2015 European Conference on Circuit Theory and Design (ECCTD)

Hardware Trojan Prevention using Layout-Level Design Approach

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

***Abstract*—Hardware Trojans (HTs) are ultimately a dangerous threat in semiconductor industry. The serious impact of HTs in security applications and global economy brings extreme importance to their detection and prevention techniques. This paper focuses on developing a HT prevention techniques through a layout level design approach. The principle is to let no available space on silicon for an attacker to insert a HT. Experiments determine the maximum occupational rate and critical empty spaces while filling with standard cells. The proposed technique makes HT insertion nearly impossible.**

Keywords—Hardware Trojans (HTs); HT prevention, layout; occupation ratio; Design for Hardware trust

I. INTRODUCTION

Today's integrated circuits (ICs) are complex and costly. Particularly to reduce the product cost, the IC chips have to cross several trusted and untrusted phases before becoming the final product. For example, one company designs the IC, but several other companies often make critical contributions in terms of fabrication, packaging, distribution etc. It results in outsourcing one or more steps of the IC production process to foreign countries. As a side effect, this outsourcing business model increases threats in hardware products. The ICs can be manipulated with the possible insertion of malicious circuitry or alterations, termed as Hardware Trojans (HTs). This can partly or entirely control the IC and give challenges to the security and trust worthiness of the hardware product.

HTs can be inserted either during design phase through RTL code or CAD tools, or during fabrication or packaging. However, fabrication phase is considered the most dangerous among others [1] where the foundry has its own freedom, and may insert HTs to the circuit without affecting the functionality or area or performance of the original design. Few hundred transistors are enough to insert malicious behavior in a circuit with a billion transistors. These threats are giving tremendous challenges in high security applications such as military, air force, communications, space crafts, health care, finance and so on.

The global economy loss of the semiconductor industry is estimated as $4 billion annually due to IP infringement [2]. These security and economy challenges from HTs motivate us to study, and investigate solution techniques to address them further. Generally, these HTs are stealthy, and will be activated only under rare conditions. It makes the HT detection an extremely challenging task. Besides HT detection, an alternative solution is to prevent their insertion in ICs. In this paper, we present such a HT prevention technique using a layout level design approach. In this technique, functional cells and shift registers are added in the existing empty spaces of a circuit layout, instead of filler cells. The ultimate goal of adding such cells is to form several blocks of combinational functions independent of each other. Shift registers are used to handle inputs and outputs of combinational functions. Testing these added functions by applying shift-in values, and observing shift-out values helps to determine whether or not the in-built functions were modified during fabrication implying that a cell were removed or replaced.

The rest of the paper is organized as follows: Section II describes layout placement analysis with respect to filler cells, and Section III explains the layout level design approach in details. Experimental results are presented in Section IV. Finally, Section V concludes the paper.

II. PLACEMENT ANALYSIS AND PREVIOUS WORK

During circuit design, placement and routing are subsequent steps of logic synthesis process. This is done according to constraints such as the clock period and the silicon occupation ratio. Standard cells are placed in order to improve routing. As a consequence of the placement, there are empty spaces between active cells. These empty spaces affects the continuity on VDD/GND rails as well as for n-well, and therefore the placement tool uses filler cells or decoupling capacitor cells to fill-up these spaces. The filler cells do not provide any functionality to the circuit but are used to establish the continuity of the n- well and the implant layers on the standard cell rows. If the design needs any modifications then design engineers can replace these non-functional filler cells with functional filler cells, which is called Engineering Change Order (ECO) but it is a seldom case.

In practice, foundries can replace these non-functional filler cells with their own version while fabricating the chip. Therefore, it is possible to replace filler cells intentionally by any functional cells in order to introduce malicious behavior to the original design. In foundry stage, inserting HTs or stealthy circuitry to the original design is quite possible, and it is almost impossible to detect them by normal testing. The reasons are that the HTs can be intelligently inserted in such a way that there is no impact on function, area, power and performance of the original design. It will also be inserted in a way that it can pass typical testing [3].

In order to address these un-trusted foundry phase attacks, the layout empty spaces should be utilized properly, and not giving any room to attackers. This idea was proposed in [4], in

This work has been funded by the French Government (BPI-OSEO) under grant FUI AAP14 HOMERE+

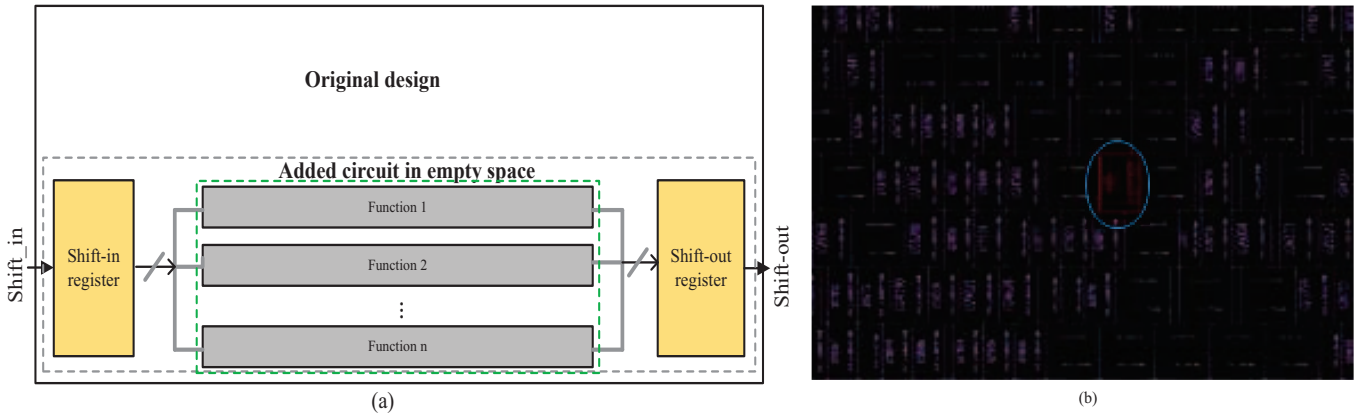

Fig. 1. (a) Block diagram of layout level design approach and (b) Magnified view of the empty space in the original design

which empty spaces are filled with functional standard cells. Therefore, if an attacker modifies these functional cells for HT insertion, then its predetermined output response will be altered. Then it will be easy to identify the possible modification, or HT insertion. The functional cell selection and the formation of combinational functions in the empty spaces in our work is influenced by [4].

However, the approach in [4] uses a test pattern generator (TPG) and output response analyzer (ORA) in its structure. An intelligent attacker can always set the desired output at the ORA or multiple input shift register (MISR), then there is a freedom to replace the existing empty space functions with any HT. Protecting TPG from attacks is therefore needed and it increases the complexity of the structure. This drawback is addressed in this work, which replaces both TPG and MISR with shift registers. Also, achieving 100% occupation ratio is impossible in complex circuits. An intelligent tactic regarding the choice of the spaces to fill in priority is of interest, which is not investigated in [4]. We therefore developed a tool that places functional standard cells first in so called "critical" empty spaces, as explained in the next section.

III. Layout Level Design Approach

Figure 1 (a) shows the block diagram of the layout level design approach, and Figure 1 (b) shows the magnified empty space in the layout of the original design. The key principle of this method is to fill the pre-determined, and prioritized critical empty spaces in the layout with functional standard cells. These functional standard cells are connected together to form several combinational functions as shown in Figure 1 (a), that are independent from the original design. Thus, the addition of these extra functions does not alter the performance of the original design.

To test the combinational functions, we need to apply input patterns, and observe corresponding output patterns. In order to apply such inputs and receive outputs, simple shift-registers are used at both input and output sides of the combinational functions. These shift-registers are also implemented within unused empty spaces. The shift-registers are using a separate clock that is used to control the test mode. If the shift-registers or combinational functions are altered during fabrication that can be detected by comparing post fabrication shift-out values against expected shift-out values under test mode. The global flow of the layout level design approach is shown in Figure 2, and it is described in the following sub-sections.

A. *Critical empty spaces*

After synthesis and placement of the original design, all the available empty spaces of the layout (GDS II) computed. Then, there is a process of identifying “critical” empty spaces. We developed a tool that performs a timing analysis to compute the slack time of each circuit signal as presented in Figure 3. Slack time at any timing is the difference of its required arrival time and its arrival time [5]. Concerning signal A for example, the arrival time is 0+2=2, and the required arrival time is 6-2=4, and therefore the slack value is 4-2=2. Once this slack information is known for each, it gives the information about signals with a large slack time. These signals are considered dangerous. The reason is that they are insensitive to HT insertions. They will indeed not result in any degradation in the overall timing performance of the original design due to HT insertion, and will be insensitive to HT detection techniques based on delay measurements [6].

The more accurate the estimation of the delay is, the better. This delay computation is therefore done after place and route of the original design in order to take into account delays of both gates and interconnects. Furthermore, this is what reflects best the information that an attacker can obtain from the GDSII sent to the foundry. Given the coordinates of the empty spaces and the slack of each signal, an empty space is close to so-called dangerous signal with a large slack, then we consider that space as a “critical” empty space. That is because there is a possibility for an attacker to insert a HT in that kind of empty spaces, and connect it to the original design by taking advantage of the large slack of the signal. These critical empty spaces will be prioritized first during the filling with flip-flops and standard functional cells. Therefore, these critical empty spaces cannot be utilized by an attacker to insert HTs.

B. *Filling and function formation*

After computing critical empty spaces from the original design, there is a process of filling them with shift-registers and standard functional cells from the physical library. The functional cells are selected from the physical library based on each cell’s a) geometrical information (ex. cell name, width, length, and coordinates of four corners), b) decoupling capacitance value, and c) the available size of critical empty

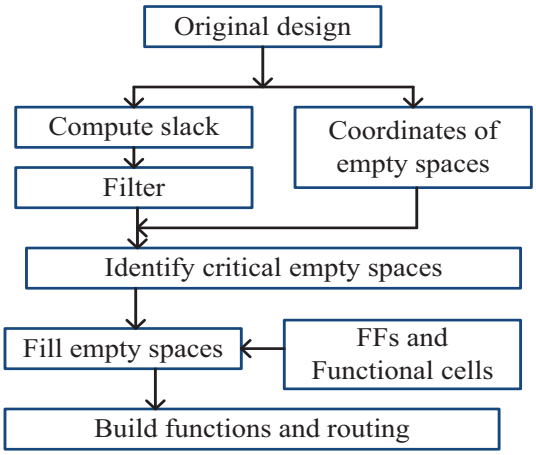

Fig. 2. Global flow

spaces. The geometrical information helps to identify the size of each cell in order to match with the size of the critical empty spaces. Also, cells with larger decoupling capacitance values helps to compensate the absence of DECAP cells [7].

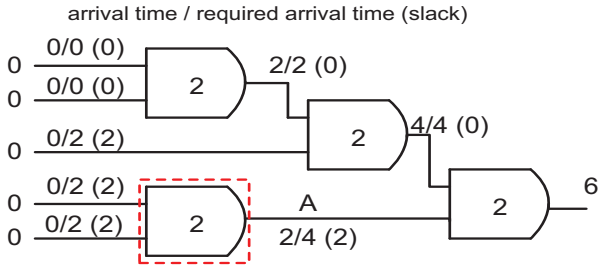

Fig. 3. Slack computation

The filling process is as follows. First, the pre-determined critical empty spaces are prioritized in the order of larger to smaller size. It means that the critical empty spaces with large size are chosen first to get filled, and then subsequent size spaces are filled. First, shift-in and shift-out registers are constructed. For any chosen empty space, flip-flops are inserted at first and then the tool computes the rest of the space until the space is not enough to accept any more flip-flops. The second step consists in additional function construction. Once registers are built-up, the tool starts to fill the left-over gap of the chosen empty space using functional cells. These cells are also prioritized according to their size. The tool tries to place large size cells first, and if they do not fit then it tries the next size until it tries the smallest cell from the library. This process is repeated in all critical empty spaces, as well as in other available non-critical empty spaces until reaching targeted final occupation ratio, or until the space left is inferior to the smallest cell from the library. However, the filling process may leave some usable empty spaces especially in complex circuits with low value initial occupation ratio because the placement & routing tool is aiming to reach the final and targeted occupation ratio while filling. If the left-over spaces are critical, and large enough then an attacker can easily utilize them to insert HTs. That's why our approach prioritizes critical empty spaces to fill. The initial and final occupation ratios are discussed in experimental results.

After all cells and shift registers are placed in the empty spaces, they have to be connected to form a number of combinational functions. The extra functions are built in such a way that a. as few gates as possible are used in each function b. as many functions as possible to be created. This criteria is retained in order to reduce the redundant gates and to increase the testability of the functions. A tree structure is used to build each function, and function inputs are independent of each other. It improves their controllability and observability during testing. Also while forming combinational functions, the closest cells are connected together in each function in order to make the routing easier. This helps to reduce interconnects lengths in functions.

IV. Experiments and Analysis

The presented layout level design approach has been evaluated in different circuits. AES engine, and ISCAS s13207, s35932, and c6288. Experiments were conducted with ST65nm library and Synopsys tools for synthesis, placement and routing together with tight timing constraints. The results are discussed in the following in terms of critical empty spaces filling, and analysis of different attacks.

A. Critical empty spaces filling

For each benchmark, Table I presents the chosen initial occupation ratio, final occupation ratio as well as, the number of flip-flops, and standard cells added by our approach. First, before filling in critical empty spaces the original design occupies a certain area in the layout which is called initial occupation ratio. It can set by the designer. Second, we start to fill the critical empty spaces with flip-flops and functional standard cells until the maximum limit or until reaching routing violations. This is called final occupation ratio.

TABLE I. Critical Empty Space Filling

Benchmark	Occupation ratio (Initial)	Occupation ratio (Final)	Added FFs	Added cells	Left usable space (%)
AES	60	94	220	9258	1
	70	93	41	6011	0.1
	80	90	34	2116	0
s13207	60	95.7	56	478	0
	70	92.8	35	296	0
	80	100	13	349	0
s35932	60	89	414	3468	0
	70	95.7	269	2249	0
	80	98	193	1361	0
c6288	60	93.75	35	297	0
	70	92.94	22	191	0
	80	95.34	18	129	0

For example, first part of Table I shows the experiment results for AES. In a first experiment, we set the initial occupation ratio to 60%., and our approach added flip-flops and standard cells upto 94% without any routing violations. We were able to add 220 flip-flops and 9258 standard cells. We increased the initial occupation ratio from 60% to 70% and 80%. When the initial occupation ratio increases, the added number of flip-flops and functional standard cells are decreasing. We could not go more than 80% since we require space for flip-flops (for shift-in and shift-out purposes), and the size of each flip-flop is higher

than other cells. We chose to form combinational functions with 10 inputs. Then flip-flops were therefore used for shift-in registers, and the rest of the flip-flops were distributed in shift-out registers in order to receive function outputs.

In order to make sure that the rest of the left over space does not contain any usable empty spaces, we again tried to add more cells from the library. For example, AES with 60% initial occupation and 94% final occupation ratio can still accept cells 1% more as shown in the last column of Table I. However, it makes the routing incomplete with violation. Then we did the same experiment on AES with the initial occupation ratio of 70%, and it results in the possibility to add 0.1% more cells, but with routing violations. Again we increased the initial occupation ratio to 80%, and, it could not accept cell insertions anymore because the size of left-over empty spaces were not usable or much smaller than the smallest cells from the physical library. Also in other benchmarks, we were able to achieve 0% usable left-over spaces.

B. *Attack analysis*

This layout level design approach is sensitive to different attacks in terms of gate level HT insertions. From the attacker point of view, it is difficult to identify the difference between added functions and original design. The reason is that both are functional, and taken from the same library. To analyze pessimistic scenario from the designer point of view, assuming that if an attacker identifies added functions then there is a possibility for potential attacks. In order to investigate these attacks, we have conducted some experiments on AES, and the results are shown in Table II. After functions formation in empty spaces of the original design, we have applied particular input and observed the generated signature which is considered a "good or expected signature". Then, we performed several possible attacks on functional cells and flip-flops in the empty spaces, and generated signatures using same inputs. The results are shown in Table II, and the purpose of these experiments is to verify the resultant effect of this HT prevention method.

a) Removak attack: To create space for HTs an attacker can remove one or more cells from the empty spaces which is called removal attack. In order to address this first we generated a signature before attack which is 24'h867ff7 for the input 10'h007. Then, we removed OR3X9, and observed the resultant signature for the same input, which is 24'h867fff. It differs from the expected one from before attack. We did similar experiments by randomly removing other gates from combinational functions, and compared its signature against the expected ones. We found that removal of any cell in combinational functions affect the resultant signature from the expected one.

b) Replacement attack: To insert a HT, an attacker can replace any cell in combinational functions which is called replacement attack. In order to address this we first generated a signature before attack (24'h867ff7) for the input 10'h007 which is the expected one. After replacing AO112X18 with AND4X6, the generated signature (24'hbefffe) differs from the expected one. It was done in several other cells, and the resultant signature after attack always differ from the expected one.

TABLE II. ATTACK ANALYSIS

Attack on empty space	**Attacked component**	**Input**	**Output Signature**	
			Before attack	**After attack**
Removal	OR3X9	10'h007	24'h867ff7	24'h867fff
	OA31X18	10'h007	24'h867ff7	24'h867ff3
Replacement	AO112X18 / AND4X6	10'h007	24'h867ff7	24'hbefffe
	XOR2x9/ NAND3X5	10'h007	24'h867ff7	24'hbeffff
FF removal	DFF	10'h007	24'h867ff7	24'hfefffZ

c) Shift-register attack: It is also possible to remove flip-flops from input or output shift-registers in order to utilize that space for HT insertions, and it is called shift in/out attack. We have removed a single flip-flop. The generated signature varies from the expected one.

After each attack the resultant signatures always differ from the expected ones. It shows the layout level design approach easily identifies HT insertions or makes HT insertion as difficult as possible to an attacker.

V. CONCLUSIONS

In this paper, we have presented an approach to identify critical empty spaces which are close to signals with large slack. The inserted functional cells protect the original design from HT insertions, and it improves the degree of confidence in the trust worthiness of the IC products particularly from untrusted foundries. This work used almost all usable empty spaces by adding cells, and therefore HT insertion is nearly impossible. Then routing become indeed an issue in case of several cells added. Besides, if the empty space cells are altered by an attacker, then it will be easily detected from the resultant signature comparisons as shown in the experimental results. Also, these added cells has negligible impact on the original design in terms of power, and zero impact on silicon area and performance.

REFERENCES

[1] Defense Science Board study on high performance microchip supply. *http://www.acq.osd.mil/dsb/*

[2] SEMI, "Innovation is at risk as semiconductor equipment and materials industry loses up to $4 billion annually due to IP infiringment", *www.semi.org/en/Press/P043775, 2008*.

[3] P-S. Ba, S. Dupuis, M.L. Flottes, B. Rouzeyre, "On the Limitations of Logic Testing for Detecting Hardware Trojans Horses". In Design and Technology of Integrated Systems (DTIS'15) conf. April 21-23, 2015, Naples, Italy.

[4] K. Xiao, M. Tehranipoor, "BISA: Built-in self-authentication for preventing hardware Trojan insertion", In International symposium on Hardware-oriented security and trust (HOST'13), pp. 45-50, 2013.

[5] G. DiNatale, S. Dupuis, M.L. Flottes, B. Rouzeyre, "Identification of Hardware Trojans triggering signals", In Workshop on Trustworthy Manufacturing and Utilization of Secure Device, 2013.

[6] 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.

[7] C. Yeh and M. Marek-Sadowska, "Timing-aware power noise reduction in placement," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 26, Issue 3, pp. 527-541, 2007.