Duplication-based Concurrent Detection of Hardware Trojans in Integrated Circuits

Manikandan Palanichamy, Papa-Sidy Ba, Sophie Dupuis, Marie-Lise Flottes, Giorgio Di Natale, Bruno Rouzeyre

Duplication-based Concurrent Detection of Hardware Trojans in Integrated Circuits

Manikandan Palanichamy, Papa-Sidy Ba, Sophie Dupuis, Marie-Lise Flottes, Giorgio Di Natale, Bruno Rouzeyre

LIRMM (Université Montpellier II /CNRS UMR 5506)
Montpellier, France
Firstname.lastname@lirmm.fr

This project has been funded by the French Government (BPI-OSEO) under grant FUI#14 HOMERE (H ardware

Abstract— **Outsourcing the fabrication process to low-cost locations has become a major trend in the Integrated Circuits (ICs) industry in the last decade. This trend raises the question about untrusted foundries in which an adversary can tamper with the circuit by inserting a malicious behavior in the ICs, referred to as Hardware Trojans (HTs). The serious impact of HTs in security applications and global economy brings extreme importance to detection as well as prevention techniques. In this paper, we introduce the idea of hardware modified dual modular redundancy (MODMOR): a prevention technique that aims at making the insertion of a stealthy HT more difficult, and at detecting it at run-time.**

Keywords—Hardware trojns, detection at run time, prevention

I. INTRODUCTION (*HEADING 1*)

Hardware devices are nowadays extremely important in many ever growing safety-critical application fields such as civil, commercial, communication, space, healthcare and military. The criticality of these applications increases the demand for trusted Integrated Circuits (ICs). Since the manufacturing of ICs is more and more complex and costly, outsourcing one or more steps of the IC build cycle has become the standard way of operation in the semiconductor industry. As an inevitable unwanted side effect this outsourcing business model increases threats to hardware products.

The ICs can be manipulated by possibly inserting malicious circuitry or alterations, referred as Hardware Trojans (HTs). An IC with more than a million transistors can easily be altered by few hundred transistors, causing severe effects such as denial of service, generating false signals, creating backdoors, bypassing authentication methods, leaking secret information processed by ICs, and even permanently damaging the device at run-time. Obviously, fabless IC companies, which send to silicon foundries their design in the form of layout (in GDSII format), cannot ensure that their final IC product is HT free. This new scenario gives security and economy challenges to semiconductor industries, and it motivates researchers to study and investigate solutions to address them further (e.g. [1-4]).

In order not to be easily detected, HTs are supposed to be stealthy and activated only under rare conditions. Therefore, detecting HTs becomes an extremely challenging task. In this paper we propose to explore a run-time technique based on the monitoring of the processed data. This type of solution has been widely studied in order to increase the reliability of digital systems [5-11]. More precisely, we explore a hardware redundancy solution based on duplication and comparison, a.k.a., Dual Modular Redundancy (DMR). In this technique, a redundant and functionally equivalent processing unit is added to the original circuit. Processed data at the output of the two replications are compared at run-time while both replications are fed by identical input sequences. Any mismatch between data processed by these replications reveals a faulty behavior.

DMR is used here for detection of HT. We thus assume that the same 'duplicated' HT cannot not be inserted in both replications. Otherwise, when triggered, both replications would output the same but faulty response and the comparison would not reveal the problem. In order to prevent easy insertion of the same HT on both replications, the main idea is to implement two functionally equivalent designs with different gate level netlists (distinct GDSII).

The rest of the paper is organized as follows: Opportunities for exploring Design for Reliability (DfR) are given in Section II. Section III describes the principle of Concurrent Error Detection (CED) technique, and experimental results are presented in Section IV. Finally, Section V concludes the paper.

II. OPPORTUNITIES FOR EXPLORING DFR

Design-for-Reliability (DfR) provides several redundancy-based solutions for fault-tolerant systems: information redundancy, time redundancy or hardware redundancy. These design approaches provide robustness to failure since each data computation is checked thanks to extra information, respectively, codes computed from data processed through the system, data computed from a second postponed execution through the same hardware, or data obtained from extra-executions through physically duplicated hardware, so called modular redundancy. Dual modular redundancy relies on two replicated elements operating in parallel. The same inputs are provided to each replication and the same outputs are expected.

This project has been funded by the French Government (BPI-OSEO) under grant FUI#14 HOMERE (Hardware trOjans : Menaces et robustEsse des ciR cuits intEgrés).

These outputs are compared at run time and any mismatch is used to trigger an alarm or a recovering process.

In this paper, we propose to use the idea of dual redundancy concept in order to detect HT (recovery is therefore out of the scope of this paper). For the same reason, we do not explore the expensive triple (or higher) modular redundancy approach where the extra circuitry used for output comparison allows determining which replication is faulty among the three, and thus outputs the correct result.

Note that dual redundancy can be used at test time just after manufacturing but also during the circuit life-time since DfR approaches, by nature, are used for detection of erroneous data at run time.

Redundancy was already explored in [13-14] for hardware Trojan detection and prevention. The authors use complementary codes, but they report an area overhead ranging from 500% to 600%. Due to coding and decoding blocks on the datapath, this technique impacts also performance.

In comparison, our technique, called MODMOR, involves roughly 100% overhead and has no impact on performances.

III. PROPOSED APPROACH

A. MODMOR

The MODMOR approach relies on the duplication of the circuit and on the comparison of the outputs.

To the best of our knowledge, such idea was not investigated earlier for HT detection.

Figure 1 illustrates the basic concept. Since the input sequences are the same and the functions are initialized to the same state, the comparator should always report OK.

In the case a HT has been inserted in one of the two circuits, and when the HT is activated during run-time, the outputs differ. Thus the comparator checks it and can be used to set an alarm for instance.

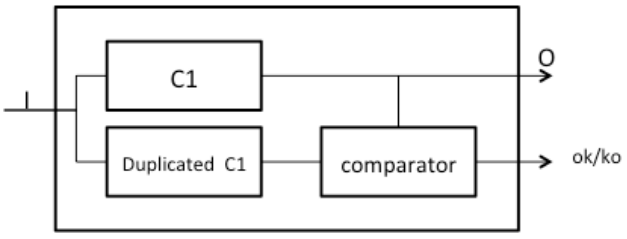

Fig. 1. General architecture

If the original circuit C1 and its replica are built exactly in the same way, an attacker might be able to insert the HT in both circuits, thus not being detected by the proposed solution.

In order to prevent such a scenario, we propose a solution that increases the complexity of inserting the same HT in both designs.

The basic principle underlying the proposed solution is based on synthesis of two different layouts for the original circuit and for its functionally equivalent replica.

The implementation of two different netlists that behave similarly and with the same time constraints is not supported by classical CAD tools during logic synthesis. The synthesis operation can indeed generate various alternative netlist embodiments from a single functionality (provided by a HDL description). Alternative netlists of the same function can be achieved by exploiting different synthesis options such as:

1. State encoding (with different number of FFS)
2. Number of state machines,
3. Design constraints,
4. Restrictions on gate libraries,
5. Different Synthesis tools.

An example is given in Fig.2, where, S1 and S2 are two identical functions implemented differently.

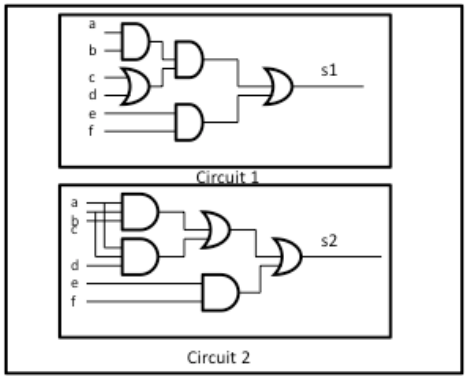

Fig. 2. Original netlist and its duplicated version

Finally, if both versions are simultaneously placed and routed with intertwining of standard scells, it is quite impossible for an attacker to identify, from the inspection of the layout, each sub-circuit.

The design flow of MODMOR approach is shown in Figure 3 in which F(X) and G(X) are two equivalent RTL codings of the same circuit.

B. Attack analysis

We assume that there are three different ways to insert an HT in such a circuit, where two different netlists are used to compute the same functionality:

1. If the HT is inserted in only one of the two circuits, this will be detected by the comparator;
2. Since that the layout has been produced as indicated above, i.e., by using two different implementations for the same function, it is very unlike that the same HT can be inserted in both versions. For instance, if the HT is inserted at point 'A' in netlist1, it will be difficult to understand the equivalent functional point 'A' in in netlist2;
3. Thirdly, an attacker may insert a HT in one of the circuit and alter the comparator output in order to always produce OK. This issue is addressed in next section.

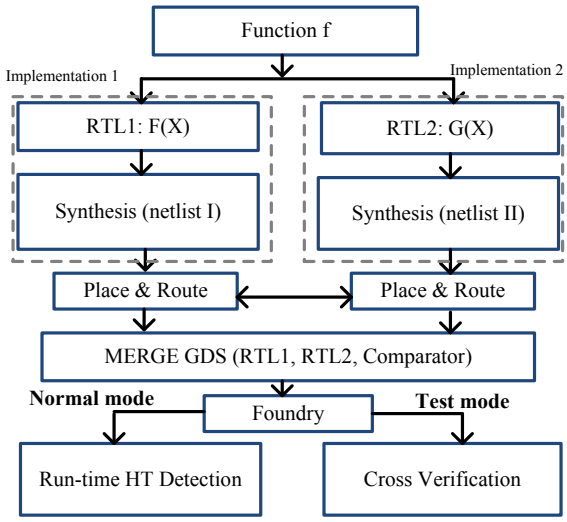

Fig. 3. MODMOR approach: design flowPrinciple of CED Technique

C. *Cross verification logic*

The detection of possible HTs strongly depends on the comparator output, and this module can become the weak link of the whole solution. For this reason, the comparator must be deeply tested, even by adding spare logic.

The idea is illustrated in Fig. 4. On this example, a supplementary input k and spare logic gates are added to one of the circuitsone of the versions. The added logic is such that S1 and S2 exhibit the same values when k=0 and are different when k=1.

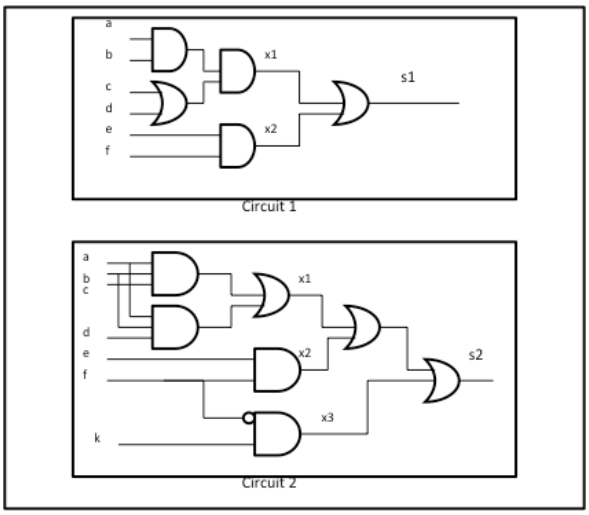

Fig. 4. Added logic to check comparator

The check of the comparator is performed off-line. In the example shown in Fig. 5, by setting k=0, S1 and S2 are equal, and the comparator should always output OK whatever the values on (a,b,c,d,e,f). At the opposite, by setting k=1 and vector (a,b,c,d,e,f) = (0,0,0,0,1,0), S1 will produce '0' while S2 will produce '1'. Thus, the comparator should output KO.

In mission mode, input k is set to 0.

IV. Experimental Results

The presented approach has been evaluated through experiments using ST65nm library for synthesis, and Synopsys IC compiler for placement and routing. As experimental vehicle, we took a RSA engine. The same timing constraints have been set during synthesis of both versions.

The RSA algorithm [15] is widely used in cryptosystems. We implemented the RSA design in VHDL. Here, Version 1 has a single state machine while the state machine has been partitioned in two concurrent ones in Version 2 in the RTL code. However, both versions have exactly the same behavior. RTL coding creates identical designs which will produce different gate level netlists after synthesize and thus very different layouts.

Table 1 shows the synthesis results for V1 and V2 individually as well as dual cores with comparator.

The synthesis results for both cores are given in the two first columns.

The whole circuit was synthesized using two flows. In case 1, we synthesized the complete design using a top module embedding the vhdl code of both designs and of the comparator. Case 1 column from Table 2 shows the optimized area results.

In case 2, both designs are synthesized separately, and combined with the comparator during placement & routing.

As expected, case 1 produces better area results.

Figure 5 shows the layout of case 1 and case 2. There is a visible difference between these two layouts. Case 1 exhibits more congested (greens lins) than case 2. Thus, adopting the first design flow seems the better option to implement Modmor.

TABLE I. Synthesis Results

	Vers. 1	**Vers. 2**	**Dual versions with comparator (Final output: Single GDSII netlist - flattened**	
			Case 1: Combined synthesis using top module, and then P&R	**Case 2: Separate synthesis of modules and combined during P&R**
Comb. cells	1556	1605	3137	3214
Sequ. cells	501	506	1013	1008
Comb. cell area	5833.8	5979.4	11589.2	11990.1
Non-comb. cell area	4691.9	4743.9	9403.1	9446.3
Total cell area	10525.8	10723.4	20992.3	21436.4
Total area after P&R (um^2)	18110	18250	33014	33782

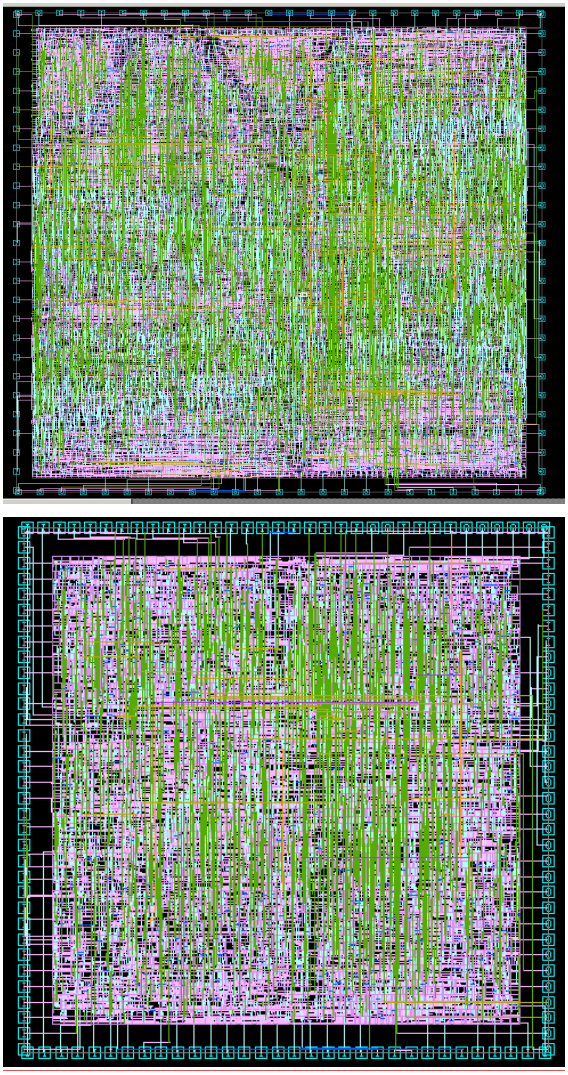

Fig. 5. Case 1 and Case 2 layouts

V. CONCLUSIONS

In this paper, we discussed about employing design-for testability and duplication based concurrent HT detection in ICs. This run-time solution addresses HT insertions not only from foundries, and also during RTL stages. Hardware duplication based CED with different implementations of the same logic function have a significant advantage in providing protection against HT insertions.

REFERENCES

[1] W. Danesh, J. Dofe, and Q. Yu “Efficient Hardware Trojan Detection with Differential Cascade Voltage Switch Logic”, in Intl. journal on Special issue on Advanced VLSI Architecture Design for Emerging Digital Systems, Vol. 2014, pp. 1-11, Jan. 2014.

[2] Y. Jin and Y. Makris, “Hardware Trojan detection using path delay fingerprint”, in Intl. workshop on Hardware-Oriented Security and Trust (HST), Pages 51-57June 2008.

[3] S. Dupuis, B. Rouzeyre, M. L. Flottes, G. Di Natale, P-S Ba, “ New Testing Procedure for Finding Insertion Sites of Stealthy Hardware Trojans”, ", in Proc. Intl. conf. on Design, Automation and Test in Europe (DATE'), March 2015

[4] M.-L. Flottes S. Dupuis, P.-S. Ba, B. Rouzeyre, "On the Limitations of Logic Testing for Detecting Hardware Trojans Horses". In Design & Technology of Integrated Systems (DTIS’15), 2015.

[5] C. V. Ramamoorthy, and Y-W Han, “Reliability Analysis of Systems with Concurrent Error Detection,” IEEE Trans. Computers, Vol. C-24, No. 9, pp. 868-878, Sept. 1975.

[6] M. Y. Hsiao, W. C. Carter, J. W. Thomas and W. R. Stringfellow, “Reliability, Availability and Serviceability of IBM Computer Systems: A Quarter Century of Progress,” IBM Journal of Research and Development, Vol. 25, No. 5, pp. 453-469, Sept. 1981.

[7] N. K. Jha, and S. J. Wang, “Design and Synthesis of Self-checking VLSI Circuits,” IEEE Trans. CAD, Vol. 12, pp. 878-887, June 1993.

[8] D. K. Pradhan., Fault-Tolerant Computer System Design, Prentice Hall, 1996.

[9] S. Mitra, N. R. Saxena and E. J. McCluskey, “A Design Diversity Metric and Reliability Analysis for Redundant Systems,” lntl. Test Con., pp. 662-671, 1999.

[10] C. Zeng, N. R. Saxena and E. J. McCluskey, “Finite State Machine Synthesis with Concurrent Error Detection,” Proc. lntl. Test Con., pp. 672-680, 1999.

[11] I. Pomeranz and S. M. Reddy, "Concurrent On-Line Testing of Identical Circuits Through Output Comparison Using Non-Identical Input Vectors", in Proc. Intl. Symp. on Defect and Fault Tolerance in VLSI Systems, pp. 469-476, Oct. 2004.

[12] I. Pomeranz and S. M. Reddy, " Selecting State Variables for Improved On-Line Testability Through Output Response Comparison of Identical Circuits", in Proc. Intl. On-line Testing Symp. (IOLTS), pp. 179-184, July 2010.

[13] Xuan Thuy Ngo, Sylvain Guilley, Shivam Bhasin, Jean-Luc Danger, Zakaria Najm, "Encoding the State of Integrated Circuits: a Proactive and Reactive Protection against Hardware Trojan Horses". WESS 2014: 9th Workshop on Embedded Systems Security.

[14] Claude Clarlet, Abderrahman Daif, Jean-Luc Danger, Sylvain Guilley, Zakaria Najm, Xuan Thuy Ngo, Thibault Porteboeuf, Cédric Tavernier "Optimized Linear Complementary Codes Implementation hor Hardware, Trojan Prevention" ECCTD 2015.

[15] Rivest, R.; A. Shamir; L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". Communications of the ACM 21 (2): 120–126