A ring oscillator based PUF proposal on FPGA

Filip Kodýtek
1st class, full-time study
Supervisor: Róbert Lórencz

CTU in Prague, Faculty of Information Technology
Thákurova 9, 16000, Prague, Czech Republic
kodytfil@fit.cvut.cz

***Abstract*—This contribution deals with design of Physical Unclonable Function (PUF) on FPGA. The goal was to propose a cheap, efficient and secure device identification or even a cryptographic key generation based on PUFs. Therefore, a design of a ring oscillator (RO) based PUF producing more output bits from each RO pair is presented. The design was tested on Digilent Basys 2 FPGA boards (Xilinx Spartan3E-100 CP132) and statistically evaluated. We also discuss its properties and analyse the proposed PUF at varying temperature and voltage. Based on the results of the experiments, we propose suitable modifications of the PUF design in order to improve the quality of its output.**

***Keywords*—Hardware security, physical unclonable function, field-programmable gate array, ring oscillator**

I. Introduction

Field-programmable gate arrays (FPGAs) are used to implement digital circuits of various functionality. Just like other implementation platforms, FPGAs require security and resilience to attacks [9]. For many security protocols, a secret key needs to be stored on FPGA. However, safe storages of keys are usually complicated and expensive to achieve and the nonvolatile memory, in which the keys can be stored, tends to be vulnerable to invasive attacks, because the key is stored in a digital form.

Physical Unclonable Functions (PUFs) offer a solution to this issue. Rather than to store the secret keys in memory, they can be generated using PUFs when they are needed. PUF is a function based on physical properties, which are unique for each device. These unique physical properties can be used to distinguish various devices from each other. Therefore, PUFs can be used for identification purposes and for cryptographic key generation.

Two major groups of PUFs, which are suitable for FPGAs according to their sources of randomness are delay-based and memory-based PUFs. A very common PUF design is based on SRAM and uses it as a source of randomness, since many electronic devices have embedded SRAM [2]. This PUF is based on the content of SRAM after power-up. However some FPGAs initialize their memory after power-up, so all randomness is lost. That led to proposals of other memory-based PUFs such as Butterfly PUF [6], Latch PUF [10] and Flip-flop PUF [8].

Delay-based PUFs exploit the random variations in delays of logic gates and interconnects. One of the first delay-based PUFs is Arbiter PUF [7]. Another examples are Ring Oscillator PUF (ROPUF) [1], [11] and Glitch PUF [12].

In this work we present a ring oscillator based PUF suitable for FPGAs which showed good results in terms of good statistical properties, simplicity and efficiency. One of the advantages of the proposed PUF design is the fact that it is easy to implement, area efficient, and additionally it does not require all ROs to be mutually symmetric, in contrast with the classical approach [1] where all ROs are mutually symmetric and the PUF output is derived from the comparison of RO frequencies of various RO pairs. However, as it is shown in experimental results in Chapter IV, when the symmetric ROs are used in our design, it enhances stability of the proposed PUF design when the physical conditions are varying.

This paper is organized as follows. Section II provides a brief description of the ROPUF, that was proposed in [4]. The performance metrics that are used to evaluate the PUF are described in Section III. Section IV presents the results of experiments. The last Section V concludes the paper.

II. The proposed ROPUF

In this section we give a brief description of the proposed ROPUF architecture. More detailed description of the main concept of this ROPUF is provided in our previous work [3], [4]. In the first part of this section, we explain the main principle of this ROPUF. Then, the proposed ROPUF circuit is described and ultimately, some modifications of the ROPUF design are presented.

A. The main principle of the proposed ROPUF

The main motivation of this proposal was the simplicity of implementation and more efficient use of ROs. In the classical approach [11], the frequencies of ROs are compared and the result of this comparison produces one output bit for PUF. In order to achieve unpredictability of the PUF outputs, this approach requires all ROs to be mutually symmetric so that the differences in frequencies of ROs are influenced only by the random variations in delays of logic gates and interconnects. As also mentioned in [11], the number of pairs of ROs for this comparison is limited, so that the bits in the PUF outputs are independent.

In our ROPUF proposal, a different technique than frequency comparison is used to generate PUF output. The PUF outputs are still obtained based on the selected RO pairs,

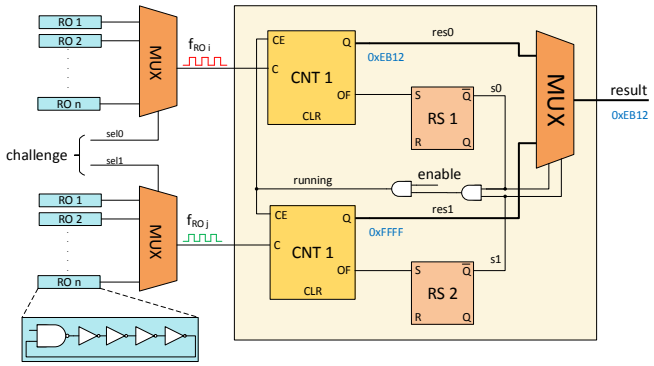

Fig. 1: The proposed ROPUF circuit.

but the problem of selecting particular RO pairs is no longer present. In addition, more bits for the PUF output are gained from each pair of ROs and this technique also does not require all ROs to be mutually symmetric. This allows us to produce longer PUF outputs using less ROs.

The basic building element of the proposed ROPUF is an ordinary five stage RO composed of 1 NAND gate and 4 inverters. Instead of measuring frequency of each RO using some reference clock, we choose one pair of ROs and count the oscillations of each RO simultaneously using two counters. As soon as one of these counters overflows, the measurement is stopped and the resulting value in the counter that did not overflow is used for further processing. This approach is shown in Fig. 1. There are two sets of ring oscillators and they are all enabled and running during the measurement. The overflow detection logic is realised by two RS Flip-flops. When implementing the logic for detecting overflow of one of the counters and stopping the other one, the routes between them may have different delays and before the second counter is stopped, it can perform some additional steps. But since these two routes are the same for all RO pairs and for all FPGAs, it will only increase the resulting value by some constant offset which is 0 or 1 in this case.

The proposed method implies that if we knew the exact frequencies of the ROs during measurement, we could determine the resulting counter value (in case of 16-bit counters) that is later processed as follows:

$$Counter\ value = \frac{f_2}{f_1} \times 2^{16}, \tag{1}$$

where f_1 is the frequency of the faster RO and f_2 is the frequency of the slower RO.

Since the obtained counter values are represented in binary code, we can use the appropriately selected part of each binary number for the PUF output. It can be assumed, that if we repeat the measurements for one RO pair, the bits that are close to the least significant bit (LSB) will vary a lot due to instability of ROs and the environmental changes. On the contrary, the bits close to the most significant bit (MSB) will be stable and the environmental changes will have almost no influence on them. The more we will be close to the MSB, the more stable the bits will be.

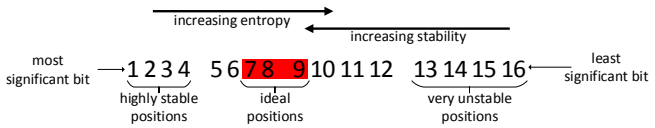

Fig. 2: The example behaviour of the bits in counter value of a 16-bit value.

Another requirement in addition to stability that needs to be met is the entropy of the selected bits. We may assume, that if we compare the measured values from two equally positioned RO pairs on two FPGAs, bits close to the MSB will not differ, while the bits that are approximately in the middle between the MSB and the LSB will be different. The bits close to the LSB will be different too, but it is caused mainly by their instability. Therefore, the ideal positions of the counter value that should be used for PUF are in the middle of the counter value. The example of described behaviour of measured counter values is shown on 16-bit counter value in Fig. 2.

B. Modifications of the proposed ROPUF

In order to eliminate some of the issues present in the original design, we proposed some modifications of the design that enhance the properties of the PUF.

The first improvement of our PUF design is the application of Gray code on the obtained counter values. One of the issues of selecting a block of bits from each counter value is that all of the selected bits may change in the next measurement since they are represented in binary code and they are a part of a counter value. So even if the final value is increased or decreased just by one, all of the bits can be influenced by this change. The first step to solving this issue involves the application of Gray code to the obtained counter values. The reason for using Gray code is the fact that two successive values differ in only one bit, so this can eliminate the partial overflow and increase the overall stability of the selected bits and even increase the number of extractable bits from each counter value. For more details see [3], [5].

The next improvement is related to the issue of the influence of various physical conditions on the behaviour of the PUF. Since the PUF design is based on ROs, the physical conditions will have a significant impact on the frequencies of ROs, however, our goal is to make the differential measurement (frequency ratio) more robust against such effects. Therefore, we present a possible solution to this problem using symmetric ROs. More detailed description of both of the modifications is provided in [3], [5].

III. Performance Metrics

To evaluate the quality of PUF, we need some metrics in order to evaluate its statistical properties. In our previous work [4], we discussed how to select good positions of the counter values for the PUF based on their statistical properties, such as stability and entropy. After selecting the suitable positions, the PUF outputs made of these positions need to be evaluated by some additional parameters to validate the selection of positions. In this section we describe the

TABLE I: The results of statistics carried out for responses composed from various bit selections for 150 RO pairs.

150 pairs of ring oscillators

positions	6–8	7–8	7–9	7–10	8–9
w [-]	3	2	3	4	2
HD_{intra} [%]	1.44	2.16	2.81	4.21	4.21
HD_{inter} [%]	43.05	48.81	49.23	49.45	49.88

evaluation method, which we used to determine the qualities of the PUF outputs. We review two common parameters that are used to evaluate the properties of PUFs, namely *Intra-Hamming distance* (HD_{intra}) and *Inter-Hamming distance* (HD_{inter}).

A. Intra-Hamming distance

To evaluate the mutual similarity of the PUF outputs, we use Intra-Hamming distance as a metric. HD_{intra} is estimated as:

$$HD_{intra} = \frac{1}{m \times k} \sum_{i=1}^{m} \sum_{j=1}^{k} HD(R_{r_i}, R_{i,j}) \times 100 \ [\%], \quad (2)$$

where m is the number of FPGAs, R_{r_i} is the reference output of the i-th PUF, which the other outputs are compared to, and k is the number of compared outputs from each PUF. As R_{r_i}, we can use either any output from the given PUF or the mean output of several outputs (this may result in lower HD_{intra}). There are several influences that affect the value of HD_{intra} such as changes in voltage or temperature which cause HD_{intra} to be of higher value.

B. Inter-Hamming distance

Another important metric that is used to evaluate the PUF quality is the uniqueness of the generated outputs among different FPGAs. We can determine the uniqueness of the generated outputs by calculating the Inter-Hamming distance, which is defined as:

$$HD_{inter} = \frac{1}{\binom{m}{2}} \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} HD(R_{r_i}, R_{r_j}) \times 100 \ [\%], \quad (3)$$

where m is the number of FPGAs and R_{r_i} is the reference output of the i-th PUF which is the mean output made of all outputs from the given PUF.

IV. Experimental results

In this section we present the results of performed measurements on Digilent Basys 2 FPGA boards (Xilinx Spartan3E-100 CP132). At first, the results for PUF with symmetric ROs that uses different selections of positions from counter values with applied Gray code are presented.

A. Evaluation of ROPUF with symmetric ROs

Table I presents the results of statistical evaluation of the PUF outputs for symmetric ROs measured 1000 times for 150 RO pairs and with Gray code applied to the selected parts of the counter values. The results indicate, that the proposed ROPUF with symmetric ROs still works correctly and enables us to reliably distinguish different FPGAs. However, the results of HD_{intra} for symmetric ROs are slightly worse than for asymmetric ROs (Table 2 in [4]).

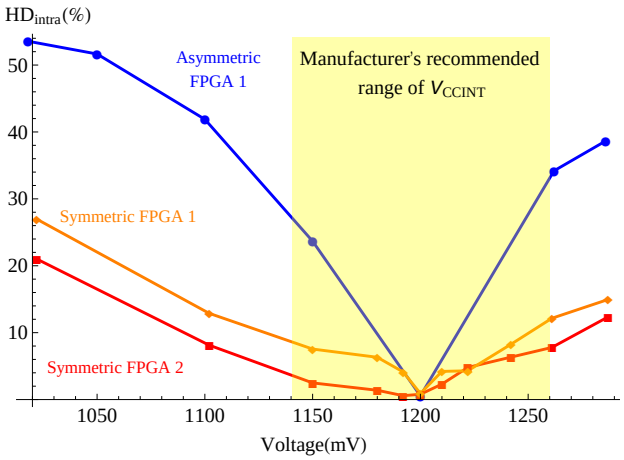

Fig. 3: Comparison of the behaviour of the proposed PUF when using mutually symmetric and asymmetric ROs for positions 7–8. The reference output for calculating HD_{intra} is the mean output from the PUF outputs measured at nominal voltage 1.2V. Yellow area represents the manufacturer's recommended range of FPGA's main power supply voltage V_{ccint}, which is from 1.14V to 1.26V.

B. Influence of supply voltage

The next measurement concerns the influence of voltage on the behaviour of the proposed ROPUF design. The measurements were performed on 2 Digilent Basys 2 FPGA boards containing Xilinx Spartan3E-100 CP132. The main power supply for the FPGA's internal logic is V_{ccint} and its nominal voltage is 1.2V. The maximum ratings for V_{ccint} are -0.5V and 1.32V, with manufacturer's recommended range from 1.14V to 1.26V. The circuit remains the same and the results presented in Fig. 3 relate to 1000 measurements for 150 RO pairs and show how the PUF outputs are different from nominal voltage, which is 1.2V. The range of tested voltages is from 1.018V to 1.286V and the selected positions of counter values for the PUF outputs are 7–8.

It can be seen in Fig. 3 that the influence of voltage is significant in case of asymmetric ROs. This is caused by the change of ratios of two frequencies of ROs in each pair. If we want the PUF outputs to remain stable with varying voltage, the ratios of the frequencies for each pair have to be the same at any voltage level. We can expect that the frequencies of ROs will change in a similar way when the ROs are mutually symmetric. Therefore, we placed the RO gates so that all ROs are mutually symmetric.

Fig. 3 presents the comparison of the behaviour of the proposed ROPUF when using mutually symmetric and asymmetric ROs for positions 7–8. The results for HD_{intra} are not ideal, but they demonstrate the improvement when using symmetric ROs compared to asymmetric ROs and show the

TABLE II: Evaluation of HD_{intra} for 150 asymmetric/symmetric RO pairs and selected positions 7–8 and different temperatures.

Asymmetric ROs					
FPGA 1		FPGA 2		FPGA 3	
Temperature [°C]	HD_{intra} [%]	Temperature [°C]	HD_{intra} [%]	Temperature [°C]	HD_{intra} [%]
36.7 → 41.2	2.67	38.4 → 42.3	2.67	37.7 → 41.8	1.0
36.7 → 51.8	7.67	38.4 → 50.1	6.67	37.7 → 50.9	5.0
36.7 → 60.4	9.33	38.4 → 60.3	9.33	37.7 → 61.3	7.0
36.7 → 71.1	11.33	38.4 → 69.9	12.67	37.7 → 70.1	12.0
Symmetric ROs					
FPGA 1		FPGA 2		FPGA 3	
Temperature [°C]	HD_{intra} [%]	Temperature [°C]	HD_{intra} [%]	Temperature [°C]	HD_{intra} [%]
33.0 → 42.4	2.67	34.4 → 40.9	1.67	34.5 → 41.1	3.67
33.0 → 50.5	3.67	34.4 → 50.5	3.0	34.5 → 51.4	6.0
33.0 → 60.6	3.67	34.4 → 60.8	4.67	34.5 → 60.6	7.0
33.0 → 71.0	4.67	34.4 → 70.2	5.33	34.5 → 70.4	7.33

way for further investigation of the influence of the placement of ROs on the stability of the PUF outputs.

C. Influence of temperature

This subsection examines the influence of change of temperature on the proposed ROPUF. The statistical properties of PUF using both symmetric and asymmetric ROs will be compared. For the purpose of our experiment, we performed measurements at different temperatures. For these measurements, FPGA was preheated to a preset temperature (e.g. 40 °C) with ROs enabled. Each of the measurements was carried out when the temperature measured on the package of the FPGA stabilized at the given value. We used 3 Digilent Basys 2 FPGA boards for this experiment.

Table II displays the values of HD_{intra} for 3 FPGAs for asymmetric and symmetric ROs. The column temperature presents the temperatures, at which the PUF outputs are compared. The values of HD_{intra} for symmetric and asymmetric ROs are almost equivalent for small differences in temperature, but for larger changes of temperature, there is a visible improvement of the PUF behaviour, when symmetric ROs are used.

V. Conclusion

We proposed a RO based PUF, which is able to provide more output bits from each pair of ROs and is also not dependent on the symmetry of ROs, implying that it is easy to implement. However, as it was shown in Section IV, the proposed PUF exhibits better behaviour at varying physical conditions in terms of stability when symmetric ROs are used.

In our future work, we would like to build a statistical model of ring oscillators and eventually model the whole ROPUF. This would be useful to evaluate the PUF and also a true random number generator, which can be based on the same circuit (also future work). Then we would like to examine the influence of aging on this ROPUF and further investigate the influence of supply voltage and temperature together with the placement of ROs. Our goal is to modify the PUF design, so that it will be resilient to environmental changes as much as possible. This is one of the conditions in order to generate cryptographic keys of sufficient length using the proposed ROPUF.

References

[1] Bossuet, L., Ngo, X. T., Cherif, Z., Fischer, V. A PUF based on a transient effect ring oscillator and insensitive to locking phenomenon. In *IEEE Transactions on Emerging Topics in Computing*, pages 30–36, March 2014.

[2] Holcomb, D. E., Burleson, W. P., Fu, K. Power-Up SRAM State as an Identifying Fingerprint and Source of True Random Numbers. *IEEE Transactions on Computers 58, 9*, pages 1198-1210, 2009.

[3] Kodýtek, F. *Behaviour Analysis and Improvement of the Proposed PUF on FPGA*. Master's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2016.

[4] Kodýtek, F., Lórencz, R. A design of ring oscillator based PUF on FPGA. In *IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems - DDECS 2015*. Belgrade, RS, April 2015.

[5] Kodýtek, F., Lórencz, R., Buček, J. Improved ring oscillator PUF on FPGA and its properties. In *Microprocessors and Microsystems*. 2016.

[6] Kumar, S., Guajardo, J., Maes, R., Schrijen, G.-J., Tuyls, P. Extended abstract: The Butterfly PUF Protecting IP on Every FPGA. In *IEEE International Symposium on Hardware-Oriented Security and Trust - HOST 2008*, pages 67–70. IEEE, Washington, DC, USA, 2008.

[7] Lee, J. W., Lim, D., Gassend, B., Suh, G.E., van Dijk, M., Devadas, S. A technique to build a secret key in integrated circuits for identification and authentication applications. In *Symposium on VLSI Circuits - VLSIC 2004*, pages 176–179, June 2004.

[8] Maes, R., Tuyls, P., Verbauwhede, I. Intrinsic PUFs from Flip-flops on Reconfigurable Devices. In *Benelux Workshop on Information and System Security - WISSec 2008*. Eindhoven, NL, 2008.

[9] Majzoobi, M., Koushanfar, F., Devadas, S. FPGA PUF using Programmable Delay Lines. In *Information Forensics and Security*. December 2010.

[10] Su, Y., Holleman, J., Otis, B. A 1.6pJ/bit 96% Stable Chip-ID Generating Circuit using Process Variations. In *IEEE International Solid-State Circuits Conference - ISSCC 2007*, pages 406–611. IEEE, February 2007.

[11] Suh, G. E., Devadas, S. Physical Unclonable Functions for Device Authentication and Secret Key Generation. In *Design Automation Conference - DAC 2007*, pages 9–14. ACM, New York, NY, USA, 2007.

[12] Suzuki, D., Shimizu, K. The Glitch PUF: A New Delay-PUF Architecture Exploiting Glitch Shapes. In Workshop on Cryptographic Hardware and Embedded Systems - CHES 2010, pages 366–382. Springer, 2010.