2014 17th Euromicro Conference on Digital System Design

FPGA Trojan Detection Using Length-optimized Ring Oscillators

Paris Kitsos
Computer and Informatics Engineering Department
Technological Educational Institute of Western Greece
and Industrial Systems Institute/RC “Athena”
e-mail: pkitsos@teimes.gr

Artemios G. Voyiatzis
Industrial Systems Institute/RC “Athena”
PSP building, Stadiou Str., Platani
GR-26504, Patras, Greece
e-mail: bogart@isi.gr

***Abstract*—Hardware Trojan horses are a realistic threat for both ASIC and FPGA systems. Ring Oscillators (ROs) can be used to detect the presence of malicious hardware functionality. The length of an RO is a significant parameter for detecting efficiently malicious logic (sensitivity) while maintaining a low space and power profile. We explore through simulation the effect of the RO length on detecting different classes of Trojan horses on an FPGA.**

***Keywords*—FPGA, security, hardware Trojan horse, ring oscillators, Trojan detection.**

I. Introduction

The globalization of the IC design and fabrication process by different actors dispersed geographically around the globe creates multiple opportunities for inserting a malicious logic to a hardware design. Such logic is commonly referred as “hardware Trojan horse” or simply “Trojan”. An attacker may be able to alter a design netlist during the design flow in such a way that s/he does not affect the original functionality of the system and then disable or destroy it at a time in the future or force the system to leak sensitive information, such as secret keys for cryptographic operations [1-2]. .

A hardware Trojan horse comprises of two parts: the trigger and the payload. The trigger is used to activate the payload once the conditions envisioned by attacker are met. Examples include timers or a specially-crafted input signal. The payload performs the malicious action designed by the attacker. The threat of hardware Trojan horse insertion is considered a realistic one and attracts the attention of both the research and industry community. The detection of a Trojan is still extremely difficult [3].

Two main classes of detection techniques relate to the physical manipulation of the system-under-test (SUT) in the case of an ASIC. Destructive techniques require the demetallization of the manufactured system and layer-by-layer image extraction using for example a Scanning Electron Microscope (SEM). Such techniques are both very expensive and time-consuming. Furthermore, they destroy the tested system and offer no guarantee that another circuit also contains a Trojan or is Trojan-free unless it is also destructed.

Non-destructive techniques include testing based on integrated circuit logic and on side channels. The circuit-logic approaches apply test vectors for activating a Trojan and detecting the effects of its malicious payload [1]. Side-channel analysis relates to monitoring of operational parameters of the circuit, such as quiescent supply current; leakage current; dynamic power trace; electromagnetic radiation (EM) due to switching activity; and path-delay characteristic [4-11]. The behaviour of the SUT is then compared with that of a reference circuit (the “golden” circuit produced in a trusted fabrication plant) for detecting deviations. Such deviations are a signal that the design of the SUT was altered at some stage of the design flow.

The FPGA technology has gained significant popularity and acceptance. Compared to the ASIC technology, it is cheaper, it is faster to deploy, and it offers re-configuration capabilities thus, it allows mapping of different circuits at the same hardware. This trend creates new challenges in the design flow, focusing on the trustworthiness of the deployed hardware [12]. Techniques and methods are needed for detecting tampering and possible Trojan insertion on an FPGA device not only during the design phase but also throughout its whole lifecycle. The need is more apparent as such devices are deployed in critical infrastructures and other mission-critical environments.

A Ring Oscillator (RO) is a circuit that oscillates due to its inherent logic. The oscillation frequency depends on the exact components and size of a circuit (its “*signature*”). Modifications of the components, such as insertion of additional gates or different placement can change this frequency [13]. This property is utilized in integrating ring oscillator(s) as to support non-destructive testing for detecting unwanted modifications of the hardware design [14-16].

In this paper, we explore the effect of the length (size) of the ring oscillators as a design tradeoff for detecting Trojans on FPGAs. A longer or more complex RO or set of ROs may have increased detection sensitivity and be able to detect different classes of Trojans. However, it may also occupy precious resources (space) on the FPGA and/or increase significantly the power consumption of the system.

The rest of the paper is structured as follows. In Section II, ring oscillators for FPGAs are discussed. In Section III, the experimental setup is presented, while Section IV analyzes the results of the experiments. Finally, Section V concludes our paper and discusses future directions of work.

This work was financially supported by GSRT Action “KRIPIS” with national and EU funds in the context of the research project “ISRTDI”.

978-1-4799-5793-4/14 $31.00 © 2014 IEEE
DOI 10.1109/DSD.2014.74

II. FPGA Ring Oscillators

A ring oscillator is realized as an odd number of inverters in order to build the oscillation loop. An FPGA ring oscillator is realized by LUTs configured as inverters. A chain of LUTs forms the oscillation loop, as depicted in Fig. 1. The amount of delay in the oscillation loop can be adjusted, as it is a “soft design”, by changing the number of LUTs (inverters), their relative placements, and interconnect used.

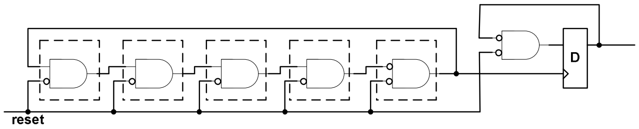

Fig. 1. Architecture of an FPGA ring oscillator

For a given circuit, the observed frequency of an RO depends on process variation during fabrication, environmental conditions (e.g., temperature), operating characteristics (e.g., voltage), and measurement noise. These factors can account for a variance up to 6.6% [17].

Ring oscillators can be utilized for detecting hardware Trojan horses. A technique to configure the circuit paths into ring oscillators in a way that the number of secured gates is maximized and the hardware overhead minimized is presented in [13-16]. A limitation of that technique is that it is very difficult to include all the circuit paths into ring oscillators. A detection methodology combining RO networks and off-chip transient power analysis is proposed in [18]. To the best of our knowledge, no previous work has explored how the length of the ring oscillator(s) affects the capability of detecting hardware Trojan horses and what is an optimal length with respect to power consumption and circuit space.

III. Experimental Setup

A cryptographic primitive is the building block for multiple security services (e.g., different modes of plaintext encryption and secure hashing). As such, it is an attractive target for a hardware Trojan horse attack. Upon activation, the Trojan can leak the key information or can disable cryptographic protection resulting into plaintext output or output with reduced security (e.g., encrypting with AES using less rounds than those recommended).

In our experiments, we use as an example the self-synchronizing stream cipher Mosquito [19]. The Mosquito cipher is designed with hardware efficiency in mind; it occupies a reasonable area and introduces very small gate delay. Mosquito uses a key of 96 bits and has a memory of 105 previous bits for self-synchronization. Fig. 2 depicts a Mosquito circuit that allows encryption and decryption with a critical path of two XOR gates.

We designed two hardware Trojan horses that are shown in Fig. 3. The first one, namely *Trojan1*, is a combinatorial Trojan formed as a tree of AND gates. The output of the tree is fed into a XOR gate that drives the signal for defining the mode of operation for Mosquito (encryption or decryption). The Trojan is activated once all key bits in eight randomly chosen positions are set to “1” (namely: 0, 12, 13, 40, 50, 51, 70, and 91). Once it is activated, the Trojan inverses the selected mode of operation for Mosquito, effectively creating a denial-of-service attack until a new key is set.

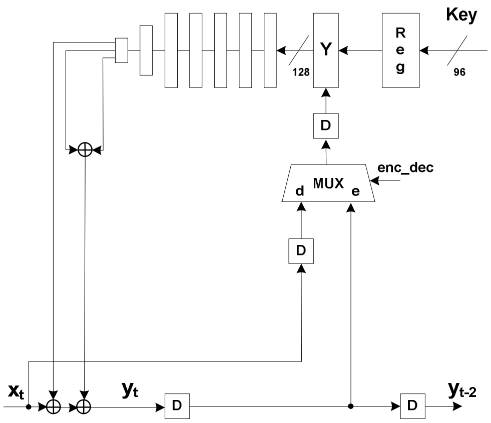

Fig. 2. Mosquito circuit for both encryption and decryption

The second one, namely *Trojan2*, is a time bomb. It consists of a synchronous free-running counter, a three-input AND gate and a XOR gate. The Trojan is activated once every 100 clock cycles. Again, when activate, the Trojan inverses the selected mode of operation for Mosquito for that cycle.

We note that only one Trojan is inserted each time. Thus, there are three designs: one that is Trojan-free (only with original Mosquito), one with Mosquito and Trojan1, and one with Mosquito and Trojan2.

When either Trojan1 or Trojan2 is activated, it changes the mode of operation for Mosquito. In this case, there is a 50% probability that the wrong signal will be fed to the internal state (“Y” in Fig. 2) of the cipher. As a self-synchronizing cipher, Mosquito will be able to recover from this error after 105 output bits (the size of its memory). With the right key, Trojan1 will be active at every cycle and thus, Mosquito will not be able to synchronize until a new key is set. Trojan2 is more stochastic in nature and may require multiple activations until it succeeds in changing the internal state of the cipher.

A ring oscillator (of variable lengths) was also designed for detecting presence of Trojans. A 32-bit counter fed by the output of the ring oscillator is used to measure the frequency of the latter.

The implementations were synthesized with XILINX ISE Webpack tool and a Spartan6 (XC6SLX25-CSG324) device was used. The Mosquito implementation achieves a frequency of 200 MHz. The Trojans are inserted either in specification level or in the design phase at the Register Transfer Level.

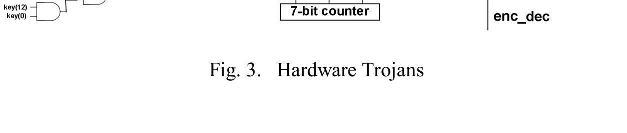

Fig. 3. Hardware Trojans

In an FPGA design flow, the insertion of any extra circuit causes the entire design to be resynthesized, replaced and rerouted [20]. Thus, increasing the RO length by two LUTs can result in significant layout changes, rendering comparisons useless. The Xilinx PlanAhead tool was utilized to overcome this obstacle. The tool allows the old (the cipher and the RO) and the new (the cipher and the RO with increased length) implementations to share alike place-and-route up to the old RO length. This property allows to collect proper measurements for comparing original (Trojan-free) and modified (with Trojans) circuits containing ROs of various lengths.

The Xilinx ISE tools can generate time models after each step of FPGA design flow. Initially, the synthesis tool is executed with information about the I/O locations and slice & LUT placement and routing. The same I/O locations are used. Also, the same component (slices and LUTs) regions are applied for all implementations. After Place-and-Route, a VHDL model of the design is created with components broken down into individual slices and LUTs. Also, delay information is added for each component. The ModelSim simulator was used in order to generate the timing results and RO counts.

IV. Experimental Results

The layouts of the three different implementations are depicted in Fig. 4. The ring oscillator layout is shown in the red rectangle at the bottom in all cases. The Trojan1 is shown within the blue rectangle on the top of the RO in (b); and the Trojan2 is shown within the black rectangle on the left of the RO in (c).

Table I depicts the area overhead introduced by the Trojans. The design that includes Trojan1 uses the same number of Flip Flops (FFs) and 1.6% more LUTs compared to the Trojan-free design, while the design with Trojan2 uses 1.3% more FFs and 2.5% more LUTs.

For the RO measurements, a simulation process for 4.9 μsec was used. The RO is enabled at the 2.45 μsec so that it can gain a constant frequency. The RO counter counts the pulses up to the end of the simulation. Fig. 5 presents the RO measurements for the three designs.

The length of the RO varies from 7 to 21 with a step equal to 2. For small lengths of RO (9 and 11), the output changes up to 33%. For greater lengths (15, 17, 19, and 21), the output change ranges from 75% down to 18% for Trojan1. Thus, a RO length of 15 is the best choice. In the case of Trojan2 and small lengths (7, 9, and 11), the output change ranges from 33% down to 14.7%. For greater lengths (15, 17, 19, and 21), it ranges between 11.4% and 21.8%. Thus, a RO length of 7 is the best choice for this case. The RO with length equal to 13 does not cause any difference in RO output.

TABLE I. Area Overhead of Trojans

Hardware Resources	Original circuit	Incl. Trojan1	Incl. Trojan2
FFs	536	536	543
LUTs	475	483	487

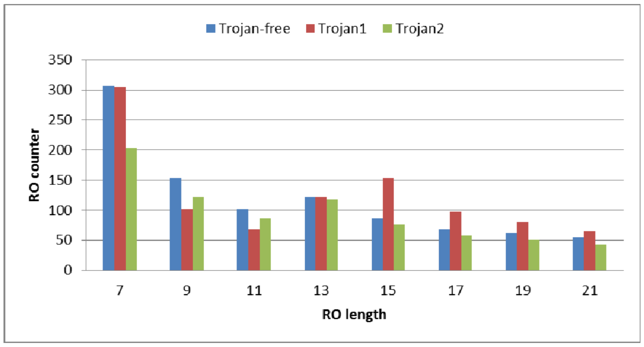

Fig.5. RO mesurements over different length

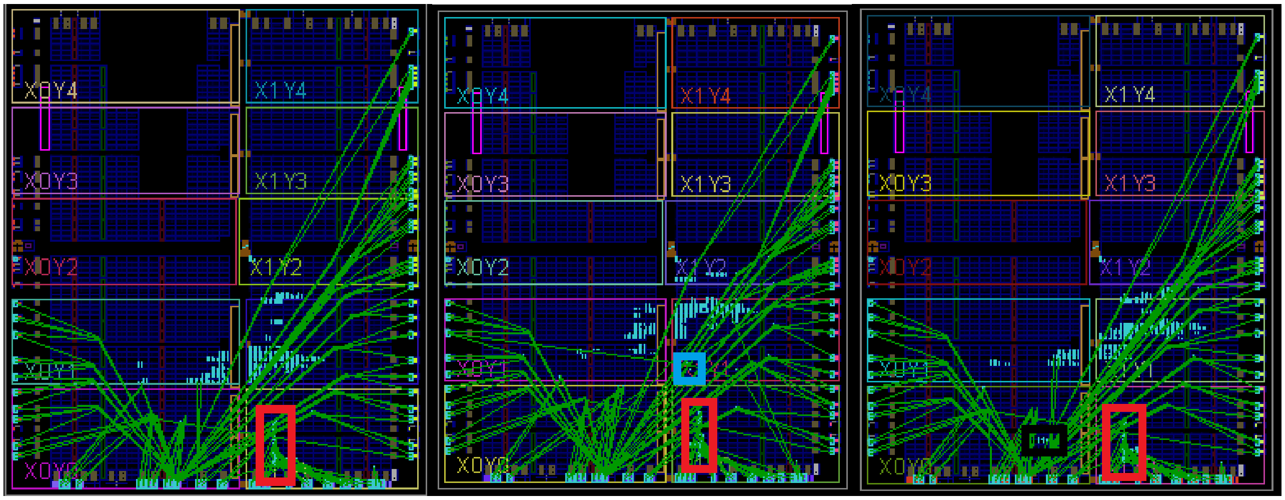

Fig. 4. Implementation layout of (a) the Mosquito cipher and the RO in red, (b) Trojan1 in blue, and (c) Trojan2 in black

The major reason that causes inaccurate measures in RO output is the slightly different place-and-route of the component inside the regions they are located. However, we note that the measured RO output changes for the inserted Trojans under the different lengths are outside the tolerance of inter-die and intra-die variations. Thus, they can detect the presence of a hardware Trojan horse.

V. Discussion and Future Work

In this paper, we explored the effect of ring oscillator length on detecting a hardware Trojan horse. Using a simulation environment based on Xilinx ISE webpack and Modelsim, we built two difference classes of Trojans affecting the operation of the Mosquito self-synchronizing stream cipher. The Trojans occupied a small percentage of the available area and ring oscillators of different lengths were placed in the circuit in a controllable fashion.

The length of the ring oscillator proved to be a crucial factor for detecting the Trojans. In some cases, the changes are negligible, way below the process variation, while in other cases we noticed changes of up to 75%. This is a clear indication that a ring oscillator will not always succeed in detecting the Trojan, no matter the size of the latter. However, by simply increasing or decreasing the length of the ring oscillator, the presence of the Trojan can be easily uncovered.

The two classes of Trojans required adjusting the length of the ring oscillator towards different directions. This is an area of future exploration, as to better understand the effect of length. Combining the input of multiple ring oscillators monitoring the same area may be a viable approach towards this direction.

Future work includes testing our assumptions using different algorithms (cryptographic primitives for encryption and hashing), different classes of Trojans (types of Trojans and activation complexity), and different FPGA implementations, towards devising an efficient and hierarchical testing methodology for reasoning on the existence of Trojans on a circuit.

References

[1] M. Tehranipoor and F. Koushanfar, “A Survey of Hardware Trojan Taxonomy and Detection”, IEEE Design and Test of Computers, Vol. 27, No. 1, pp. 10–25, January 2010.

[2] P. Kitsos and A. G. Voyiatzis, “Towards a Hardware Trojan Detection Methodology”, 2nd EUROMICRO/IEEE Workshop on Embedded and Cyber-Physica Systems (ECYPS 2014), Budva, Montenegro, 15-19, June 2014.

[3] X. Wang, M. Tehranipoor and J. Plusquellic, “Detecting Malicious Inclusions in Secure Hardware: Challenges and Solutions”, in Proc. of the 2008 IEEE International Workshop on Hardware-Oriented Security and Trust, Washington, DC, USA, pp. 15-19, 2008.

[4] R. S. Chakraborty, S. Paul, S. Bhunia, “On-demand Transparency for Improving Hardware Trojan Detectability”, IEEE International Workshop on Hardware-Oriented Security and Trust (HOST’08), pp. 48-50, 2008

[5] Y. Jin and Y. Makris, “Hardware Trojan Detection Using Path Delay Fingerprint”, in IEEE International Workshop on Hardware-Oriented Security and Trust (HOST’08), pp. 51-57, 2008.

[6] R. M. Rad, J. Plusquellic, M. Tehranipoor, “Sensitivity Analysis to Hardware Trojans Using Power Supply Transient Signals”, in IEEE International Workshop on Hardware-Oriented Security and Trust (HOST’08), pp. 3-7, 2008.

[7] S. Dutt and L. Li, “Trust-Based Design and Check of FPGA Circuits Using Two-Level Randomized ECC Structures”, ACM Transactions on Reconfigurable Technology and Systems (TRETS), Volume 2 Issue 1, March 2009.

[8] K. Xiao, X. Zhang, and M. Tehranipoor, “A Clock Sweeping Technique for Detecting Hardware Trojans Impacting Circuits Delay”, IEEE Design & Test, Vol. 30, Issue. 2, pp. 26-34, 2013.

[9] F. C. Bao and A. Srivastava, “Temperature Tracking: An Innovative Run-Time Approach for Hardware Trojan Detection”, IEEE/ACM International Conference on Computer-Aided Design (ICCAD), November 2013.

[10] I. Exurville, J. Fournier, J.-M. Dutertre, B. Robisson, A. Tria, “Practical Measurements of Data Path Delays for IP Authentication & Integrity Verification”, 8th International Workshop on Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC 2013), pp. 1-6, 2013.

[11] S. Bhasin, J.-L. Danger, S. Guilley, X. T. Ngo and Laurent Sauvge, “Hardware Trojan Horses in Cryptographic IP Cores”, Tenth Workshop on Fault Diagnosis and Tolerance in Cryptography, August 20, 2013, Santa Barbara, CA, USA.

[12] C. E. Irvine and K. Levitt, “Trusted Hardware: Can it be Trustworthy?”, 44th annual Design Automation Conference, pp. 1–4, June 2007.

[13] J. Rajendran, V. Jyothi, O. Sinanoglu, R. Karri, , “Design and Analysis of Ring Oscillator Based Design-for-Trust Technique”, VLSI Test Symposium (VTS), 2011 IEEE 29th , pp. 105-110, 1-5 May 2011.

[14] X. Zhang and M. Tehranipoor, “RON: An On-chip Ring Oscillator Network for Hardware Trojan Detection”, Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011, pp.1,6, 14-18 March 2011.

[15] J. Rilling, D. Graziano, J. Hitchcock, T. Meyer, X. Wang; P. Jones, J. Zambreno, “Circumventing a Ring Oscillator Approach to FPGA-based Hardware Trojan Detection”, *Computer Design (ICCD), 2011 IEEE 29th International Conference on* , pp.289,292, 9-12 Oct. 2011.

[16] A. Ferraiuolo, X. Zhang, and M. Tehranipoor, “Experimental Analysis of a Ring Oscillator Network for Hardware Trojan Detection in a 90nm ASIC”, Computer-Aided Design (ICCAD), 2012 IEEE/ACM International Conference on, pp.37,42, 5-8 Nov. 2012.

[17] A.Maiti, J. Casarona, L. Mchale, and P. Schaumont, “A Large Scale Characterization of RO-PUF”, IEEE International Symposium on Hardware Oriented Security and Trust, pp. 94-99, June 2010.

[18] X. Zhang, A. Ferraiulo and M. Tehranipoor, “Detection of Trojans Using a Combined Ring Oscillator Network and Off-Chip Transient Power Analysis”, ACM Journal on Emerging Technologies in Computing Systems (JETC), Volume 9 Issue 3, Article No. 25, September 2013.

[19] J. Daemen and P. Kitsos, “The Self-synchronizing Stream Cipher Mosquito”, First Phase of ECRYPT Stream Cipher Project Report 2005/018, 2005, Scandinavian Congress Center, Aarhus, Denmark, 26-27 May 2005.

[20] W. Xinmu, S. Narasimhan, A. Krishna, T. Mal-Sarkar, S. Bhunia, “Sequential Hardware Trojan: Side-channel Aware Design and Placement” Computer Design (ICCD), 2011 IEEE 29th International Conference on , pp.297,300, 9-12 Oct. 2011.