TERO-based Detection of Hardware Trojans on FPGA Implementation of the AES Algorithm

Paris Kitsos
Digital IC Design and Systems Laboratory (DICES Lab)
Computer and Informatics Engineering Department
TEI of Western Greece, Greece
e-mail: pkitsos@teimes.gr

Kyriakos Stefanidis
Industrial Systems Institute
“Athena” Research and Innovation Center in ICT and Knowledge Technologies
Platani, Patras, Greece
e-mail: stefanidis@isi.gr

Artemios G. Voyiatzis
SBA Research
Vienna, Austria

Abstract—A Transient Effect Ring Oscillator (TERO) is a special case of a Ring Oscillator (RO) design that exhibits increased sensitivity to intrinsic noise. It can serve as a basis for implementing a True Random Number Generator (TRNG) or a Physically Uncloneable Function (PUF). Also, as a digital sensor for detecting insertion of malicious hardware logic (Trojans) in digital circuits. Here, we explore the application of TERO for detecting hardware Trojans injected in FPGA implementations of the AES cryptographic algorithm. Experiments and comparisons are reported in terms of the frequency as a function of the TERO length. Our findings indicate that TERO-based digital sensors can be used to efficiently detect the presence of the Trojan.

Keywords—FPGA security; time analysis; ring oscillators; Transition Effect Ring Oscillator; hardware Trojan horse.

I. INTRODUCTION

It is very hard, nowadays, to guarantee that a hardware design is not infected by some malicious logic as a consequence of the globalization of integrated circuit (IC) manufacturing and the distributed supply chains involved in the design of a system.

The malicious logic inserted in the design is often mentioned as “hardware Trojan horse” or simply a “Trojan”. The malicious logic may perform different actions, depending on the motivations of its designer, ranging from circuit destruction to control take over and leakage of sensitive information through the primary outputs or side channels [1-2]. Hence, the need to integrate appropriate defenses in the IC design so as to detect the presence of malicious logic and defend against their potential activity, even if not detected in first place. Integrated mechanisms for Trojan detection are very useful given that exhaustive testing of all possible circuit states under all environmental conditions is not feasible in most of the cases.

A Ring Oscillator (RO) is a closed loop chain of an odd number of inverters. An example RO of length five is depicted in Fig. 1. An RO oscillates at a fixed frequency depending on the exact components, the size of a circuit, the operating characteristics (e.g., voltage) and the environmental conditions (e.g., temperature). Even minimal modifications of the circuit can result in a frequency change, rendering it very sensitive to process variations [3-4]. Many research works already proposed the integration of an RO close to sensitive parts of the circuit so as to detect Trojans [5-8].

A Transient (or Transition) Effect Ring Oscillator (TERO) is, in principle, a more sensitive variant of a ring oscillator. Previous works proposed the use of TEROs for implementing True Random Number Generators (TRNGs) and Physically Uncloneable Functions (PUFs) [9-10].

We introduced the use of TERO as a digital sensor for Trojan detection and studied its applicability in the case of simple Trojans against the cryptographic algorithms SNOW3G and Mosquito in [14, 15]. In this paper, we perform a comparison on the timing sensitivity of TERO against RO, towards introducing TERO as an alternative means for detecting Trojans implanted in FPGAs.

The rest of the paper is organized as follows. In Section II, we discuss RO and TERO implementations in FPGAs. In Section III, we describe the experimental setup and in Section IV, we analyze the results of our experiments. Finally, Section V provides the conclusions of this paper.

II. BACKGROUND

A ring oscillator with a sequence of five inverters in a closed loop (i.e., an RO with length 5) is depicted in Fig. 1. A counter fetches the output of the RO in order to measure the oscillation frequency.

![Fig. 1. Ring oscillator in FPGA](image-url)

A Transient Effect Ring Oscillator (TERO) is composed of an SR flip-flop implemented with two XOR gates and two AND gates [9]. This architecture has two control signals, for start and reset. The correct place-and-routing for a TERO is important so as to ensure the same length of the interconnections between the XOR gates.
Here, we use a simpler TERO architecture, where the XOR and AND gates are merged into NAND gates with some inverters in the feedback loop, as depicted in Fig. 2. The advantage of this approach is that only one control signal is used either for resetting or oscillating the TERO circuit.

The reset occurs when the control signal, `enable`, is set to ‘0’ and drives the loop to the same initial conditions before generating its output. When the control signal switches from ‘0’ to ‘1’, the TERO circuit starts to oscillate. An asynchronous counter is used to measure the TERO frequency.

### III. Experimental Setup

In order to investigate the effectiveness of the TERO, we realized a hardware Trojan horse against an FPGA-based implementation of the well-known AES cryptographic algorithm. We chose for our experiments the AES implementation that is provided on a Spartan 6 (XC6SLX75-2CSG484C) FPGA part of the SAKURA-G board [16]. The hardware architecture is discussed in detail in [17].

We designed a combinational Trojan, as depicted in Fig. 3. The Trojan trigger part comprises a tree of AND gates that monitors the values of a randomly selected subset of 8 out of the 128 key bits (namely, positions 3, 20, 41, 62, 75, 90, 100, and 119).

The Trojan payload part consists of a XOR gate that drives the `enc_dec` signal for defining the mode of operation (encryption or decryption) for the AES. Once the malicious logic is triggered (when the key bits match the pattern for Trojan), it inverses the selected mode of operation. This effectively creates a denial-of-service attack.

All the designs were captured in VHDL and were synthesized in the same FPGA board. The implementation of the RO sensor consumes 4-7 slices, while the one for a TERO sensor consumes 8-14 slices.

Fig. 5 depicts the area covered by the 27-bit-long RO sensor and the 19-bit-long TERO one respectively as a white trace. The same layout for the circuits with AES and TERO/RO is achieved and the hardware resources are placed and routed on the same FPGA locations.

The left part of Fig. 6 depicts the layout of the AES design with the 19-bit-long TERO (green rectangle in sub-region 2) and the Trojan (blue rectangle next to it, in sub-region 5). The right part of Fig. 6 depicts the layout of the AES design with the 27-bit-long RO (green rectangle in sub-region 9) and the Trojan (again, blue rectangle in sub-region 2).

The Trojan occupied a very small percentage of the available FPGA area compared to that needed for the AES implementation. Each slice in Spartan 6 comprises four 6-input Look-Up-Tables (LUTs) and eight flip-flops (FFs). The implementation of the RO sensor consumes 4-7 slices, while the one for a TERO sensor consumes 8-14 slices.

### Table 1. Sub-region positions of RO and TERO sensors

<table>
<thead>
<tr>
<th>RO positions</th>
<th>TERO positions</th>
</tr>
</thead>
<tbody>
<tr>
<td>3: X9Y66</td>
<td>3: X9Y66-X10Y66</td>
</tr>
<tr>
<td>6: X19Y66</td>
<td>6: X19Y66-X20Y66</td>
</tr>
<tr>
<td>2: X9Y57</td>
<td>2: X9Y57-X10Y57</td>
</tr>
<tr>
<td>5: X19Y57</td>
<td>5: X19Y57-X20Y57</td>
</tr>
<tr>
<td>8: X28Y57</td>
<td>8: X28Y57-X29Y57</td>
</tr>
<tr>
<td>1: X9Y46</td>
<td>1: X9Y46-X10Y46</td>
</tr>
<tr>
<td>4: X19Y46</td>
<td>4: X19Y46-X20Y46</td>
</tr>
<tr>
<td>7: X28Y46</td>
<td>7: X28Y46-X29Y46</td>
</tr>
</tbody>
</table>

Fig. 4. RO/TERO counter value capturing
The experimental setup comprises a personal computer interfacing the SAKURA-G board, the AES algorithm in ECB mode, the RO and TERO components with their respective counters, and the aforementioned Trojan design.

We experimented with the design using 500 randomly-generated test vectors and derived the oscillation counts for the RO- and TERO-enhanced designs. We also studied the effect of different lengths for the RO and the TERO, as proposed in [12].

In order to capture the value of the RO/TERO counters, the design implementation is instrumented with the Xilinx Chipscope (a Virtual I/O - VIO and an ICON) from the computer [13]. The counters are implemented in a DSP48A1 slice and have a width of 18 bits.

IV. RESULTS

The oscillation counts collected in our experiments for each sub-region are summarized in Table 2 (for TERO) and Table 3 (for RO). Each experiment was repeated five times to mitigate any measurement error by averaging the counts.

The distance (difference) between the oscillation counts of the Trojan-free and the infected circuits is the best metric regarding the reliability and the detection sensitivity.

The TERO sensor is more sensitive (increased counts) when the length is smaller. The closer the TERO sensor to the Trojan, the biggest the count difference when a Trojan is inserted. The sensitivity decreases consistently as the length increases for all sub-regions. For lengths greater than 19, no sub-region sensor is able to detect any difference with the Trojan-free design.

The RO sensor exhibits a similar behavior with TERO in count differences as a function of the length. However, we noticed that the RO sensor counts are totally unstable from run to run in our experiments. Hence, they cannot be used to reliably detect the presence of the Trojan, as they produce many false positives. Also, due to this instability, it is not possible to directly compare the sensitivity of the TERO sensor against the RO one and, thus, confirm the theoretical result that a TERO would oscillate at about double the frequency of an RO of equal length [9].

V. CONCLUSIONS

A TERO-based design can serve as a sensor for detecting malicious hardware logic inserted in a design. We explored the case of protecting the AES cryptographic algorithm implemented on an FPGA board and in comparison with a RO sensor. The latter proved quite unstable; the reasons for this instability should be explored in the future work. The TERO sensor performed reliably in all cases. It was possible to observe significant differences in the oscillation counts when a tiny combinational Trojan was inserted in the design. These differences were more evident when sensors with small lengths were used. Thus, TERO-based sensors can be used as an efficient defense mechanism for FPGA designs.

As future work, we plan to further investigate the performance of TERO sensors of different lengths, as a means to increase their sensitivity and also study the effects of using multiple sensors of different lengths (in one sub-region or spread in different ones). Also, to confirm our findings with different samples of Trojans and on different FPGA boards.
Table 2: Oscillation counts (values in hexadecimal) with TERO

<table>
<thead>
<tr>
<th>Length</th>
<th>Region 1</th>
<th>Region 2</th>
<th>Region 3</th>
<th>Region 4</th>
<th>Region 5</th>
<th>Region 6</th>
<th>Region 7</th>
<th>Region 8</th>
<th>Region 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>5208</td>
<td>5014</td>
<td>3478</td>
<td>5014</td>
<td>4F3D</td>
<td>5014</td>
<td>4A38</td>
<td>4C4B</td>
<td>5014</td>
</tr>
<tr>
<td>19</td>
<td>3D04</td>
<td>3FDC</td>
<td>3E80</td>
<td>3E80</td>
<td>3E80</td>
<td>3E80</td>
<td>3E80</td>
<td>3E80</td>
<td>3E80</td>
</tr>
<tr>
<td>23</td>
<td>34BC</td>
<td>34BC</td>
<td>34BC</td>
<td>34BC</td>
<td>34BC</td>
<td>34BC</td>
<td>34BC</td>
<td>34BC</td>
<td>34BC</td>
</tr>
<tr>
<td>27</td>
<td>2CEC</td>
<td>2CEC</td>
<td>2CEC</td>
<td>2CEC</td>
<td>2CEC</td>
<td>2CEC</td>
<td>2CEC</td>
<td>2CEC</td>
<td>2CEC</td>
</tr>
</tbody>
</table>

Trojan-free

Table 3: Oscillation counts (values in hexadecimal) with RO

<table>
<thead>
<tr>
<th>Length</th>
<th>Region 1</th>
<th>Region 2</th>
<th>Region 3</th>
<th>Region 4</th>
<th>Region 5</th>
<th>Region 6</th>
<th>Region 7</th>
<th>Region 8</th>
<th>Region 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>EF59</td>
<td>5468</td>
<td>874C</td>
<td>473D</td>
<td>2526</td>
<td>D40D</td>
<td>34CD</td>
<td>4838</td>
<td>513B</td>
</tr>
<tr>
<td>19</td>
<td>459E</td>
<td>53E2</td>
<td>F4FB</td>
<td>3258</td>
<td>245C</td>
<td>3B03</td>
<td>3709</td>
<td>2DF4</td>
<td>23D0</td>
</tr>
<tr>
<td>23</td>
<td>3FD3</td>
<td>4303</td>
<td>3186</td>
<td>3BBF</td>
<td>16D0</td>
<td>268A</td>
<td>2EF0</td>
<td>208F</td>
<td>2939</td>
</tr>
<tr>
<td>27</td>
<td>336A</td>
<td>3E24</td>
<td>31C4</td>
<td>203E</td>
<td>3690</td>
<td>2E8D</td>
<td>2296</td>
<td>2B8F</td>
<td>2B03</td>
</tr>
</tbody>
</table>

Trojan-free

ACKNOWLEDGEMENT
This work was partially supported by the COMET K1 program of the Austrian Research Promotion Agency (FFG).

REFERENCES