2015 10th International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS)

On the Limitations of Logic Testing for Detecting Hardware Trojans Horses

Marie-Lise Flottes, Sophie Dupuis, Papa-Sidy Ba, Bruno Rouzeyre
LIRMM (Université Montpellier/CNRS UMR 5506)
Montpellier, France
{firstname.lastname}@lirmm.fr

Abstract — **The insertion of malicious alterations to a circuit, referred to as Hardware Trojan Horses (HTH), is a threat considered more and more seriously in the last years. Several methods have been proposed in literature to detect the presence of such alterations. Among them, logic testing approaches consist in trying to activate potential HTHs and detect erroneous outputs by exploiting manufacturing digital test techniques. Besides the complexity of this approach due to the intrinsic stealthiness of the potential HTH, we will show that a particular HTH targeting the test infrastructure itself may jeopardize the possibility of detecting any other alterations.**

***Index Terms*—Hardware Trojan; Logic testing**.

I. INTRODUCTION

With ever-shrinking transistor technologies, the cost of new fabrication facilities is becoming prohibitive and outsourcing the fabrication process to low-cost locations has become a major trend in Integrated Circuits (ICs) industry in the last decade. This trend raises the question about untrusted foundries that might intentionally insert malicious circuitry or alterations, referred to as Hardware Trojan Horses (HTHs) [1, 2, 3]. Furthermore, various steps of the design flow can be outsourced, among others, the use of third-party Intellectual Properties (IPs). This leads to various vulnerabilities. Potential threats such as HTHs insertion have become a major concern and therefore extensive research has been conducted focusing on techniques to detect such threat in the last years [4, 5]. The most investigated approaches rely on side-channel measurements [6, 7], manufacturing testing techniques [8, 9], as well as optical microscopic imaging [10].

Due to the diversity of HTHs, different classifications have been proposed. The classification in [8] proposes a basic model of a HTH circuit. This model presents the two major components of a HTH circuit: the activation mechanism (referred as the *trigger*) and the introduced effect (referred as the *payload*). The trigger can consist in the occurrence of several signals to a certain value. It can also be delayed by a counter. The payload can result in a default service, or a secret information leakage.

A logic trigger and payload are modeled as shown in Figure 1. The triggering logic monitors a set of inputs to activate the payload. Based on the assumption that the HTH should be stealthy in order to minimize its detection, it is assumed that the triggering condition is a function of some signals having low controllability. This type of HTH is referred to as *rare-value triggered HTH*.

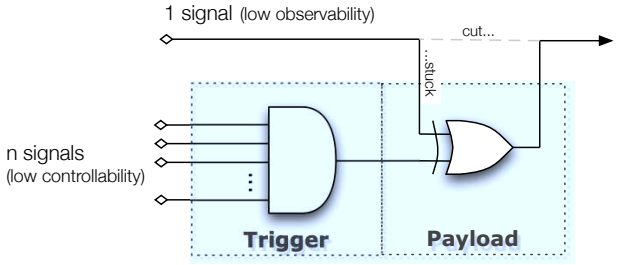

Figure 1. Rare value triggered HTH circuit model [8]

In this paper we focus on the activation of the rare-value triggers by using manufacturing test techniques and infrastructures. Indeed, Design-for-Testability (DfT) structures, such as scan-based design, allow easily justifying any state of an IC, thus reducing the test application time and the effort required by Automatic Test Pattern Generator (ATPG) tools to detect efficient test patterns. Logic testing techniques to detect HTHs make the assumption that scan chains can be used to find the alteration of the circuit. Nevertheless, we will show that a very small trigger would be able to prevent the activation of the HTHs at test time, i.e. may evade from HTH detection approaches based on logic testing techniques.

This paper is organized as follows. In Section II, we recall the different proposed HTHs detection methods based on logic testing. In Section III, we briefly describe the scan-based test method and we present a possible trigger that would disable the payload of the HTH during test operations. In Section IV, we present some alternatives to counteract this type of trigger, by showing the limitations of such approaches. Finally, Section V concludes the paper.

II. HTH DETECTION BY TEST TECHNIQUES

Logic testing consists in triggering potential HTHs in order to detect them during test procedure [8, 9]. HTHs are inserted in stealthy nature, i.e., they are inactive most of the time unless triggered by a rare condition. Therefore, the main concern is to find test patterns that can maximize the chances of triggering potential HTHs. The most important

978-1-4799-1999-4/15/$31.00 ©2015 IEEE

advantages of logic testing are that, as opposed to side channel analysis for instance, it is robust with respect to environment and process variability and it does not require a golden circuit for result comparison.

The assumption considered in all papers in literature is that a HTH has a hidden nature, i.e., it is activated under very rare conditions because otherwise classical production testing activities would easily reveal it. Exhaustive testing, i.e., the use of all possible combinations of inputs and states of the circuit, would eventually trigger a HTH. However, this technique is not feasible with modern circuit sizes. The goal is thus to identify a reduced set of input patterns that maximizes the probability of activating potentials HTH.

The first logic-based detection approach is presented by Wolff et al. in [8]. The idea is to find the "most likely target sites" to attach and stitch a HTH and then to generate test patterns according to this prediction. The procedure to find potential triggers sites consists in identifying low controllability circuit internal signals. An exhaustive testing is done when possible, otherwise, a test with a pseudorandom set of input patterns. Considering a HTH with q triggers, the goal of this simulation is to collect all values occurring on any combinations of q signals with a low controllability. All values with a low frequency of occurrence under a given threshold (possibly 1, i.e., a value occurring for only one pattern during the whole simulation) are possible trigger values. The procedure to find potential payload sites is based on the use of a fault simulator to identify signals with a low observability. From the set of *Q* target triggers (with their trigger values) and *P* target payloads, *QxP* possible HTH circuits are considered. An ATPG tool is then used to find the corresponding HTH test vectors. Since it checks whether each trigger value can be propagated to the circuit output, it results in a compacted set of trigger vectors.

In [9], Chakraborty et al. propose a methodology called Multiple Excitation of Rare Occurrence (MERO). The assumption is that the number of times a HTH trigger condition is satisfied increases with the number of times the trigger signals have been individually excited to their rare value. This results then in increasing the probability to trigger the HTH. The procedure is based on a set of random patterns, a list of rare signals and the number of times to activate each signal to its rare value. For each random pattern, the procedure counts the number (*Sr)* of signals whose rare value is satisfied. The random patterns are sorted in decreasing order of *Sr*. Each pattern in the sorted list is modified with the perturbation of one bit at a time. If a modified pattern increases *Sr*, the pattern is accepted. The procedure repeats until each signal satisfies its rare value condition for the desired number of times.

Test vector generation has been also used in conjunction with side-channel analysis methods. Indeed if only a portion of a HTH is activated (and therefore is not detected by logic testing), the HTH will nevertheless consume more dynamic power [2]. In [11], Banga and Hsiao proposed a vector generation technique to magnify potential HTH contributions while minimizing the circuit activity. The idea is to repeat multiple times each test vector in order to ensure the reduction of extraneous toggle within genuine circuits. In [12] the same authors propose to generate test vectors that maximize the switching activity within one region while simultaneously minimizing the switching activity for the rest of the circuit.

These methods have the same goal: generating a reduced set of test vector that are more likely to trigger or to excite potential HTHs. All methods are based on the fact that test vectors can be applied to the circuit.

Nevertheless, the application of test vectors to the circuit (even for random values) is effective only if scan chains are used, thus allowing higher controllability and observability of the states of the circuit. The methods in [8] and [9] do not investigate the use and the limitations of scan chains.

In the next sections we first recall how scan chains are used for digital testing, and we show how scan chains can be tampered in order to hide even more the presence of a HTH.

III. HTH WITHIN THE TEST INFRASTRUCTURE

Before describing the proposed HTH, we briefly recall the test strategy based on scan design. The description we present in this paper is not exhaustive and it mainly presents the basic principles of scan chain designs. While the description does not present advanced solution used in real industrial designs, it provides all the elements required to understand how a HTH might be introduced in such an infrastructure.

A. Test infrastructure

The scan design methodology consists first in converting selected storage elements into scan cells (cf. Fig. 2) and then stitching them together to form one or several shift registers called scan chain (cf. Fig. 3). The scan cell is composed of a D Flip-Flop (FF) and a multiplexer that uses a Scan Enable signal to select between the data input (Din) and the scan input (Scan In). The design can therefore work in two different modes of operation: functional and scan. In functional mode (Scan Enable equal to 0), all FFs receive the functional output of the combinational part of the original circuit. In scan mode (Scan Enable asserted), the Scan In input is used to shift into the FFs a desired state of the IC, while the previous content of the cell is shifted out. Any test stimulus and test response can therefore be shifted in and out of the N-cell scan chain in N clock cycles.

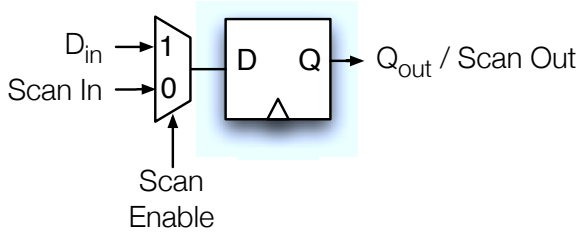

Figure 2. Multiplexed D scan cell

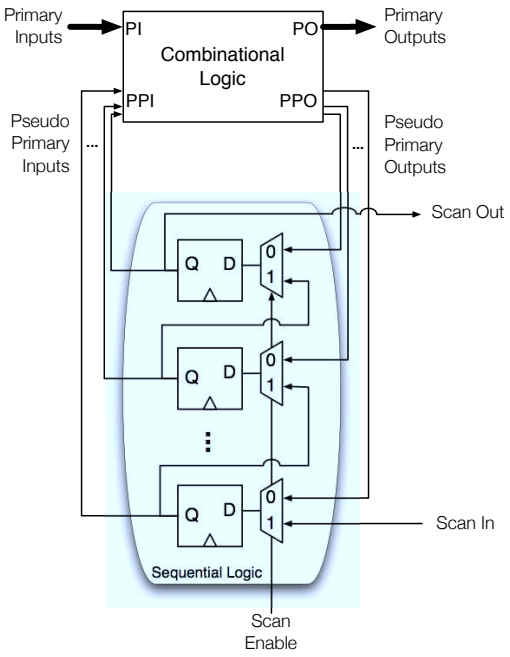

Figure 3. Scan-based Testing

Detecting a stuck-at fault in the combinational part of the circuit consists in switching to scan mode (Scan enable = 1) to shift the desired target state, then switching back to functional mode while required test patterns are applied to the primary inputs in order to capture the possible fault effect in both primary output and FFs. Finally, switching back to scan mode to shift out FFs content to allow comparison with expected values. Shifting in the next test stimulus for detecting another stuck-at fault can be performed concurrently with the shift-out operation of the last test response. Shift-in/out operations are generally performed at slow-speed in order to prevent voltage drop and over-eating due to the tremendous number of switching activities in the circuit exercised by test data shifted in the scan chain. This procedure is summarized in Fig. 4a.

Delay defects that affect the functionality of the design if the circuit is run at high speed can also be detected thanks to the scan design approach. Two test stimuli, vectors V1 and V2, are required in this case because a transition has to be launched in order to propagate across the path under test. V1 initializes the circuit and V2 launches the transition. Delay test can be applied in different ways: Launch-On-Capture (LOC) and Launch-On-Shift (LOS). For LOS testing, the first stimulus V1 corresponds to the last clock cycle of the scan mode (i.e., the last shift in the FFs causes the fault to be excited). The capture operation performed in functional mode allows then observing the presence of a possible delay fault. On the contrary, for LOC testing, the scan mode is used to reach the state where V1 is applied in normal mode, followed by a second clock cycle still in normal mode (scan enable is not asserted for two consecutive clock cycles). This procedure is summarized in Fig. 4b.

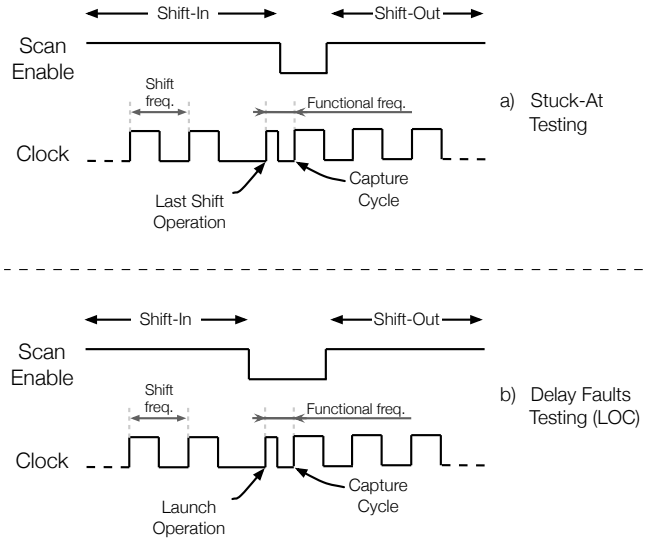

Figure 4. Test procedures with scan chains

B. *Proposed HTH*

A potential attacker might consider tampering the test infrastructure in such a way that the payload of the HTH is not activated when test patterns are applied through the scan chain. Besides strongly affecting the possibility of activating an HTH through test vectors, such an alteration is straightforward to design and easy to implement.

In fact, detecting test operations is possible by observing the Scan Enable and the clock signals. If a functional clock cycle (i.e. a capture cycle, Scan Enable = 0) is surrounded by several shift operations (Scan Enable = 1), it means that a test procedure is running. In other words, in the case of the test procedure targeting delay faults, if in the previous two clock cycles the scan enable signal is asserted then the HTH must not be activated. Fig. 5 shows the implementation of a state machine able to detect such a condition.

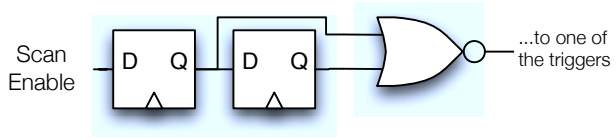

Figure 5. Proposed trigger for inhibiting the HTH activation during test operations

In some cases, the number of capture cycles during the test procedure can be higher than two. For instance, the test of embedded memories can require up to ten capture cycles between two shift operations. In order to make the proposed trigger even more robust against detection techniques, an attacker might consider adding as many flip-flops as necessary.

IV. Detection Methods and Limitations

In order to evaluate the limitation of logic testing, we implemented several HTHs in an AES circuit. All of these HTHs are triggered on a rare value condition, as presented in Fig. 6. Moreover, all the triggers possess the mechanism proposed in Fig. 5 to inhibit their activation during scan test operations. Besides, all of these HTHs payloads consist in a XOR gate inverting the logic value of an internal signal when triggered (cf. Fig. 1).

We used the process described in [13] to find the set of low controllable signals in the circuit. These signals were then used to implement six different three-input triggers Ti such that e.g. Ti = AND (S1, /S2, S3), where Si signals are low controllable to 1, and /Si signals are low controllable to 0. Rare values that would activate the trigger when they have the value "1" are directly connected to the AND gate, while those who are activated for the "0" are first inverted.

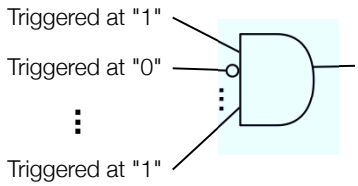

Figure 6. Generic trigger model

We used an ATPG to find a test pattern able to excite each triggering condition (i.e. Ti=1). The idea was to generate a vector to cover a stuck-at-0 at the output of each trigger (see Fig. 6). Indeed, a test vector detecting the stuck-at-0 would justify the value "1" at the output of the AND, that is possible only if all input values are verified (i.e., the triggering condition).

Since we assumed that the scan chain couldn't be used, we first used a commercial sequential ATPG [14]. The tool was able to generate a test pattern for only one trigger condition (Ti=1) out of six, due to limitations in sequential test pattern generation. A question then arises: how to generate test patterns for such triggers?

We made the assumption that the protection of the trigger against the scan testing is based on the check of the Scan Enable signal during the last two clock cycles. We therefore aim at finding a test pattern from which the application of three consecutive input vectors at the primary inputs will allow detecting the HTH (see Fig. 7).

A possible procedure is then to: (1) use the ATPG to find which pattern to set in the scan chain to trigger the HTH (denoted P1 afterwards), (2) find the corresponding circuit state three functional clock cycle before (denoted P2). The circuit will then be tested in the following way: (1) scan in P2 by setting the scan enable to 1, (2) set scan enable to 0, (3) apply three clock cycles. The pattern P1 is then set in the scan chain in functional mode.

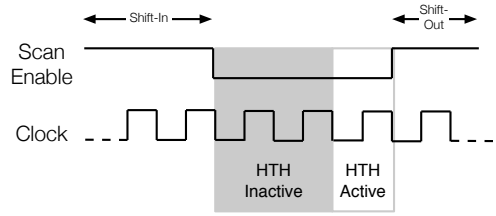

Figure 7. Possible detection techniques for the proposed trigger

With this procedure, we are able to compute test patterns for sequential circuits without relying on pure sequential ATPG. Furthermore, the test pattern is applied without activating the trigger inhibitor presented in Fig. 5.

From a practical point of view, the process to generate the test pattern consists in unrolling the sequential part of the circuit for three time frames (see Fig. 8).

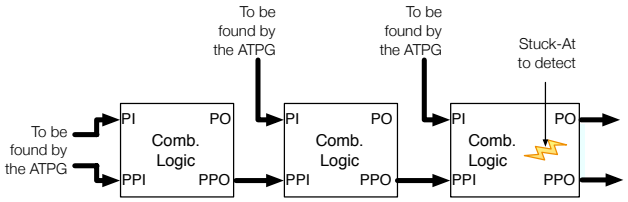

Figure 8. Unrolling technique for generating test vectors unaffected by the proposed trigger

By running the ATPG with the procedure described before wards, we obtained test patterns for five HTHs over six.

The testing procedure of Fig. 8 is however limited by complex circuits that have several divergences/re-convergences and a high number of inputs. This is the reason why the pattern for the last HTH was not found by the ATPG. Moreover, adding some FFs in the proposed trigger (Fig. 5) would even more reduce the possibility of enabling the HTH, but would result also in a more complex HTH.

V. Conclusion

Several methods have been proposed in literature to detect the presence of HTHs. Among them, approaches based on logic testing consist in trying to activate potential HTHs and detect erroneous outputs by exploiting manufacturing digital test techniques. Besides the complexity of these approaches due to the intrinsic stealthiness of the potential HTH, we have shown how a very small alteration of the HTH trigger would deny the possibility of using test infrastructures in order to detect a possible HTH.

Therefore, the test of the test infrastructure itself is of primary importance in order to assure its reliability. Possibly, other techniques, based on both side-channels analysis and/or optical inspection, will be necessary to increase the confidence of the use of test infrastructures in detecting possible HTHs.

REFERENCES

[1] X. Wang, M. Tehranipoor and J. Plusquellic. Detecting Malicious Inclusions in Secure Hardware: Challenges and Solutions. In *IEEE International Workshop on Hardware-Oriented Security and Trust (HOST'08*), pp.15–19, 2008.

[2] M. Tehranipoor and F. Koushanfar. A Survey of Hardware Trojan Taxonomy and Detection. In *IEEE Design & Test of Computer,* 27:10– 25, 2010.

[3] S. Bhunia, M. S. Hsiao, M. Banga, S. Narasimhan. Hardware Trojan Attacks: Threat Analysis and Countermeasures. In *Proceedings of the IEEE, Special Issue on Trustworthy Hardware*, 102(8):1229–1247, 2014.

[4] J. Rajendran, O. Sinanoglu, R. Karri. Regaining Trust in VLSI Design: Design-for-trust Techniques. In *Proceedings of the IEEE, Special Issue on Trustworthy Hardware*, 102(8):1266–1282, 2014.

[5] P. Kitsos, A. Voyiatzis. Towards a Hardware Trojan detection methodology. In *Mediterranean Conference on Embedded Computing (MECO'14)*, pp. 18–23, 2014.

[6] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, B. Sunar. Trojan Detection using IC Fingerprinting. In *IEEE Symposium on Security and Privacy (SP'07)*, pp. 296–310, 2007.

[7] Y. Jin, 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.

[8] F. Wolf, C. Papachristou, S. Bhunia and R. S. Chakraborty. Towards Trojan-Free Trusted ICs: Problem Analysis and Detection Scheme. In *Design, Automation and Test in Europe (DATE'08)*, pp. 1362–1365, 2008.

[9] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia. MERO: A Statistical Approach for Hardware Trojan Detection. In *International Conference on Cryptographic Hardware and Embedded Systems (CHES'09),* pp. 396–410, 2009.

[10] S. Bhasin, J.-L. Danger, S. Guilley, X. T. Ngo, L. Sauvage. Hardware Trojan Horses in Cryptographic IP Cores. In *Fault Diagnostic and Tolerance in Cryptography (FDTC'13)*, pp. 15–29, 2013.

[11] M. Banga, M. Hsiao. A Novel Sustained Vector Technique for the Detection of Hardware Trojans. In *International Conference on VLSI Design (VLSI'09)*, pp. 327–332, 2009.

[12] M. Banga, M. Hsiao. A Region Based Approach for the Identification of Hardware Trojans. In *International Workshop on Hardware-Oriented Security and Trust (HOST'08)*, pp. 40–47, 2008.

[13] G. Di Natale, S. Dupuis, M.-L. Flottes, and B. Rouzeyre. Identification f Hardware Trojans triggering signals. In *First Workshop on Trustworthy Manufacturing and Utilization of Secure Devices (TRUDEVICE'13), 2013.*

[14] Synopsys Tetramax, http://www.synopsys.com/Tools/Implementation/RTLSynthesis/Test/Pages/TetraMAXATPG.aspx