An Efficient Reconfigurable Ring Oscillator for Hardware Trojan Detection

Filippos Pirpilidis
Computer Informatics and Engineering Department
TEI of Western Greece
Antirrion, Greece

Artemios G. Voyiatzis
SBA Research
Favoritenstraße 16
Vienna, Austria

Lambros Pyrgas
Computer Informatics and Engineering Department
TEI of Western Greece
Antirrion, Greece

Paris Kitsos
Computer Informatics and Engineering Department
TEI of Western Greece
Antirrion, Greece
pkitsos@teimes.gr

ABSTRACT

The threat of inserting malicious logic in hardware design is increasing as the digital supply chains are becoming more deep and span the whole globe. Ring oscillators (ROs) can be used to detect deviations of circuit operations due to changes of its layout caused by the insertion of a hardware Trojan horse (Trojan).

The placement and the length of the ring oscillator are two important parameters that define an RO sensitivity and capability to detect malicious alternations.

We propose and study the use of ring oscillators with variable lengths, configurable at the runtime. Such oscillators push further the envelope for the attackers, as their design must be undetectable by *all* supported lengths. We study the efficiency of our proposal on defending against a family of hardware Trojans against an implementation of the AES cryptographic algorithm on an FPGA.

CCS Concepts

•Security and privacy → Malicious design modifications; •Hardware → Reconfigurable logic and FPGAs; Bug detection, localization and diagnosis; Hardware test; *On-chip sensors;*

Keywords

Ring Oscillator; Hardware Trojan Horse; FPGA; AES; cryptographic algorithm

1. INTRODUCTION

A hardware Trojan Horse (HTH or simply "Trojan") is an extra circuit in a legitimate integrated circuit (IC) aiming to leak information to an adversary or even downgrade the performance of the IC at a specific time during the operation [17, 15, 2].

Novel methodologies and workflows were proposed to reduce the risk of Trojan infection in first place and detect the presence of a hardware Trojan on an IC [8, 3]. After more than a decade of active research on the topic, efficient means for hardware Trojan horse detection remains an open question [18].

Invasive or destructive Trojan detection techniques can be very successful in identifying and locating the presence of a Trojan [11]. However, they can be used as a last resort, since the inspected chip is destroyed. Also, the presence in one chip does not disclose the number of infected chips, unless they are destroyed as well. Hence, it is necessary to devise detection techniques that can detect the presence of a Trojan based on deviations in the logic (e.g., erroneous output) and physical domains (e.g., increased path delay or power consumption).

Hardware sensors embedded in the circuit design can be a helpful means towards reducing the testing time and focusing the test efforts on specific regions of a circuit that are more sensitive to modifications and prone to error (e.g., the area used to implement a cryptographic algorithm). Integrating hardware sensors allows for a more fine-grained signal collection, allowing to detect deviations that would go unnoticed otherwise. For example, the effect of adding a few more gates on the overall power consumption of a circuit is negligible and often beyond the accuracy of the measurement instruments. Yet, it has been shown that only a handful of gates suffice to silently introduce hardware faults in AES computations and derive the cryptographic key used [1].

A "*ring oscillator*" (RO) comprises an odd number of NOT gates in a ring, whose output oscillates between two voltage levels, representing true and false. The NOT gates, or inverters, are attached in a chain and the output of the last inverter is fed back into the first. The period of the oscillator is sensitive to environmental conditions (e.g., temperature) as well as input voltage. Thanks to their sensitivity and implementation simplicity, they are often used

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

PCI '16, November 10-12, 2016, Patras, Greece

© 2016 ACM. ISBN 978-1-4503-4789-1/16/11…$15.00

DOI: http://dx.doi.org/10.1145/3003733.3003808

to realize embedded hardware sensors, especially for FPGA designs [5].

Ring oscillators are proposed in the literature as a means for Trojan detection [19, 12]. The assumption is that if they are carefully placed and have an appropriate length, then their frequency will change beyond the process variation once additional circuitry (i.e., a Trojan) is introduced in a design. However, the exact placement of one or more ring oscillators can be the decisive factor with respect to its capability to detect a Trojan insertion as well as the ability for an attacker to insert a Trojan in first place [13].

In this paper, we turn our focus on the selection of an appropriate ring oscillator length. We propose a RO design of a variable-length that can be configured dynamically at the runtime. We then study its performance in detecting Trojans against an implementation of a cryptographic algorithm and draw conclusions on its efficiency.

The rest of the paper is organized as follows. Section 2 provides some background information on ring oscillators and hardware Trojan horse detection. Section 3 introduces a ring oscillator design with configurable length. Section 4 describes the experimental setup for testing the efficiency of the proposed design, while Section 5 presents and discusses the results of our experiments. Finally, Section 6 concludes the paper and outlines future directions of work.

2. BACKGROUND INFORMATION

Efficient non-invasive testing of both the functional (logic) and physical characteristics of integrated circuits is much needed for detecting hardware Trojan horses, as the search space is exponential to the IC inputs and beyond contemporary processing capabilities [4, 16]. For example, it would take 2^{128} to test all possible AES keys for detecting one that might be activating a Trojan in its implementation.

The concept of ring oscillators for hardware Trojan detection is widely studied in the literature [18]. We intrdoduced the problem of selecting an appropriate RO length in [7]. There, we showed that different lengths can detect different kind of Trojans, hence, there is no "golden" length. Rather, the IC manufacturer must carefully experiment and asess the threats in their environment for integrating a RO sensor with the most appropriate length.

We also proposed TERO, an alternative implementation of a RO based on the transient effect [10]. TERO is more sensitive than a classical RO. However, our study indicated that it is also more unreliable and unstable, producing many false positives [9, 6]. Again, TEROs with different lengths are able to detect some Trojans, while other Trojans go totally unnoticed.

The aforementioned situation creates an asymmetrical advantage for an attacker. The manufacturer must select one length for their RO (classical or TERO) and stick with it. On the other hand, the attacker is assumed to have access to the design so as to insert the Trojan. Thus, the attacker has ample time to study the deployed ROs and experiment with different Trojans designs until the ROs cannot detect them. In the next section, we propose an approach to reverse this status by requiring the attacker to spend extended time experimenting and still have no guarantee that the deployed attack will go undetected during legitimate testing.

3. RING OSCILLATOR DESIGN

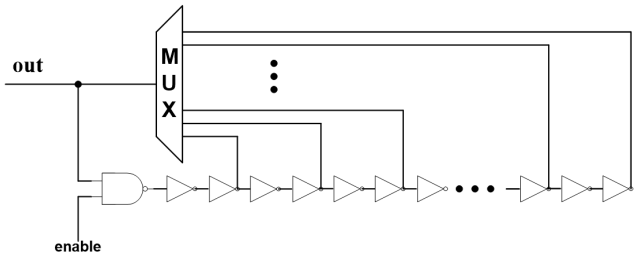

Figure 1: Ring oscillator with configurable length.

In the previous works the ring oscillator (RO) has a fixed length. We propose and describe here a design of a ring oscillator with a configurable-at-the-runtime length. A general block of the proposed design is depicted in Figure 1. It comprises an even number of NOT gates and a NAND gate for enabling (activating) it.

The inputs of the multiplexer (MUX) control the number of the NOT gates taken into account for the operation of the RO. The distance between two successive MUX input is two gates, hence, the total number of NOT gates is an even number. The MUX output is fed back to the RO and is also used as the final output of the RO.

The oscillation frequency changes based on the input provided to the MUX, since a different number of gates is used to form the ring. The `enable` signal controls the operation of the RO: when the signal equals to "1", the circuit oscillates; when it equals to "0", the feedback path is broken and there is no oscillation. This design decision provides the capability to dynamically switch the RO length at the runtime. This is useful when testing a circuit so as to adjust the RO and achieve maximum sensitivity without restarting the testing procedure.

The flexibility of dynamically adjusting the RO length allows to create circuit signatures based on different RO length sequences. This can potentially make attacks harder: the attacker needs to design a Trojan that remains undetected for each and every supported RO length *and* under each possible sequence of applied RO lengths.

4. EXPERIMENTAL SETUP

We opted for a Digilent Basys 3 FPGA development board for experimentation. The Basys 3 features the Xilinx Artix-7-FPGA XC7A35T-1CPG236C. It provides 33,280 logic cells organized in 5,200 slices. Each slice contains four 6-input LUTs and 8 flip-flops. The board can be interfaced with various means, including 5 user pushbuttons, a USB HID Host, and a USB-UART bridge.

We realized on this board the AES cryptographic algorithm and four instances of the configurable-length ROs at various distances from the algorithm. The resulting layout is depicted in Figure 5.

The AES implementation lies within the red frame of Figure 5. We opted for the implementation provided with the SAKURA-G development board[1]. A detailed architecture of the specific implementation is provided in [14].

The four white lines are from left to right RO1, RO2, RO3, and RO4. RO1 is close to the pushbuttons, while

[1]http://satoh.cs.uec.ac.jp/SAKURA/hardware/SAKURA-G.html

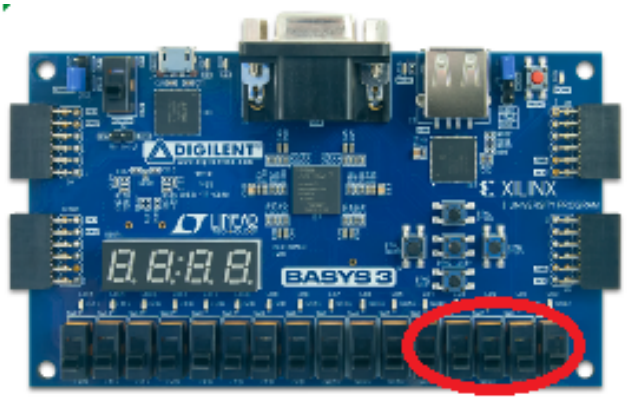

Figure 2: Basys3 FPGA development board with marked pushbuttons.

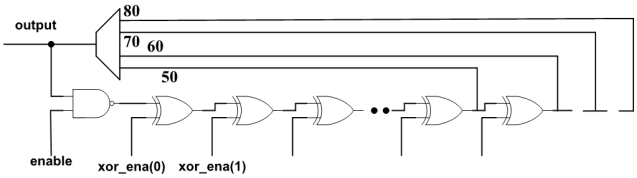

Figure 3: Variable-length ring oscillator design.

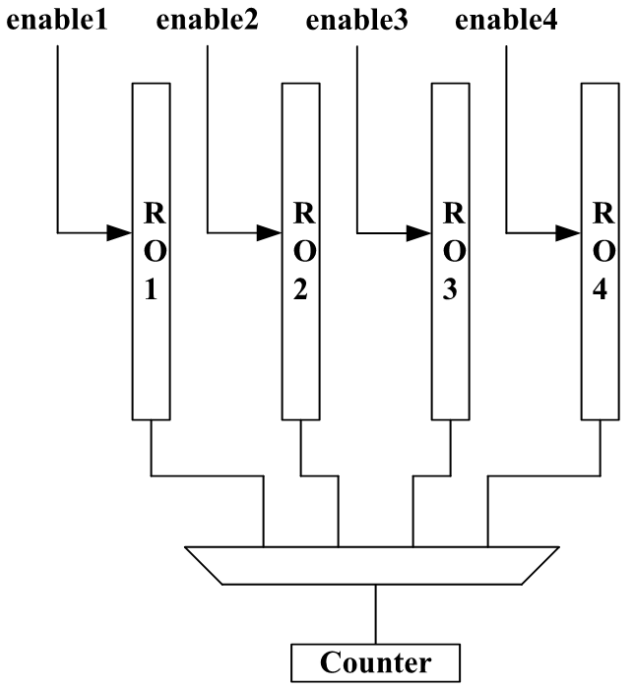

Figure 4: RO logic and counter output.

RO3 is close to the AES implementation. The ring oscillator design is depicted in Figure 3. It comprises one NAND gate for activation and a series of XOR gates for realizing the oscillation circuit. Each XOR is fed with the output of the previous one and one `enable` signal. By changing the latter on a gate by gate case, we can control the total length of the ring oscillator. Our basic design extends to a length of 80. For realizing a ring oscillator with length 70 out of it, a signal of 70 ones followed by 10 zeroes is provided. The same pattern is used for realizing any other length: a signal of n ones followed by $80 - n$ zeroes realizes an RO with length n. This is a good practice in order to reduce the noise introduced by the unused gates.

The activation of the ring oscillators is controlled by the pushbuttons of the Basys3, using a multiplexer, as depicted in Figure 4. At each time, only one ring oscillator is active and its output is fed to the counter for logging and detecting possible manipulations.

The random white dots within the red frame depict the logic used to realize the hardware Trojan. We assume that this logic is inserted at a later stage of the design. The malicious logic comprises a tree of AND gates that monitors the values of eight out of the 128 key bits (namely, positions 10, 21, 30, 40, 90, 101, 117, and 125). The Trojan payload comprises a XOR gate that inverts the mode of operation for the AES circuit (i.e., perform encryption or decryption). Thus, when the activation pattern is detected, the payload destroys the computation, effectively resulting in a Denial-of-Service (DoS) attack.

The communication between the Basys 3 board and the controlling computer takes place over the USB-UART bridge. A wrapper interface was developed to prepare the 128-bit wide AES data (plaintext, key, ciphertext) into 8-bit wide inputs and outputs. On the computer side, a graphical interface allows the tester to insert the desired test vectors and control the runs. The counts of the ROs are displayed on the Basys 3 display and recorded for further processing.

5. RESULTS AND DISCUSSION

We studied the effect of the RO placement and length by performing a series of repeated experiments. For each run, we provided the AES algorithm with ten pairs of plaintext and key (fixed for all the experiments) and repeated the execution of the algorithm 50 times. We also varied the length of the ring oscillator from 51 to 81 with a step of 10. In all cases, we used three groups of input pairs for the AES algorithm: one group (again, fixed values) contained ten pairs that *did not* activate the Trojan. The second group contained five pairs that *did* activate the Trojan and five that *did not*. Finally, the third group contained ten pairs that all *did* activate the Trojan.

5.1 Effect of length

In this set of experiments, we used the set of test patterns that *do not* activate the inserted Trojan. While the design is the same from the resource consumption perspective, the sensitivity *decreases* with the greater lengths, as depicted in Figure 6. This is valid for all the four ROs we used.

We further experimented with RO lengths up to 121. We observed that the oscillator counter dropped to nearly zero. We observed a similar behavior to the one described above even when a Trojan was inserted in the design.

5.2 Effect of placement

In this set of experiments, we used again the set of test patterns that *do not* activate the inserted Trojan. The sensitivity exhibits *mixed* behavior regarding the placement of the RO, as depicted in Figure 7. For smaller lengths, the

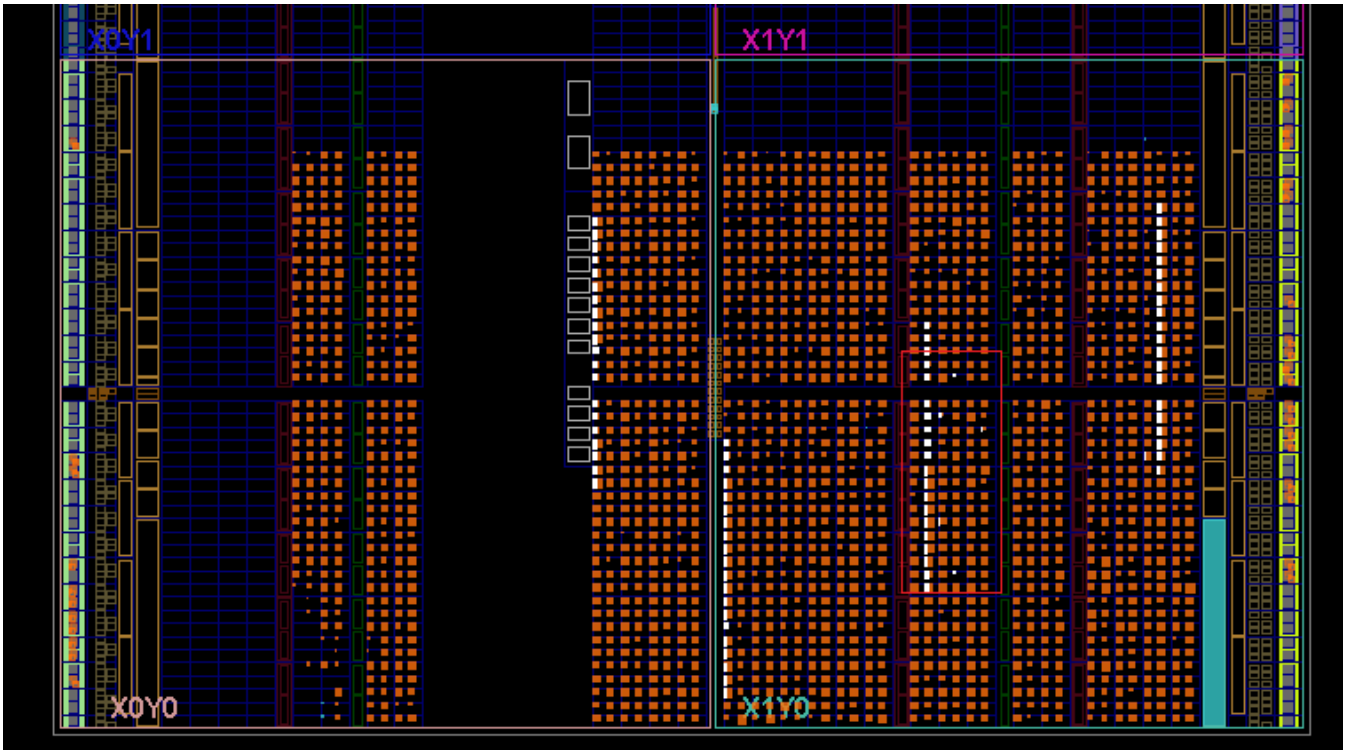

Figure 5: Layout of AES, four ring oscillators, and malicious logic (white dots in the red frame).

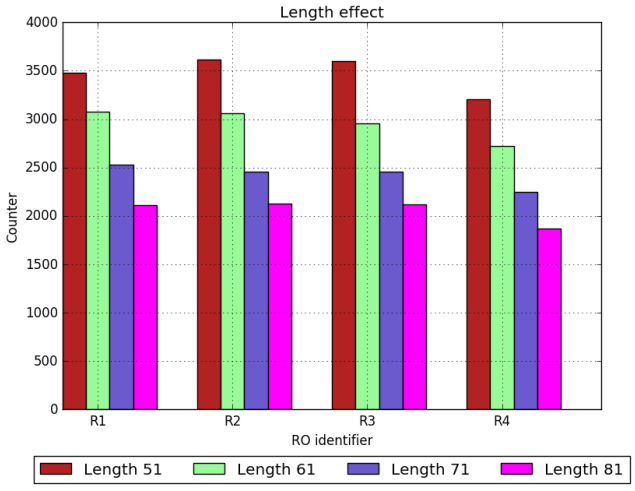

Figure 6: Effect of RO length on design with no Trojan

nearby RO2 performs slightly better than RO3 and both outperform the distant RO1 and RO4. However, for the middle lengths (i.e., 61 and 71), there is a ranking pattern and RO1 is slightly better than the other three. This is a quite interesting finding, as RO1 and RO4 are almost equally distant from the AES implementation but exhibit quite different behavior. Also, because the more distant RO1 performs better than RO3 that is nearby the resource-intensive AES implementation. Finally, for the long length of 81, the first three ROs are performing equally well, while RO4 has clearly a smaller oscillation frequency.

5.3 Effect of test pattern

In this set of experiments, we used a set of test patterns that *always* activates the inserted Trojan. We also used a design that incorporates the Trojan itself so as to see its malicious actions. As depicted in Figure 8, the test patterns themselves do not affect the sensitivity of the RO. This validates the expected behavior, i.e., a RO performance following the patterns discussed in the previous paragraphs and being independent of the provided inputs.

5.4 Trojan detection

In this set of experiments, we used a mixed set of test patterns that *both* activate and do not activate the Trojan payload. We also used two designs, a Trojan-free (golden) one and one that incorporates the Trojan itself. Figure 9 depicts the differences in the oscillation counts measured between the two designs. Figure 10 quantifies these differences as a percentage of the base counts of the golden design.

We observe that RO1 is more sensitive, although it is far away from the AES implementation and the injected Trojan. It scored consistently over 4% count differences. RO2 exhib-

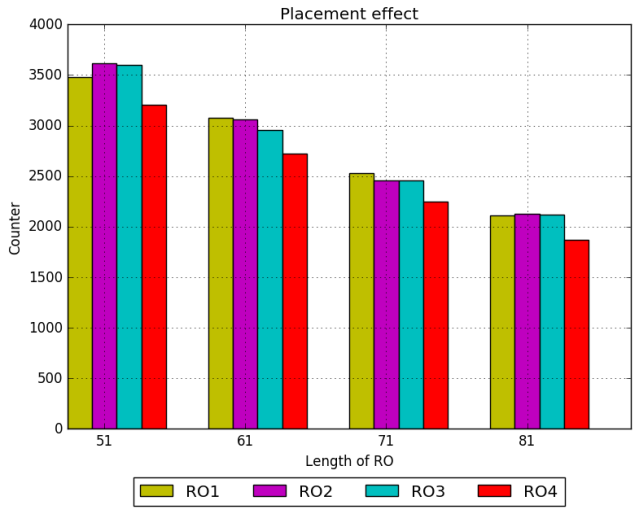

Figure 7: Effect of RO placement on design with no Trojan

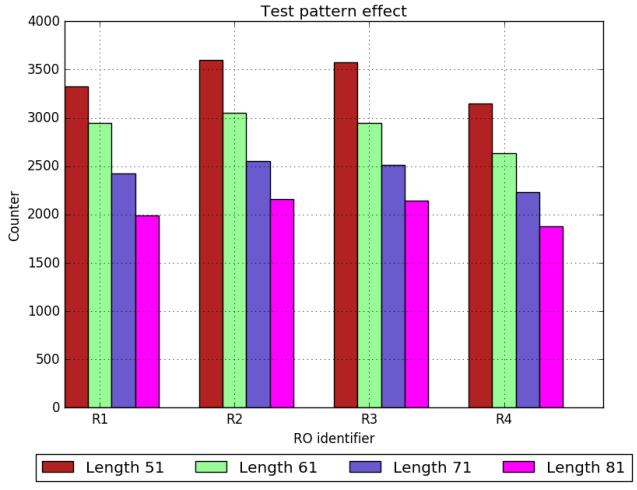

Figure 8: Effect of test patterns

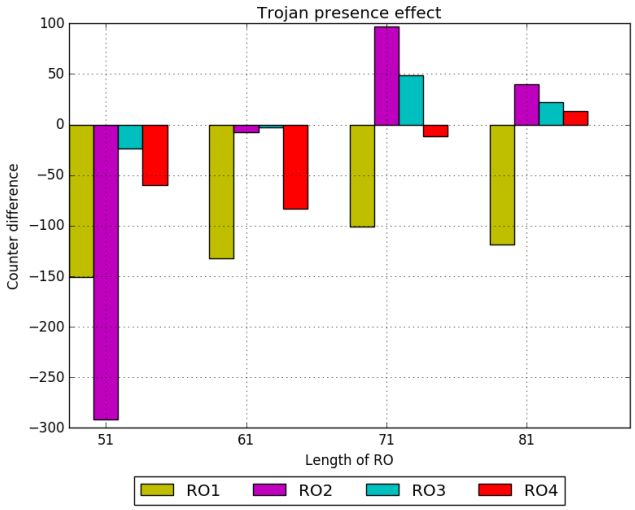

Figure 9: Effect of Trojan presence (absolute difference

ited the strongest difference of 8% on the small RO length and almost reach the 4% for a RO length of 71. It did not register any suspicious activity for the lengths of 61 and 81. It is quite interesting that RO3, despite being within the region of the AES implementation and the Trojan did not score any significant difference for any RO length. Finally, RO4 performed the worse of all in all cases.

6. CONCLUSIONS

On-board hardware sensors, such as ring oscillators, are a line of defense against malicious alternations of the hardware logic in form of Trojans. The length and the placement of a ring oscillator are important parameters that affect its detection capability against different Trojan designs.

In this paper, we proposed the use of multiple ring oscillators with length configurable at runtime. Such an approach raises the bar for an attacker as they must remain undetected for all possible lengths for any ring oscillator employed. Hence, they must invest a significant amount of resources to ensure that.

Our experimental results suggest that in the case of an FPGA, more distant ROs can be equally or more sensitive to the ones close to the attack target. This makes the work of the attackers even harder. Also, the length of the oscillator affects its sensitivity.

We experienced improved sensitivity when the RO length was smaller. After a specific length, the RO was unable to detect any changes, no matter how much the length was increased. This contrasts results published in the literature; it is a topic of future research to further validate this finding with other FPGAs and hardware designs. Also, it is interesting to study the detection capability when two or more ROs are run concurrently, with equal or mixed lengths.

7. ACKNOWLEDGMENTS

This work was supported in part by the Austrian Research Promotion Agency (FFG) under the COMET K1-Centres programme line (SBA2), while A.G. Voyiatzis was with SBA Research.

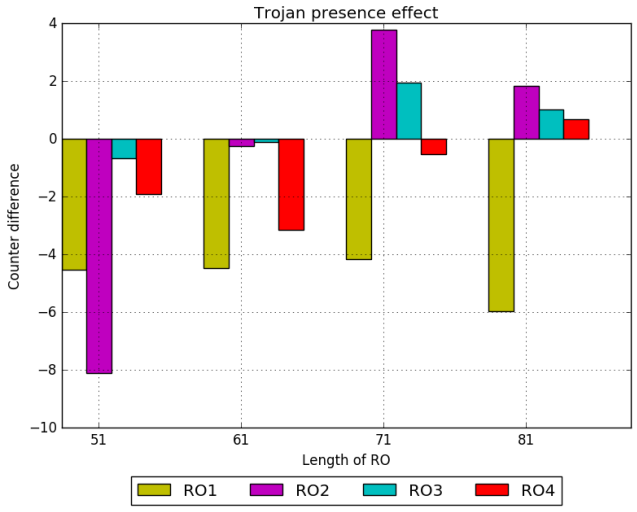

Figure 10: Effect of Trojan presence (percentage difference)

8. REFERENCES

[1] S. Bhasin, J.-L. Danger, S. Guilley, X. T. Ngo, and L. Sauvage. Hardware Trojan horses in cryptographic IP cores. In *Fault Diagnosis and Tolerance in Cryptography (FDTC), 2013 Workshop on*, pages 15–29. IEEE, 2013.

[2] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan. Hardware Trojan attacks: Threat analysis and countermeasures. *Proceedings of the IEEE*, 102(8):1229–1247, 2014.

[3] A. Dabrowski, H. Hobel, J. Ullrich, K. Krombholz, and E. Weippl. Towards a hardware Trojan detection cycle. In *Availability, Reliability and Security (ARES), 2014 Ninth International Conference on*, pages 287–294, Sept 2014.

[4] P. Kitsos, D. Simos, J. Torres-Jimenez, and A. Voyiatzis. Exciting FPGA cryptographic Trojans using combinatorial testing. In *2015 IEEE 25th International Symposium on Software Reliability Engineering (ISSRE 2015)*, pages 69–76. IEEE CPS, 2015.

[5] P. Kitsos, N. Sklavos, and A. Voyiatzis. Ring oscillators and hardware Trojan detection. In N. Sklavos, R. Chaves, G. D. Natale, and F. Regazzoni, editors, *Hardware Security and Trust: Design and Deployment of Integrated Circuits in a Threatened Environment*. Springer, 2016.

[6] P. Kitsos, K. Stefanidis, and A. Voyiatzis. TERO-based detection of hardware Trojans on FPGA implementation of the AES algorithm. In *19th EUROMICRO Conference on Digital System Design (DSD 2016)*. IEEE CPS, 2016.

[7] P. Kitsos and A. Voyiatzis. FPGA Trojan detection using length-optimized ring oscillators. In *17th EUROMICRO Conference on Digital System Design (DSD 2014)*. IEEE CPS, 2014.

[8] P. Kitsos and A. Voyiatzis. Towards a hardware Trojan detection methodology. In *2nd EUROMICRO/IEEE Workshop on Embedded and Cyber-Physical Systems, ECYPS 2014*, 2014.

[9] P. Kitsos and A. Voyiatzis. A comparison of TERO and RO timing sensitivity for hardware Trojan detection applications. In *18th EUROMICRO Conference on Digital System Design (DSD 2015)*, pages 547–550. IEEE CPS, 2015.

[10] P. Kitsos and A. G. Voyiatzis. Investigating tero for hardware trojan horse detection, 2015.

[11] J. Kumagai. Chip detectives [reverse engineering]. *IEEE Spectrum*, 37(11):43–48, 2000.

[12] J. Rajendran, O. Sinanoglu, and R. Karri. Regaining trust in VLSI design: Design-for-trust techniques. *Proceedings of the IEEE*, 102(8):1266–1282, 2014.

[13] J. Rilling, D. Graziano, J. Hitchcock, T. Meyer, X. Wang, P. Jones, and J. Zambreno. Circumventing a ring oscillator approach to FPGA-based hardware Trojan detection. In *Computer Design (ICCD), 2011 IEEE 29th International Conference on*, pages 289–292. IEEE, 2011.

[14] A. Satoh, S. Morioka, K. Takano, and S. Munetoh. A compact Rijndael hardware architecture with S-box optimization. In *Advances in CryptologyASIACRYPT 2001*, pages 239–254. Springer, 2001.

[15] M. Tehranipoor and F. Koushanfar. A survey of hardware trojan taxonomy and detection. *IEEE Design and Test of Computers*, 27(1):10–25, 2010.

[16] A. Voyiatzis, K. Stefanidis, and P. Kitsos. Efficient triggering of Trojan hardware logic. In *19th IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS 2016)*, pages 200–205. IEEE, 2016.

[17] X. Wang, M. Tehranipoor, and J. Plusquellic. Detecting malicious inclusions in secure hardware: Challenges and solutions. In *Hardware-Oriented Security and Trust, 2008. HOST 2008. IEEE International Workshop on*, pages 15–19. IEEE, 2008.

[18] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. Tehranipoor. Hardware Trojans: Lessons learned after one decade of research. *ACM Trans. Des. Autom. Electron. Syst.*, 22(1):6:1–6:23, May 2016.

[19] X. Zhang and M. Tehranipoor. RON: An on-chip ring oscillator network for hardware Trojan detection. In *2011 Design, Automation & Test in Europe*, pages 1–6. IEEE, 2011.