Implementation of a PUF circuit on a FPGA

Conference Paper · March 2011

DOI: 10.1109/NTMS.2011.5720638 · Source: IEEE Xplore

Berna Ors
Istanbul Technical University

Implementation of a PUF Circuit on a FPGA

Mehmet Soybali
Istanbul Technical University
Faculty of Electrical Electronics Engineering
Maslak, Istanbul, Turkey
Email: soybali@itu.edu.tr

Berna Ors
Istanbul Technical University
Faculty of Electrical Electronics Engineering
Maslak, Istanbul, Turkey
Email: siddika.ors@itu.edu.tr

Gokay Saldamli
Bogazici University
The School of Applied Disciplines
Bebek, Istanbul, Turkey
Email: gokay.saldamli@boun.edu.tr

Abstract—**Having a robust and tamper proof structure, recently proposed pyhsical unclonable functions (PUF) are considered as a promising instrument that would be used for secure key generation and storage, integrated circuit (IC) authentication and generating chip-unique signatures. We describe a delay-based PUF architecture suitable for RFID devices mostly suffer from low computational power and tiny sillicon area. In order to match these constraints, we design a mux and an arbiter based PUF circuit that is implemented on an Field Programmable Gate Array (FPGA) for experimental purposes. Based on our measurements which is extended to variable environmental conditions, we state the length of a reliable PUF circuit.**

I. Introduction

Mechanisms emerging the IC authentication, IP (intellectual property) protection, secure key generation and storage, random number generation are some of the real world problems that industry eagerly seeks for efficient solution. Most of the considerable proposals to these intense problems involve complicated cryptographic schemes and procedures that bring extra burden on system design. Moreover, if the target platform is a constraint environment, this burden is amplified and even the most efficient solutions become infeasible. Therefore, designers tend to use the ad-hoc methods that possibly have serious security risks. However, having a robust and tamper proof structure, recently proposed physically unclonable functions (PUF) attracted the interest of the security community, and many schemes containing PUF components have been proposed [1], [2], [3], [4], [5], [6].

The authentication problem in RFID systems has a similar nature that could find some answers in PUF realizations. As a result of their low production costs and tiny size, RFID tags are considered as the replacement technology for bar codes and other means of traditional identification tools which traditionally find many applications in manufacturing, supply chain management and inventory control.

Although, public key cryptography has the necessary primitives to solve this kind of problem in typical networks, implementing these primitives on RFID nodes seems not possible. In fact, it is a challenging task to design authentication protocols for low-cost RFID tags resisting all of the known attacks and threats and at the same time fulfill the so called RFID tag specifications implying serious cost margins. Many authentication protocols have been proposed recently. However, it is shown that majority of these proposals do not provide security [7], [8], [9], [10], [11], [12], [13], [14]. In fact, those satisfying the security measures mostly suffer from addressing the requirements to a satisfactory extend.

In this study, our goal is to investigate the use of PUF circuits in RFID systems. We design a delay-based PUF architecture consist of a mux and an arbiter that matches the low power and tiny area constraints of RFID systems. We implement our design on FPGAs for experimental purposes. Based on our measurements which is extended to variable environmental conditions, we state the length of a reliable PUF circuit.

The outline of the paper is the following. In Section II, we briefly describe the physically one-way and unclonable functions as well as the multiplexer based PUF circuits. Section IV explains our implementation details where in Section V, measurements including setup circuit length determination, temperature and voltage measurements are compared and discussed. Finally, we conclude in Section VI.

II. Mathematical background

A. Physically one-way functions

One-way functions are a mainstay of public-key cryptography [15]. The concept of one-way function, shows itself in a very practical context. Think of a computer system that contains multiple user accounts. Whenever an account is built, the user chooses a password and it's raw form is saved to the system's password file. Passwords are asked to the users in every entry attempt and received passwords are compared with registered ones. In such a situation the security of the user verification depends on the security of password files. Needham recognizes that the authentication process could be made without really knowing the passwords. According to his system whenever a user enters the password PW, system automatically calculates $f(PW)$ function and saves this value instead of the password. When a user wants to login, the user should enter the password X and the computer compares $f(X)$ and $f(PW)$. If two values are equivalent, users could enter the system. Whenever, the function f is chosen as a one-way function even if f and $f(PW)$ are known, it is difficult to calculate the password.

In other words, one-way functions are easy calculate but difficult to reverse. In fact, it is possible to have multiple

The authors would like to thank to the anonymous reviewers for their helpful comments in improving this presented material. This study is partially funded by TUBITAK research project No: 109E180.

978-1-4244-8704-2/11/$26.00 ©2011 IEEE

one-way functions. For example the usual modular function is such type of function since infitely many input may give the same result under modular operations. Though mentioned one-way functions which represents the algorithmic aspects of the one-way functions are all mathematical objects. Nevertheless, there are some physical structure that has same features of these functions. These structures are called physical one-way function. we simply characterize the function with the phrase, “Physical one-way function easy to make but difficult to copy” [16].

B. Physical unclonable function

A PUF is a function that maps the challenges a physical object to its responses [17]. It satisfies the following properties:

1. Easy to evaluate: the physical object can be evaluated in a short amount of time.
2. Hard to characterize: from a number of measurements performed in polynomial time, an attacker who no longer has the device and who only has a limited (polynomial) amount of resources can only obtain a negligible amount of knowledge about the response to a challenge that is chosen uniformly at random.

It is possible to implement a PUF circuit with many different physical systems, but in this thesis, a MUX and arbiter based PUF circuits with their own timing and delays are focused. Even with identical layout masks, the variations in the manufacturing process cause significant delay differences among different ICs [15]. Silicon PUFs derive digital secrets from the complex delay characteristics of the wires and transistors in integrated circuits (ICs). Since silicon PUFs tap into the random variation that occurs during an the IC fabrication process, the secrets are intrinsic to the silicon itself. Therefore, it is extremely difficult to predict or ”program” these structures in advance.

C. Multiplexer based PUF circuits

Figure 1 shows a multiplexer and arbiter based PUF circuit. The multiplexers having the same vertical alignment have the same selective inputs. The bit sequence connected to these selective inputs are called as the challenge. Notice that the challenge length is 64-bits in Figure 1.

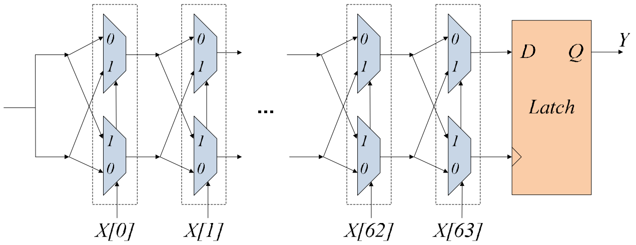

Fig. 1. PUF Circuit.

Whenever power is on the rising edge signal is separated into two signal and these signals race on the two paths crossed according to value of the challenge. At the final step, rising edge signals reach the positive edge triggered D-type flip-flop. The delay between these two signals determine the output of the circuit. If rising edge first reach the data input of the D-type flip-flop, the output will be logic1. Otherwise, the output will be logic0. The bit length of the challenge is implementation specific, but the response is only a single bit. For larger length of responses, the circuit should be run desired times.

III. Previous Work

In their work, Lee *et. al* and Lim *et. al* introduce PUF candidates and analyze their designs security and reliability in [18], [5]. They report that for a given 64-bit challenge, it takes 50 ns for an input rising edge to transmit across the 64-stage parameterized delay circuit and evaluate an output at an arbiter. The test chip was built in TSMC’s 0.18-μm, single-poly, six-level metal process with standard cells. The total area of the chip is 1212 μm$\times$1212 μm. The maximum allowable frequency is 100 MHz, and each arbiter-based PUF circuit consumes 137 μW.

Suh and Devadas [19] propose the use of ring oscillators in PUF design and reported that these are quite suitable for FPGA platforms.

Ozturk *et. al* implement a tristate PUF and compared it against an implementation of a switch-based PUF built using multiplexers [20]. The I/O for both circuits consisted of 64-bit challenge inputs, a single bit pulse input and a response output. Both designs were developed into Verilog modules and synthesized using the Synopsys tools Design Compiler and Power Compiler with the TSMC 0.13 μ m ASIC library. Simulation results show that the proposed PUF circuit consumes less power (23% at 100 MHz, 18% at 10 MHz) and requires 23% less area compared to the MUX based version.

TABLE I
Above works’ power and area consumption.

	Total Power(μW) 10MHz	Total Power(μW) 100MHz	Area (Gates)
Tristate PUF	18.78	152.93	351
Mux PUF	23.14	193.67	450

Lin *et. al* and Holcomb *et. al* demonstrate the feasibility of 45nm sub-threshold arbiter PUF for low-power applications, where speed is not a critical design consideration [21]. Derived from the previous analyses of 16-stage PUF circuits,a complete 64-stage PUF is designed. It contains 418 NAND logic gate equivalents, which can be laid out in a 36μm by 50μm die area. The power consumption of 45nm sub-threshold arbiter PUF is 0.047 μW at 1 MHz.

Anderson proposed the first FPGA-specific PUF design - one that takes advantage of the FPGA logic and routing architecture [22]. Measured results on 65 nm Virtex-5 FPGAs demonstrate the PUF signature uniqueness and its reliability at high temperature.

In their work, Morozov *et. al* have analyzed how the peculiarities of FPGA routing affect the implementations of delay based PUFs [23]. Understanding how a particular PUF architecture maps into FPGA fabric allows to select a promising architecture for further investigation and characterization of PUF circuits in FPGAs.

IV. Implementation of the PUF circuit

A. *Positive Edge Triggered D-Type Flip-Flop*

In Xilinx FPGAs, all synchronous elements controlled by the clock net which is not good for our purposes. To avoid this situation, we implement a positive edge triggered D-type flip-flop by using Look Up Tables (LUT). To implement this flip-flop, six LUTs are converted to NAND gate by entering suitable values to the truth tables of the LUTs.

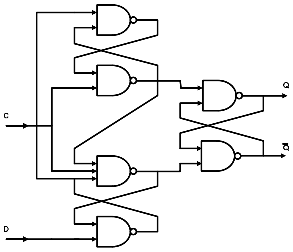

Fig. 2. Positive Edge Triggered D-Type Flip-Flop.

B. *Relative Location*

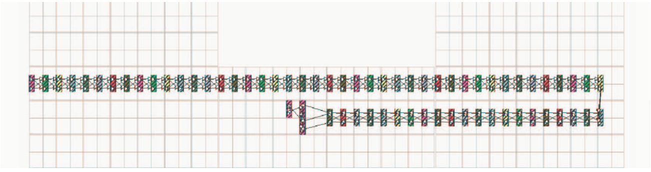

Fig. 3. Layout Of The PUF Circuit (64–bit).

Note that PUF circuit must be symmetrical. Otherwise, the delay difference between the two paths increases. In fact, this situation makes it easier to predict the output of the circuit.

In a normal situation, Xilinx ISE places the circuit components according to its place and routing algorithm. However, this is not desirable for our purposes since If a new module is added to the project, all the placement is changed. In order to get a symmetrical placement, we decided to use a relative locationing, hence, we control the placement of the circuit components in the FPGA.

Figure 3 shows the shape and location of the PUF circuit. In this situation, the PUF circuit's location is fixed and other module's locations are decided by Xilinx ISE. Moreover, if we add a new module to the project, the PUF circuit's location, shape or symmetry would not be changed.

V. Measurements on PUF Circuit

A. *Observation of the PUF circuit's operation*

For observation of the PUF circuit's operation, the circuit embeded to the Spartan 3E kit and tested by the equipment on the kit. The following lists the circuit's input and output values and their links to the Spartan 3E device.

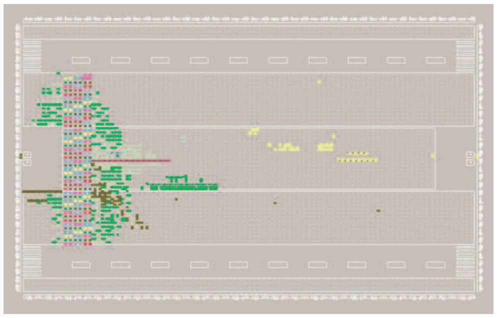

Fig. 4. Layout Of The All Circuit (128–bit).

1. Challenge Input⟶ Slide Switches.
2. Clock Input⟶ On-Board 50 MHz Oscillator.
3. Rising Edge Input⟶ Rotary Push-Button Switch.
4. Output⟶ Discrete LEDs.

In the test section, the challange was set by slide switches. The rising edge signal was given the circuit by rotary push-button switch. For different challenges, the PUF circuit gave the different outputs to the led. We saw that the output did not change until another rising edge signal. Thus, we conclude that the PUF circuit embeded in the FPGA works successfully.

B. *Measurement Setup*

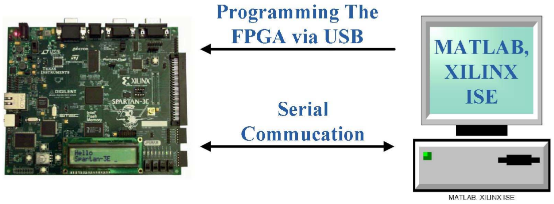

Fig. 5. Measurement Setup.

To make some measuremnets on the circuit, different challenges are given to the circuit and responses are saved. Since there are so many challanges and responses, manual testing the circuit is very difficult. To overcome this situation, we decided to use MATLAB since MATLAB can both send and receive data via serial communication.

To setup a serial communication between MATLAB and Spartan 3E kit, UART module was added to the main circuit and a serial object was created in MATLAB.

C. *Determination of the circuit length*

Note that several parameters plays role on determinig the length of the PUF circuit. Naturally, we need the smallest circuit length for low power and small area consumption. On the other hand, the big circuit length increases the complexity. Thus, the ciruit becomes more secure.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1j} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2j} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ a_{i1} & a_{i2} & \dots & a_{ij} & \dots & a_{in} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mj} & \dots & a_{mn} \end{bmatrix} \tag{1}$$

In Eq. (1), a challange matrix is shown. Each line is a n-bit challange. Whenever the circuit is runned with any of the challange line, the output is an m-bit response.

- m-bit response $\Rightarrow 2^m$ responses can be produced
- n-bit challange $\Rightarrow 2^n$ challenges can be produced
- $m \times n$ challenge matrix $\Rightarrow 2^{m\times n}$
- $2^{m\times n}/2^m \Rightarrow 2^{(n-1)\times m}$ challenges corresponding to the one response

As many challenges have the same response, when we try to invert the PUF, we face so many alternatives. In fact, this matches with the description ”A one-way function is a function which is easy to compute but hard to invert” [16] given earlier.

In ideal condition is that all the responses have the same number ($2^{(n-1)\times m}$) of corresponding challenges and the number of 0 and 1 are equal in the response space. In other words, all the response securities are equal, though the PUF circuits does not guarantee this condition. Therefore, in some cases, the number of 1s may be the majority of the response space or vice versa. Obviously, this shows the differences in security levels of the responses. In fact, if these differences are too big, some responses should not be used. Thus, in our experiments, we started to search for the circuit length which gives the best balanced bit frequecy output as one of our contributions. We tested 8, 16, 32, 64 and 128 bits respectively. Figure II shows that 64-bit circuit length gives the most balanced design in terms of the responses.

For improving PUF reliability, averaging and redundancy techniques can be applied. For example, Suh and Devadas suggested BCH error correction codes to correct bit flips in PUF signatures [19]. After this compensation, the size of a challenge vector space become smaller and the total area$-$power consumptions are increased. When RFID systems are considered, it would be reasonable to select challenge length from the table.

Therefore, in the following sections, we use 64-bit PUF circuit for measurements and tests.

TABLE II
DETERMINATION OF THE CIRCUIT LENGTH.

Bit number	Response number	Logic-1 Number	%
8	256	201	78.5
16	1000	776	77.6
32	1000	739	73.9
64	1000	654	65.4
128	1000	701	70.1

D. Temperature measurements

As the PUF circuits is used for many applications, circuit's sensivity to temperature is very important. Nobody wants reading different results from the PUF circuit with varying temperatures. In order to test the tempature sensivity, we changed the FPGA's core temperature and took responses for the same challange matrix. We tested the circuit at 18, 24, 40 and 60 degrees. Fortunately, most of the bits did not tend to change with varying tempature, accept we observe around 2% change in bit values at 60 degree. We believe this is a negligible percentage that can easily be handled using some error correction methods.

E. Voltage measurements

The sensivity to voltage is also important for reliability. We naturally expect that responses should not be changed by various voltage values. However, we had hard times in analyzing the results of our measurements. According to Spartan 3E data sheet, the voltage supplied to all the internal logic is VCCINT and its value is 1.2V having absolute maximum ratings -0.5V and 1.32V.

In our experiments, we decrement the VCCINT from starting voltage 1.25V to 1.05V by 0.05V steps. Although at the voltage 1.25V, there were no change in response bits, we saw up to 74% change at 1.15V but this large difference is reduced down to 13% at 1.05V. Due to these measurements, we agreed on leaving the analysis of voltage sensitivity to future research.

F. Different FPGA measurements

We measure the responses of the same design on different FPGA kits. In fact, as we modify all the characteristic of the PUF circuit, the new design components and their delays creates an entirely different circuit. Naturally, this new circuits responses have to be different for the same challenges.

Our test results from different Spartan 3E measurements supported this observation; the responses are changed according to new characteristics. After that, we took some measurements by changing the location of the circuit in the same FPGA. Again, all the responses were different. Therefore, we observe that these differences are due to the variations occurring during manufacturing process as expected.

G. Speed, Power and Area Consumption

We tested the delay based arbiter PUF circut in the Xilinx XC3S500E FPGA and at the 50 MHz. It takes $65.79ns$ to evaluate an output at an arbiter for a given 64-bit challange.

Table III shows a comparison between our arbiter PUF circuit and the previous works. However, this comparison is not that healthy since designs other than ours are ASIC design and the working frequencies are not the same. Despite the unhealthy comparison conditions, the table gives an idea about the arbiter PUF designs on FPGAs.

When all the measurements are considered, it is seen that the PUF circuit embedded in the Spartan 3E FPGA kit, is robust against the temperature variations. And the different results taken from different FPGAs or different FPGA locations,

TABLE III
COMPARISON OF THE ARBITER PUFS.

	Total Power	Delay	Area	Challange Length
Lee [18]	137μW (100 MHz)	50ns	1212 $\times$1212μm	64
Tristate PUF [20]	153μW (100 MHz)	-	351 Gates	64
Lin [5]	0.047μW (100 MHz)	-	301212 $\times$50μm	64
Our work	83mW (50 MHz)	66ns	509 Slices (491 LUTs)	64

increase the reliability. Because the different circuits can distinguish different ICs, RFID tags or FPGAs and they are not effected by climates. But the sensivity to temperature shows that some response bits are effected more than others. These response bits have very little delay differences between two paths. Even the slightest changes can effect their results. May be, these challange-response pairs are detected and grouped according to their security levels. Each group can be used according to security levels required by the applications.

VI. CONCLUSIONS

In this study, we implement, route and test an arbiter based PUF circuit on FPGAs. It is important to note that with their extremely simple circuit structures, arbiter based PUFs are quite suitable for PUF applications. Their low power characteristics, small area deployments and high speed data paths make these modules a good candidates for RFID systems.

When all of our measurements are considered, it is seen that the arbiter based PUF circuits embedded in the Spartan 3E FPGA kit, are robust against the temperature variations. The sensitivity to temperature shows that some response bits are effected more than others. These response bits have very little delay differences between two paths. Even the slightest changes can effect their results. We consider that these challenge-response pairs could be detected and grouped according to their security levels and each group can later be used with respect to the security levels required by the applications. Future work will involve PUF applications for RFID systems.

REFERENCES

[1] L. Bolotnyy and G. Robins, “Physically Unclonable Function-Based Security and Privacy in RFID Systems,” in *Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications (PERCOM)*, 2007, pp. 211–220.

[2] B. Gassend, “Physical random functions,” Master’s thesis, Massachusetts Institute of Technology, 2003.

[3] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, “Silicon physical random functions,” in *in Proceedings of Computer Communications Security Conf.*, 2002, pp. 148–160.

[4] D. Lim, “Extracting secret keys from integrated circuits,” Master’s thesis, Massachusetts Institute of Technology, 2004.

[5] D. Lim, J. W. Lee, B. Gassend, G. Suh, M. van Dijk, and S. Devadas, “Extracting secret keys from integrated circuits,” *IEEE Transactions on VLSI Systems*, vol. 13, no. 10, pp. 1200–1205, October 2005.

[6] R. Pappu, B. Recht, J. Taylor, and N. Gershen-Feld, “Physical one-way functions,” *Science*, vol. 297, pp. 2026–2030, 2002.

[7] H. Chien and C. Chen, “Mutual Authentication Protocol for RFID Conforming to EPC Class 1 Generation 2 standards,” *Computer Standards & Interfaces, Elsevier Science Publishers*, vol. 29, no. 2, pp. 254–259, February 2007.

[8] M. Ohkubo, K. Suzuki, and S. Kinoshita, “Cryptographic Approach to “Privacy-Friendly” Tags,” in *RFID Privacy Workshop*, MIT, Massachusetts, USA, November 2003.

[9] K. Rhee, J. Kwak, S. Kim, and D. Won, “Challenge-Response based RFID Authentication Protocol for Distributed Database Environment,” in *International Conference on Security in Pervasive Computing – SPC 2005*, ser. Lecture Notes in Computer Science, D. Hutter and M. Ullmann, Eds., vol. 3450. Boppard, Germany: Springer-Verlag, April 2005, pp. 70–84.

[10] D. Nguyen Duc, J. Park, H. Lee, and K. Kim, “Enhancing Security of EPCglobal Gen-2 RFID Tag against Traceability and Cloning,” in *Symposium on Cryptography and Information Security*, Hiroshima, Japan, January 2006.

[11] B. Song and C. J. Mitchell, “RFID Authentication Protocol for Low-cost Tags,” in *ACM Conference on Wireless Network Security, WiSec’08*, V. D. Gligor, J. Hubaux, and R. Poovendran, Eds. Alexandria, Virginia, USA: ACM Press, April 2008, pp. 140–147.

[12] T. Dimitriou, “A Lightweight RFID Protocol to protect against Traceability and Cloning attacks,” in *Conference on Security and Privacy for Emerging Areas in Communication Networks – SecureComm*. Athens, Greece: IEEE, September 2005.

[13] D. Henrici and P. Müller, “Hash-Based Enhancement of Location Privacy for Radio-Frequency Identification Devices Using Varying Identifiers,” in *International Workshop on Pervasive Computing and Communication Security – PerSec 2004*, R. Sandhu and R. Thomas, Eds., IEEE. Orlando, Florida, USA: IEEE Computer Society, March 2004, pp. 149–153.

[14] D. Molnar and D. Wagner, “Privacy and Security in Library RFID: Issues, Practices, and Architectures,” in *Conference on Computer and Communications Security – ACM CCS*, B. Pfitzmann and P. Liu, Eds., ACM. Washington, DC, USA: ACM Press, October 2004, pp. 210–219.

[15] S. Devadas, E. Suh, S. Paral, R. Sowell, T. Ziola, and V. Khandelwal, “Design and implementation of puf-based “unclonable” RFID ICs for anti-counterfeiting and security applications,” in *Proceedings of the IEEE International Conference on RFID*, April 2008, pp. 58–64.

[16] P. S. Ravikanth, “Physical one-way functions,” Ph.D. dissertation, Massachusetts Institute of Technology, 2001.

[17] P. Tuyls and L. Batina, “Rfid-tags for anti-counterfeiting,” in *Topics in Cryptology - CT-RSA 2006, volume 3860 of LNCS*. Springer Verlag, 2006, pp. 115–131.

[18] J. Lee, L. Daihyun, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, “A technique to build a secret key in integrated circuits for identification and authentication applications,” in *Proceedings of the Symposium on VLSI Circuits*, 17-19 June 2004.

[19] G. E. Suh and S. Devadas, “Physical unclonable functions for device authentication and secret key generation,” in *Proceedings of the Design Automation Conference (DAC)*, San Diego, California, USA, June 4-8 2007.

[20] E. Ozturk, G. Hammouri, and B. Sunar, “Physical unclonable function with tristate buffers,” in *Proceedings of the International Symposium on Circuits and Systems (ISCAS)*. Seattle, Washington, USA: IEEE, 18-21 May 2008, pp. 3194–3197.

[21] L. Lin, D. Holcomb, D. K. Krishnappa, P. Shabadi, and W. Burleson, “Design optimization and security validation of sub-threshold PUFs,” in *Proceedings of the Secure Component and System Identification Workshop (SECSI)*, Cologne, Germany, 2010.

[22] J. Anderson, “A puf design for secure fpga-based embedded systems,” in *Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific*, jan. 2010, pp. 1 –6.

[23] S. Morozov, A. Maiti, and P. Schaumont, “An analysis of delay based puf implementations on fpga,” in *Reconfigurable Computing: Architectures, Tools and pplications*, ser. Lecture Notes in Computer Science, P. Sirisuk, F. Morgan, T. El-Ghazawi, and H. Amano, Eds. Springer Berlin / Heidelberg, 2010, vol. 5992, pp. 382–387.