similar environmental condition, ring oscillators placed in adjacent locations on the FPGA are compared.

The ring oscillator is implemented as a hard macro and the gates and interconnections between them are manually placed and routed. Fig. 1 and Fig. 2 show the estimated delays between gates and interconnections provided by the Xilinx FPGA Editor. The implementation is made on Spartan 3E XC3S500E device and Spartan 6 XCSLX45 device.

The ring oscillator generates a periodical signal. The periodically generated signal is given in (1).

$$t = tp_{invertor} + tp_{\mathrm{int}erconexion} + \Delta_{process\_\mathrm{var}iations} \qquad (1),$$

where $tp_{inversor}$ is the CMOS inverter propagation delay, $tp_{conexiuni}$ is the propagation delay on interconection line and $\Delta_{proces\_fabricare}$ are the process variations.

The latch based PUF circuit is composed of a digital latch and a capture signal which could be the terminal counter signal generated by a counter when it over- or under- flows. Starting from the premise that interconnections between gates are identical and symmetrical and applying a 'high' active signal on the circuit's input, the two outputs of the NAND gates will oscillate. In reality, even with the NAND gates and the interconnections between them manually placed and routed, there will be differences between the interconnections, and the two outputs of the NAND gates will randomly oscillate. On account of the FPGA routing complexity and limitations, delay differences between the interconnections start to appear. The output of the latch circuit is thus biased because of this inconvenience. In order to obtain an unpredictable response from the latch, despite the delay differences, the two NAND outputs oscillate for a period of time and their values are captured in a moment specified by a capture signal. The oscillating period depends on how long the applied input is active 'high'. The output response depends on the process variations due to the following: i) the signal's oscillating period T depends on the propagation time of the NAND gates $t_{NAND}$, on the propagation time of the interconnect routes $t_{interconnects}$ and on the process variations

$_1$ ii) the propagation time of the interconnect between the circuit that generates the oscillating signal and the NAND gates is also influenced by the process variations $\Delta p_2$. iii) the propagation time of the interconnect between the circuit that generates the capture signal and the outputs of the NAND gates is also determined by the process variations $\Delta p_3$. The process variations from three distinct places $\Delta\ _1,\ _2,\ _3$ determine the unpredictable response. Fig. 3 and Fig. 4 present the implementation details of latch based PUF as a hard macro component. The gates, the flip-flops and the interconnections between them were manually placed and routed. Many attempts were made in order to achieve the optimal and correct timing, including tricks like the ones presented in PDLs but without selection. However, the propagation time values discussed in this paper and in other research articles are estimated values by the Xilinx FPGA Editor. In reality, those values may have slight differences.

### *B. Standalone PUF ID design on FPGA device*

In order to measure the two important PUF properties, uniqueness and reproducibility, two cases are considered.

In the first one, n PUF circuits at one moment are considered (Fig. 5a). The answer is measured and then the PUF circuits are relocated to other FPGA hardware configuration logic blocks and the measurements are repeated. There are 128 PUFs in case of the ring oscillator circuit and 77 PUFs in case of the Latch based circuit for one measurement. In the second case, the FPGA is divided in 4 parts and 4 unique identifiers are considered. The FPGA may be viewed as 4 distinct devices (Fig. 5b). Compared with the previous scenario, the 4 unique identifiers are placed simultaneously on the device. The 4 unique identifiers are each of 32-bits length.

In both cases, the PUF answers which compute the unique identifier are observed using the Xilinx ChipScope Analyzer. The binary sequences generated using 128 Ring Oscillator PUFs or 77 latch based PUFs were tested using temperature variations. The commercial temperature range is between 0 Celsius degree and 85 Celsius degrees while the Industrial temperature range is between -40 Celsius degree and 100 Celsius degree. The FPGA device was gradually heated using a hairdryer in a range of temperature from 22 Celsius degree to 78 Celsius degree. The FPGA was also cooled to -5 Celsius degree. The number of unreliable bits was observed in these conditions. Spartan 3E XC3S500E devices are used in order to measure the uniqueness and the reproducibility. The measurements are made on 30 identical devices.

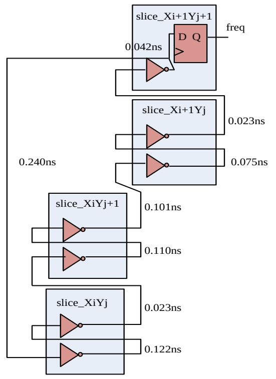


Fig. 1. RO Spartan 3E.

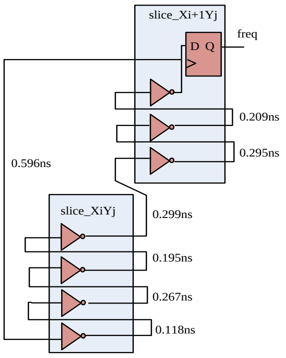


Fig. 2. RO Spartan 6.

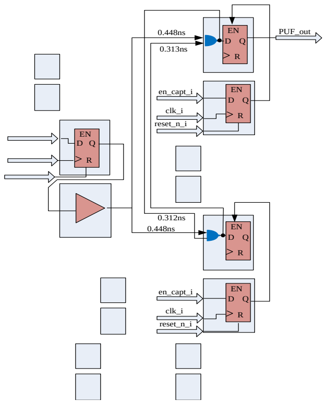


Fig.3. Latch Based PUF Hard Macro - Spartan 3E.

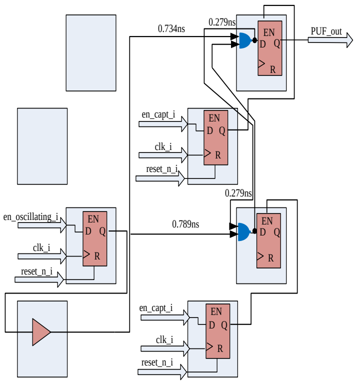


Fig. 4. Latch Based PUF Hard Macro - Spartan 6.

### C. *PUF ID design in combination with RTL complex system or System on Chip on FPGA devices*

Generally a system-on-chip, which also includes security technology based on PUF, contains: i) the logic that implements the functionality of the system and ii) the PUF circuits and some additional logic in order to process the PUFs' answer. The logic which implements the functionality of the system could be a larger digital system implemented in hardware description languages such as phone switches, factory controllers, large stationary installations (traffic lights) or systems based on microprocessors which run complex software/firmware programs. There are two issues that must be analyzed: i) if the PUF together with the complex logic of digital circuits, placed on FPGA hardware resources, have the same answers as the PUF placed on FPGA hardware resources without the complex logic of digital circuit ii) if the PUF answers remain constant (or the answer varies with certain limits) during the system on chip operation. In the first case, even if the PUF answers are not the same in both situations, this will not affect the security primitive based on PUF. It may be said that the changes contribute to the uniqueness of the identifier. In the second case if the answer varies with certain limits, with the help of an error correcting code, the security primitive based on PUF could be used – the reproducibility property is not affected. Three cases are considered.

In the first case, a simple digital design is considered, consisting of a few binary counters. The values of the PUF answers may be analyzed in three scenarios: i) collecting the PUF answers when only PUF circuits are instantiating on FPGA hardware resources; ii) collecting the PUF answers when PUF circuits are placed and routed near counters; iii) placing the PUFs and the counters as in case b, except that the counters are inactive.

In the second case, a complex design written in VHDL (a controller for a TFT touch screen) provided by Digilent Inc is considered. The PUF instances are placed near the TFT controller, as can be seen in Fig. 6. The IP cores which are content in the TFT controller were placed as can be seen in Fig. 6, using the PlanAhead software tool. The PUF answers are tested with and without the TFT controller, keeping the same hardware configuration logic blocks for PUF circuits.

The third scenario considers a different approach based on a software implementation running on a build around the Microblaze microprocessor that runs a simple C program which copies the values from the switch buttons to the LEDs. The PUF circuits are placed as shown in Fig. 7. The PUF answers are tested with and without the SoC, maintaining the same locations for PUF circuits.

The hardware equipment used for the experimental results contains an Atlys board with Spartan 6 FPGA and an LCD Touch Screen. The software implementation tools used are: Xilinx 12.1 with ISE Design Tools, FPGA Editor, Planahead for manual placing and routing and EDK Development Kit.

## IV. Results and Analysis

The results obtained through the methodology presented in Section III are analyzed in order to conclude if the PUF circuits may be used in the newly introduced security methods.

### A. *Results of PUF Design Method using manual placement and routing on FPGA resources*

Implementation of the ring oscillator PUF occupies entirely one configurable logic block with 4 and respectively 2 Slices on Spartan 3E and Spartan 6 devices. Two other binary counters and a crossing clock domain circuit are necessary in order to compute one ring oscillator PUF answer. This implementation uses 13 bits width binary counters in order to obtain an accurate response.

Implementation of the latch based PUF occupies partially three configurable logic blocks on both Spartan 3E and Spartan 6. Many other configurations were tried but this one with 3 CLBs was chosen due to its closely related interconnections, as can be seen in Fig. 3 and Fig. 4. One 13 width binary counter to 15 latch based PUF circuits is needed in order to generate the oscillating and capture signals.

Considering this, it can be said that the latch based PUF implementation is more compact and less area expensive then the Ring Oscillator PUF implementation. The results illustrated in Fig. 1, Fig. 2, Fig. 3 and Fig. 4, indicate that the hard macro in Spartan 6 has much more delay compared to Spartan 3E implementation. There are two factors which cause this. The first one relates to the differences in FPGA Slice architectures: in the case of Spartan 3E, the Slices have 2 LUTs with 4 inputs each whereas in the case of Spartan 6, the Slices have 4 LUTs with 6 inputs each. The second factor relates to the requirement of balanced timing (equal propagation times) for interconnections leading to different routing and placement of logic gates inside the hard macro, as can be seen from Fig. 3 and Fig. 4. In order to obtain minimum differences between the propagation times inside the hard macro, the delays are larger for Spartan 6, although the Spartan 6 FPGA families are faster. The differences inside PUF hard macros, obtained for different FPGA families, emphasize the randomness or uniqueness of PUF circuits in different FPGA families.

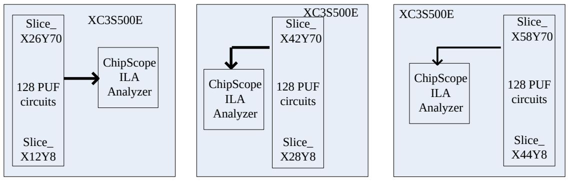


Fig. 5a. Scenario 1 Ring Oscillator on Spartan 3E.

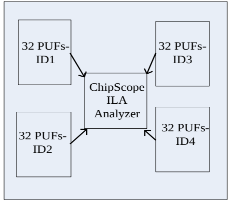
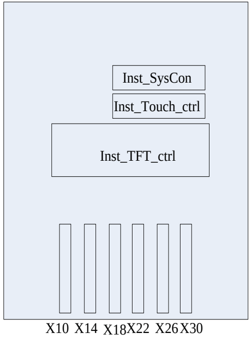


Fig. 5b. Scenario 2 Ring Oscillator / Latch PUF on Spartan 3E.



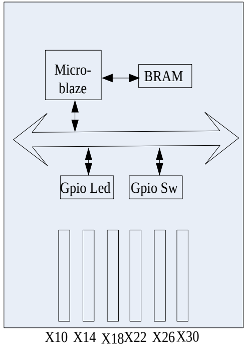


Fig. 6. Case 2: Complex IP core instance + PUF instances.

Fig. 7. Case 3: Embedded System + PUF instances.

### *B. Results of Standalone PUF ID design on FPGA device*

Considering the sequences of unique identifiers based on PUFs, it may be said that two important properties are: uniqueness and reproducibility. One possible measure of uniqueness, which is provided in most experimental results, is the inter-Hamming distance histogram, summarized by its average value. The reproducibility property is clear from its description. The responses to different evaluations of the same challenge x on the same PUF should be closed in the considered distance metric. For experimental results, this is mostly measured by the intra- Hamming distance histogram and summarized by its average value. From the statistical point of view, the FPGA population, including all the Spartan 3E XC3S500 devices, is defined. Computing parameters about FPGA population are desired but there is no access to all devices around the world. So a sample of the population is considered: 30 FPGA available devices.

Fig. 8 presents the inter- Hamming distance histogram for case III.B.1 using 128 ring oscillator PUFs. Fig. 9 shows the inter- Hamming distance histogram for case III.B.1 using 77 latch based PUF. Fig. 10 and Fig. 11 illustrate the inter-Hamming distance histogram for case III.B.2 using 4 of the 32 length unique identifiers generated using ring oscillator PUF respectively latch based PUF. All figures show that the inter-Hamming distance histogram has a normal distribution. Considering the definition of Central Limit Theorem, the mean and the standard deviation of all samples from the same population will be approximately equal to the mean and standard deviation of the population. Fig. 8, 9, 10 and 11 show the inter-chip Hamming distance measured between two different ID sequences in both cases A and B. The inter-Hamming distance must be approx. 50% of different bits of their total number. Fig. 12 and 13 show the Hamming distance between two identification sequences generated under the same conditions. The intra-Hamming distance must be about 10% unreliable bits from their total number.

The inter-Hamming distance results show that the FPGA could be identified using these sequences. Moreover, Fig. 10 and Fig. 11 show that there may be distinct IDs on the same FPGA meaning that there may be IDs or IP cores for each domain. This will allow it to implement the security protocol where IP cores are divided into different security level domains and each domain will receive a different ID.

The intra- Hamming distance results show that there are a few unreliable bits (unstable bits) from one running to another and they may be detected and corrected using error-correcting codes, like BCH. The experiments in conditions of temperature variation show that there are maximum 8 unreliable bits in case of 128 Ring Oscillator PUFs and maximum 5 unreliable bits in case of 77 Latch based PUFs.

In case of integrated circuits, ASICs or FPGAs, the process of aging affects the same physical parameters as process variations: negative bias temperature instability, hot carrier injection, oxide breakdown, electro migration. Moreover, the aging effects are similar with temperature and voltage variations. The problem with the irreversible aging process of integrated circuits is that it may affect the reliability of the PUF responses. The practical experiments carried out, which are presented in this paper, show that the aging may not be seen as a reliability problem due to following reasons: i) The implementation of the ring oscillator PUF was performed at the end of 2012, wheares the implementation of the Latch PUF was carried out at the beginning of 2014. Since then, the PUFs were measured and analyzed several times on different scenarios and the reliability was the same – maximum 8 % of unreliable bits from the total number of PUF responses that was considered; ii) The FPGAs used in those experiments are Spartan 3E and Spartan 6. The Spartan 3E boards were purchased in 2004 and since then they were used frequently in student laboratories or home work projects. Despite the fact that these boards have been used intensively for more than 10 years, the results of PUF properties are satisfactory; iii) based on the idea that aging is employed in constructing a PUF [22], the analysis in this paper shows that the aging process may increase the randomness and implicitly also the uniqueness of PUF responses. In case of a large number of unreliable bits, the PUF based identification scheme may be regenerated. Moreover, for the application introduced in the present paper, a large value of unreliability will increase the security of the mechanism. The encryption between peripherals may be based on a pseudorandom generator started with a PUF based binary sequence.

According to the experimental results, the validation of the use of the ring oscillators PUF or latch based PUF to generate a unique identifier for FPGA devices has succeed.

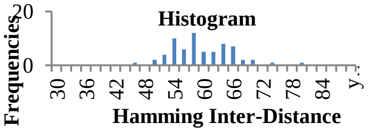


Fig. 8. Inter-Distance Histogram 128 Ring Oscillator: $\mu = 58{,}58$, $\delta = 5{,}92$.

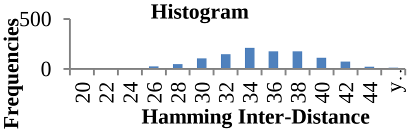


Fig. 9. Inter-Distance Histogram 77 Latch based $\mu = 34{,}73$, $\delta = 4{,}37$.

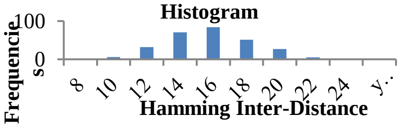


Fig. 10. Inter-Distance Histogram 4 id Ring Oscillator PUF:$\mu = 15{,}21$,$\delta = 3{,}14$.

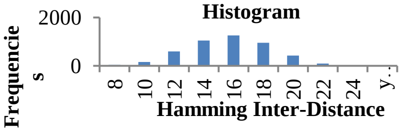


Fig. 11. Inter-Distance Histogram 4 id Latch PUF: $\mu = 15{,}01$, $\delta = 2{,}85$.

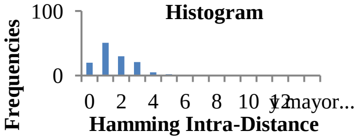


Fig. 12. Intra –Distance Histogram 128 Ring Oscillator PUF.

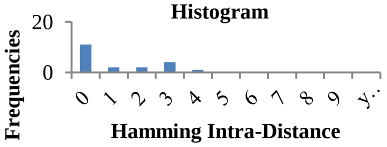


Fig. 13. Intra-Distance Histogram 77 Latch Based PUF.

### *C. Results of PUF ID design in combination with RTL complex system or System on Chip on FPGA devices*

The results regarding how the circuit activity influences the two PUF properties are presented considering the three scenarios described in Section III.C. In the first scenario, only 24 PUF instances are taken into consideration, based on Ring Oscillator. For the three scenarios, the results are satisfactory: in both cases there are related values for PUF answers and there are a maximum of 4 unreliable bits (unstable bits), which can be corrected using an error correction code. When the PUF outputs are measured without count instances, the entire design is re-synthesized, which may explain the different position of unreliable bits between the three cases. Another 24 PUF circuits are instantiated near the previous one. Also, in this case, all situations mentioned above are analyzed. The results are related with the previous ones; the 4-5 different bits could be corrected using error correcting codes.

Secondly the ring oscillator PUF instances are replaced with latch-based PUF instances and also increase the number of latch-based PUF instances. The locations and the first scenario presented in Section III.C are maintained.

The results are presented in Fig. 14 including the following 7 cases: i) PUF instances without binary counter instances, ii) PUF instances without binary counter instances after re-synthesis iii) PUF instances without binary counter instances after another re-synthesis iv) PUF instances with a binary counter instance on each row v) PUF instances with three binary counter instances on each row vi) PUF instances with more binary counter instances vii) PUF instances with maximum number of binary sequences which fit the FPGA hardware resources.

In the second scenario, the design was synthesized for two times and the ring oscillator PUF instances were placed and routed identical with the ones used in the first scenario. The PUF outputs with and without the complex VHDL design are measured. The first remark is that, even if the ring oscillator PUF instances have the same placement and routing, the ID values are different in the two scenarios.

This may be explained considering the following assumptions: i) the PUF instances are collected using the logic analyzer ChipScope Pro Analyzer and that could influence the fan out of the digital gate contained in the PUF instances; ii) the counters and the comparators used to compute the PUF instances are placed differently by the Xilinx synthesis tool in the two scenarios and this influences the fan out; iii) the circuit activity influences the PUF responses through temperature variations. To limit the effects of the fan out on the gate propagation delays, a flip-flop circuit is placed on the ring oscillator output and the flip-flop output is connected to the logic analyzer or to the counters and comparators. Considering these, it can be said that the digital circuit activity influences the PUF responses in the first two scenarios.

In case of the Latch based PUFs the conclusions are the same as in the ring oscillator case. Fig. 15 presents the results regarding the unreliable bits for ring oscillator PUF and Latch based PUF with the following two cases: i) PUF instances without complex RTL design, ii) PUF instances with complex RTL design.

The third scenario considers a system design build around the Microblaze microprocessor which runs a simple C program which copies the values from the switch buttons to the leds. The results are presented in Fig. 16 considering the two cases: i) PUF instances without SoC implementation ii) PUF instances with SoC implementation.

The PUF responses have been measured since the PUF implementations were in the final version (the end of 2012 for the ring oscillator; the beginning of 2014 for the latch based PUF). Each time the measurements were performed, the reliability property had the same value; less than 8% of the bits from the total number of generated PUF responses are unreliable. Each time the measurements were performed, around 10 collected results were taken into account. During the last 2-3 years, the PUF results were carefully monitored several times and the uniqueness and reliability properties were successfully achieved every time. The final measurement lasted 3 weeks, while the PUF responses were collected and monitored. The results obtained were successful, as in the previous cases.

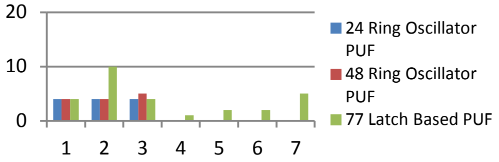
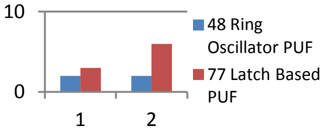


Fig. 14. Scenario 1 Number of Unreliable Bits.



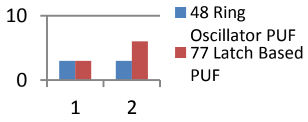


Fig. 15. Scenario 2 Number of Unreliable Bits.

Fig. 16. Scenario 3 Number of Unreliable Bits.

Theoretically, the responses of the PUF circuits have the same value, whether the instances are placed on different FPGAs or on the same FPGA but in different locations. Practically, there will be differences between the responses of PUF circuits due to process variations. These are produced by physical imperfections that may arise during the execution phase of the integrated circuits, as the manufacturing process produces involuntary deviations of physical parameters from the normal values. Changes in these parameters cause electrical parameters to vary, such as sheet resistance and threshold voltage. Section III.A contains details regarding where and how the process variations are emphasized by the PUF circuits. Changes in electrical parameters affect the switching speed of digital signals, the temperature or voltage variations produced by the circuit activity, the power supply noise, the propagation time through digital gates or interconnections introducing random jitter. The entire FPGA surface, and, as a consequence also the entire digital design implemented on FPGA are affected by process variations. As it was shown in the scenarios described and implemented within this paper, although these variations in temperature, voltage and delays are small enough not to alter the normal circuit behavior (their values are included in the worst case scenario provided by the synthesis tool), they are significant for PUF circuits.

According to the experimental results presented in Fig. 12, Fig. 13, Fig. 14, Fig. 15 and Fig. 16, there is an 8% total number of few unreliable bits from the total number of instantiated PUF circuits. For a sequence of PUF answers, which can be used as a cryptographic key for FPGA devices, it is important that the number of unreliable bits is convenient and could be corrected in order to obtain the same key on each situation. The number of accepted errors that may be corrected with an error correction and detection algorithm is maximum 10% of the identifier length [1]. Examples of error correction and detection algorithms are BCH (Bose, Ray-Chaudhuri, Hocquenghem) and Reed-Solomon codes. The mathematical theory and the hardware implementation of the error detection and correction BCH algorithm are discussed in [4]. For example, regeneration of a cryptographic key based on 128 Ring Oscillator PUFs has two stages: 1) the generation of a helper data and 2) the regeneration phase. The first phase is used only once (or each time the FPGA is powered up) for a cryptographic key. It generates the 128-bits length identifier (cryptographic key) based on Ring Oscillators. From this 128-bits length sequence, a helper data set is generated, which will be used each time for regeneration. The second phase involves the reconstruction of the identification sequence.

In the case of FPGA devices, it is not mandatory to store the helper data in non-volatile memory. Such data may be stored in LUT registers or BRAM memories available on the programmable device. Erasing the stored data after losing the power supply voltage is not an inconvenient in case of FPGA devices. The authentication phase may be executed after each power up. There are two advantages in this case. The first one is that storing the helper data on FPGA hardware resources will increase the difficulty of a recovery attack and will affect the PUF responses (any attempt to read the value of helper data means direct contact with the physical device, which will lead to PUF responses modifications). Obtaining helper data capable of correcting maximum 10% of total number of bits will not help to obtain the PUF secret key and this is also the case when the storage is a non-volatile memory. The second advantage is related to unreliable bits and aging of integrated circuits. The execution of the authentication phase will occasionally eliminate the unpleasant effects of increasing the number of unreliable bits, in case such effects exist. Moreover, this paper introduces two weak PUFs (the number of challenges the PUF can accept is very limited) for using them in secure key generation. The machine learning attacks reported in literature are mostly convenient to strong PUFs (with a large number of challenge-response pairs, such as Arbiter PUF). No generic manipulation of helper data have been published so far [24]. The malicious modification of helper data will lead to PUF responses alterations or even chip destructions. Moreover, there may be cases in which the modification of helper data will generate a modified PUF based key that may be used forward in secure key generation, since the PUF key value has not been disclosed.

## V. CONCLUSIONS

Conceptual design and implementation details of two PUF circuits, which are reliable for FPGA security methods, are presented. The use of the ring oscillators PUF and latch based PUF to generate a unique identifier for FPGA devices are validated through implementation and the experimental results presented on Section IV. The experiments were performed on two FPGA families. Section IV shows that the on-chip temperature variations contribute to the uniqueness property and does not affect the reproducibility property.

Even if the circuit design influences the PUF answers, the number of unreliable bits is small enough to be corrected using an algorithm for error correcting codes. It has been shown that the circuit activity contributes to the uniqueness of the ID generated using PUFs: for different designs on the same FPGA chip, the ID will be slowly different due to temperature variations produced by internal activity. The security application introduced in Section II may use a pseudo random generator based on PUF cryptographic key. In this case, the unreliable bits will increase the security strength of the mechanism. Environmental variations generated by circuit activity are also distinct from chip to chip due to process variations. These environmental variations could increase the random level of the PUF answers and not their unreliability.

Moreover, it has been demonstrated that distinct IDs on the same FPGA and distinct IDs for each domain or IP cores may exists on the same device. This will allow the implementation of the security protocol where IP cores are divided into different security level domains and each domain will receive a different ID.

## REFERENCES

[1] A.R. Sadeghi, D. Naccache, *Towards Hardware Intrinsic Security,* 1st ed, ser. Information Security and Cryptography, Springer Berlin Heidelberg, Oct. 2010, ch. 1, pp. 3-37.

[2] J. Freijedo, J. Semiao, J. J. Rodriguez-Andina, F.Vargas, I. Teixeira, J.P. Teixeira, "Modeling the effect of Process, Power-Supply Voltage and Temperature Variations on the Timing Response of Nanometer Digital Circuits", *Journal of Electronic Testing*, vol. 28, pp. 421-434, May 2012.

[3] X. Wang, M. Tehranipoor, "Novel Physical Unclonable Function with Process and Environmental Variations", in *IEEE Proc. of Design, Automation & Test in Europe Conference & Exhibition,* March 2010, pp. 1065-1070.

[4] A. Stanciu, A. Craciun, "Generating an Unique Identifier for FPGA Devices", in *IEEE Proc. Of 14th International Conference on Optimization of Electrical and Electronic Equipment*, May 2014, pp. 802-808.

[5] H. Hata, S. Ichikawa, "FPGA Implementation of metastability-based true random number generator", *IEICE Trans. INF. & SYST,* vol. E95-D, no. 2, pp. 426-436, Feb. 2012.

[6] D. Yamamoto, K. Sakiyama, M. Iwamoto, K. Ohta, M. Takenaka, K. Itoh, "Variety enhancement of PUF responses using the locations of random outputting RS latches", *Journal of Cryptography Engineering*, vol. 3, no. 4, pp. 197-211, Nov. 2013.

[7] J.J. Rodriguez-Andina, M.J. Mourem, M.D. Valdesl, "Features, Design Tools, and Application Domains of FPGAs*", IEEE Trans. on Ind. Electron.*, vol. 54, no. 4, pp.1810-1823, Aug. 2007.

[8] Y.J. Huang, W.C. Lin, H.L Lim, "Efficient Implementation of RFID Mutual Authentication Protocol", *IEEE Trans. on Ind. Electron*., vol. 59, no. 12, pp. 4784-4791, Dec. 2012.

[9] G.D. Sutter, J. Deschamps, J.L. Imana, "Efficient Elliptic Curve Point Multiplication Using Digit-Serial Binary Field Operations", *IEEE Trans. on Ind. Electron*., vol. 60, no. 1, pp. 217-225, Jan. 2013.

[10] M. Mozaffari-Kermani, R. Azarderackhsh, "Efficient Fault Diagnosis Schemes for Reliable Lightweight Cryptographic ISO/IEC Standard CLEFIA Benchmarked on ASIC and FPGA", *IEEE Trans. on Ind. Electron*., vol. 60, no. 12, pp. 5925-5932, Dec. 2013.

[11] JL. Zhang, Q. Wu, Y.P. Ding, Y.Q. Lv, Q. Zhou, Z.H. Xia, "Techniques for design and implementation of an FPGA-specific physical unclonable function", *Journal of Computer Science and Technology*, vol. 31, no. 1, pp. 124-136, Jan. 2016.

[12] Jl. Zhang, G. Qu, Y.Q. Lv, Q. Zhou, "A survey on silicon PUFs and recent advances in ring oscillator PUFs", *Journal of computer science and technology*, vol. 29, no. 4, pp. 664-678, July 2014.

[13] E. Simpson, P. Schaumont, "Offline HW/SW Authentication for Reconfigurable Platforms", in *Criptographic Hardware and Embedded Systems – CHES 2006 8th International Workshop*, October 2006, pp. 311-323.

[14] P. Clarke, "Xilinx launches Spartan-6, Virtex-6 FPGAs", *EE Times Magazine* [Online], February 2009.

[15] Xilinx, Annual Report Pursuant to Section 13 or 15(D) of the securities exchange act of 1934.

[16] S. Khoshroo, "Design and Evaluation of FPGA-based hybrid Physically Unclonable Functions", Master Thesis, 2013, The University of Western Ontario.

[17] O. Sander, B. Glas, L. Braun, K. D. Muller-Glaser, J. Becker, "Exploration of Uninitialized Configuration Memory Space for Intrinsic Identification of Xilinx Virtex-5 FPGA Devices*", International Journal of Reconfigurable Computing*, Oct. 2011.

[18] A. Wild, T. Guneysu, "Enabling SRAM-PUFs on Xilinx FPGAs", in *Field Programmble Logic and Applications (FPL), 24th International Conference on*, Sept 2014, pp. 1-4.

[19] M. Majzoobi, F. Koushanfar, S. Devadas, "FPGA-based True Random Number Generation using Circuit Metastability with Adaptive Feedback Control", in *Workshop on Cryptographic Hardware and Embedded Systems CHES*, Sept. 2011, pp. 17-32.

[20] M. Majzoobi, F. Koushanfar, S. Devadas, "FPGA PUF using programmable delay lines", in *Information Forensics and Security (WIFS), IEEE International Workshop on*, Dec. 2010, pp.1-6.

[21] J. Guajardo, S. Kumar, G-J. Schrijen, P. Tuyls, "Brand and IP Protection with Physical Unclonable Functions", in *IEEE International Symposium on Circuits and Systems*, May 2008, pp.3186-3189.

[22] S. Meguerdichian, M. Potkonjak, "Device aging-based physically unclonable functions", in *Proc. 48th DAC*, Jun 2011, pp. 288-289.

[23] A.R. Sadeghi, D. Naccache, *Towards Hardware Intrinsic Security,* 1st ed, ser. Information Security and Cryptography, Springer Berlin Heidelberg, Oct. 2010, ch. 2, pp. 39-53.

[24] J. Delvaux, D. Gu, D. Schellekens, I. Verbauwhede, "Helper Data Algorithms for PUF-Based Key Generations: Overview and Analysis", *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 6, pp. 889-902, June 2015.

**Alexandra Stanciu** is a third year doctoral student active in the System Engineering research field at the "Transilvania" University of Brasov, Romania. The working title of her thesis is "PUF based security in Systems on Chip". Her research is focused on trusted systems on chip with untrusted IP cores. She received her B. Sc. in computer and information technology and her M. Sc. in electronics engineering and telecommunications, both from Transilvania University of Brasov, in 2011 and 2013 respectively.

**Marcian N. Cirstea** (M'97-SM'04) received a degree in electrical engineering in 1990 from Transilvania University of Brasov, Romania, and a Ph.D. (1996) from Nottingham Trent University, UK. He is currently Professor of Industrial Electronics and Head of the Computing and Technology Department at Anglia Ruskin University, Cambridge, UK, after having worked previously for De Montfort University, UK. His research is mainly focused on FPGA design and digital controllers for power electronics; he has published over 135 works in these areas. He is Associate Editor for IEEE Transactions on Industrial Electronics / Informatics and has chaired a range of IEEE conferences (ISIE'08, OPTIM'12, OPTIM'14, INDIN'15). He has coordinated an European FP6 renewable energy project consortium. Since 2013, Prof. Cirstea is Vice-President for Membership Activities in the IEEE Industrial Electronics Society. In January 2016, his achievements were celebrated through the award of the prestigious Doctor Honoris Causa title by Transilvania University of Brasov, Romania.

**Florin Dumitru Moldoveanu (M'01)** received the B. Sc., and Ph. D. degrees in electrical engineering from Transilvania University of Brasov, Romania, in 1975 and 1998, respectively. He is currently Professor as part of the Department of Automation and Information Technology, Faculty of Electrical Engineering and Computer Science, Transilvania University of Brasov. His main research interests focus on digital circuits, discrete event systems, sliding mode control and digital image processing.

Since 2001, Prof. Moldoveanu is member of the IEEE Industrial Electronics Society.