Hardware Trojan Detection Techniques Using Side-Channel Analysis

Thi-Tam Hoang[1], Thai-Ha Tran[2], Van-Phuc Hoang[2], Xuan-Nam Tran[2] and Cong-Kha Pham[1]

[1]The University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo, Japan
[2]Le Quy Don Technical University, 236 Hoang Quoc Viet Str., Hanoi, Vietnam
Email: hoangtam@vlsilab.ee.uec.ac.jp

***Abstract*— Hardware Trojan (HT) is a malicious modification of the integrated circuits during design or fabrication processes. It can extract the secret information or change the circuit behavior. Many HT detection techniques have been proposed in recent years. This paper focuses on side channel analysis (SCA) based approach due to its advantages over other approaches. We review the state-of-the-art researches, present a proposed path delay based SCA method and discuss the trend for developing new HT detection techniques in the future.**

Keywords*— *Hardware Trojan, side channel analysis, path delay

I. INTRODUCTION

Nowadays, in order to reduce the production cost as well as shorten the production time, almost chip manufacturers outsource the chip production process to other companies. However, this also increases the risk of insecurity. Normally, manufacturers perform chip testing upon completion, but these tests are not always able to detect hardware Trojan (HT) because HT is often designed to be very small or only activated under special conditions. Therefore, methods for effectively detecting HT is urgently needed. This article focuses primarily on analyzing methods based on side channel analysis because it has many advantages and is currently being researched and developed by many researchers.

HT can be inserted into the circuit by an adversary in some steps in chip fabrication process. Figure 1 shows the HT threat at different stages of IC life cycle [1]. As can be seen, the structure of a circuit can be modified by an adversary in fabrication or design process. The objective of this paper is to give an overview about state-of-the-art SCA based HT detection techniques and point out a typical implementation of these techniques in FPGA platforms.

The remainder of this paper is organized as follows. Section II provides an introduction about HT detection techniques. Section III focuses on the existing techniques for HT detection based on SCA. Section IV presents the proposed path delay based SCA method for HT detection in FPGA. Then, section V concludes the paper.

II. HARDWARE TROJAN DETECTION TECHNIQUES

An HT consists of two parts by common, namely Trigger and Payload. Trigger is the condition that HT changes from the inactive state to the active state. Payload executes the HT's function. Depending on the target of attacker, HT can extract secret information or change behavior of the circuit. Classification of the existing HT detection techniques is shown in Fig. 2. Firstly, HT detection techniques is divided into two categories including destructive and non-destructive methods. Destructive technique utilizes reverse engineering to rebuild the netlist or layout of the circuit. In this technique, layers of the circuit is reconstructed by chemical or optical methods [2]. Destructive technique can verify the existence of HT in the circuit at very high accuracy. However, this technique has some disadvantages such as high cost, time consuming and destructive tested circuit. This means that the tested circuit cannot be used anymore. Therefore, most of the academic researches focus on developing the non-destructive techniques.

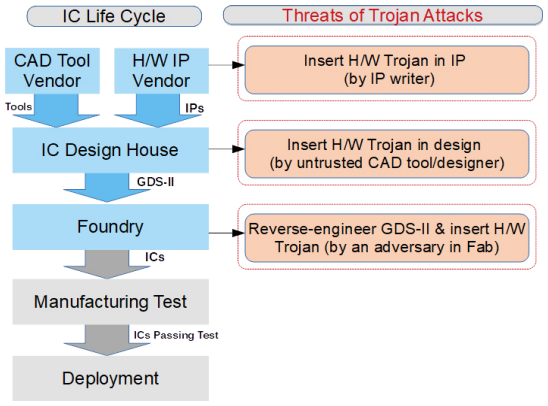

Fig. 1. Hardware Trojan threats at different stages of IC life cycle, adapted from [1].

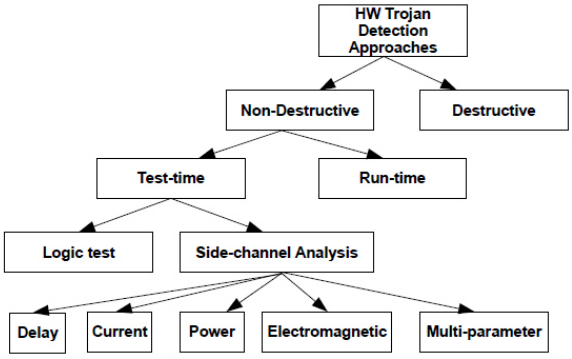

Fig. 2. HT detection techniques.

Thereafter, non-destructive techniques are divided into test-time and run-time techniques. The run-time techniques use an online monitoring system that tracking the circuit activity during operation while test-time techniques tries to

detect HT before the release. Test-time techniques consist of two categories: logic test and SCA based techniques. To avoid being detected during post-silicon test, HT is often designed to be triggered in very rare conditions. Logic testing methods tries to generate test vectors that activates HT so that HT will be detected. Logic testing methods is effective for ultra-small HT and strong to process noise. Compared to logic testing, side channel analysis methods have some advantages. Firstly, in logic testing approach, HT is only detected when it is activated. Secondly, it is difficult to generate test vector. On the other hand, for ultra-small HT, SCA-based methods face a huge difficulty especially when the test environment is not stable. A small change in test condition can affect the experiment result. In the next section, SCA-based techniques will be discussed in more detail.

III. SIDE CHANNEL ANALYSIS BASED HARDWARE TROJAN DETECTION TECHNIQUES

Side-channel analysis (SCA) is considered as one of the most effective technique to detect HT. Since any change in circuit will lead to change in physical parameters, SCA is expected to be able to detect a wide range of HT. In SCA-based HT detection techniques, side channel signals such as current, power, electromagnetic and delay are employed. Among them, power and delay are more popular. In comparison, in most of power-based methods, Trojan is detected only when it is active while delay-based methods can detect HT even when it is inactive. However, in most of delay-based methods, extra structure is added to the original circuit. So, they cannot applied for circuit that received after fabrication or the circuit that cannot be modified [1].

Beside the process and environmental variation, in SCA-based methods, HT detection probability is strongly affected by measurement accuracy of side channel parameters. Higher measurement accuracy will improve HT detection probability. More and more HT detection method has been proposed. However, it is not easy to compare and evaluate them because the set of HT for testing is not identical. An approach can reach a high detection rate with Trojans have a certain attribute. Table 1 gives comparison of up-to-date researches on SCA-based HT detection methods.

A. *Power Analysis*

In [4], HT detection is based on fingerprinting method. Fingerprint of an IC family is a set of side-channel signals (power signal is used in this research) getting at I/O test on some HT free ICs of the family. HT free ICs is validated by destructive test. Other ICs are checked under the same I/0 test to compare their side-channel signals with the side-channel fingerprint of the family. The proposed technique can identify Trojan that is as small as 0.01% of the total IC size in the presence of ±7.5% process variations.

B. *Current Analysis*

Both static current [3] and transient current [4] can be used to identify existence of HT in the circuit. In [3], static current is measured simultaneously at multiple places on the chip. The proposed region-based approach can deal with process and environment variation. The outstanding point of this research is that it is carried out on chip, not on FPGA as other researches.

C. *Path Delay Analysis*

Path delay is also a useful parameter used in HT detection techniques [5]-[7]. In [5], the method is based on clock glitches. The comparison of the critical path between golden model and tested ICs is used to detect HT. The HT detection technique in [7] is similar to technique in [8]. However, rather than power fingerprint, path delay fingerprint of the whole chip is used. The number of chips used to gather path delay fingerprint is more than those in [8]. The proposed technique can detect quite small HT in the presence of ±7.5% process variations. However, measuring all paths in large size ICs is not practical.

D. *EM Analysis*

Another SCA type used to detect HT is the electromagnetic (EM) signal [2], [9]. In [9], sequential denial-of-service (DoS) Trojan is placed next to the I/Os of the FPGA for experiment. Trojan was inserted in 6 places in the 6 different FPGA layouts: in the bottom-left corner, in the top-right corner, in the center, distributed over the whole layout, at the right side near the I/O lines and automatically after the design hat been completely re-routed. By comparing the EM emission of HT inserted FPGAs with genuine ones, it is shown that HTs placed at the corner of the FPGA and in re-routed design are more detectable than other ones.

E. *Multi-Parameter Analysis*

Multi-parameter analysis is expected to improve performance of HT detection technique. In [8], the authors utilize relationship between the maximum operating frequency (Fmax) and dynamic current of a circuit to eliminate effect of process variations on the circuit. To improve test time and Trojan detection coverage (in term of Trojan type and size), the combination of SCA-based and logic testing method is also proposed.

F. *Golden Model Requirement*

Most of the SCA-based HT detection techniques require a golden model to compare side channel information of golden model with side channel information of chip under the test. Golden model is a data set of side channel information that is obtained from Trojan free chips. In [8], an approach called IC fingerprinting for obtaining golden model was proposed. Golden model is built by following step:

- Step 1: Randomly select some ICs from a set of ICs that manufactured in the same fab;
- Step 2: Collect side-channel information of the selected ICs;
- Step 3: Validate the selected ICs by destructive testing;
- Step 4: Save the data in step 2 as golden model if all of the tested ICs are Trojan-free.

In recent years, researchers try to develop golden model free techniques [10], [11], [6] as a promising research direction since these free techniques are immune to process variations. Authors in [10] proposed a method called equal-power self-authentication to identify patterns that consume equal power in the same IC. Trojans are detected by comparing equal power pairs. The technique can detect Trojans equal to 0.023% of total circuit area. Similarly, to eliminate the golden model, a self-referencing technique is used in [6]. The key idea is to find symmetric paths inside an

IC to compare path delays. The limitation of this method is that it is not easy to find symmetric paths in large ICs. Another golden model free technique is the analysis of the electromagnetic (EM) in frequency domain [11]. In this research, the golden model is acquired by simulating Register Transfer Level (RTL) codes of the original design. Then simulated EM spectrum is compared with the chip's actual spectrum from experiment to detect HT. To sum up, Table I lists the surveyed SCA methods for HT detection. In the next part, we will propose an implementation of SCA method by using delay path analysis in FPGA platform.

TABLE I. OVERVIEW OF SCA-BASED HT DETECTION APPROACHES.

Work	Side-channel information	Golden chip requirement	Benchmarks	Trojan model	Trojan size (HT/ Main design)
Agrawal et al. [8]	Dynamic power	Yes	RSA	16 bit counter; 8 bit sequential comparator; 3 bit combination comparator	4.4% 0.12% 0.01%
Yoshimizu et al. [6]	Parth delay	No	ISCAS-85	NAND gates	0.625%
Narasimhan et al. [4]	Fmax and dynamic current	Yes	AES IEU	24-FF sequential form; 10-FF; 3-FF; 8 bit comparator	1.10% 0.40% 0.11% 0.04%
Hossain et al. [10]	Dynamic power	No	ISCA89 ITC AES-128	32 bit comparator; 32 bit counter; 2 NAND & 1 AND gate	0.023% 0.023% 0.023%
He et al. [11]	Electromagnetic	No	AES	Trust-HUB Trojan repository	small
Ngo et al. [5]	Delay EM	Yes Yes	AES AES	Input state detection (combinational) 32 bit counter	0.5% 0.94%

IV. PATH DELAY BASED HT DETECTION

This section presents a SCA-based HT detection method using path delay analysis. In [7], the authors have built the fingerprint of the IC by collecting the path delay information so that it can achieve the detection probability of 36% with the HT size of 0.36% of the main design. However, this technique is only suitable for explicit HT. Moreover, it requires the measurement of all paths in the designs. Another approach proposed in [12] focuses on using timing reports generated automatically by EDA tools with certain number of timing paths (k). The results shown that by using 20 paths with the accuracy of 0.01 ns, the detection probability of 80% can be achieved. However, the main drawback of this method is the low flexibility because it uses the reports from Timing Analyzer tool to get delay paths. In addition, these reports only include information about paths from input to output ports. In this section, a method to detect HT based on the difference in distance between points in during signal propagation. Normally, the clock frequency of the system is chosen so as not to generate errors during the working process. However, when the clock is being adjusted in increasing direction, a critical value will be obtained at which the error occurs. Comparing this critical value with the original reference that was tested and stored in the database, if any difference is observed, HT will be detected.

In this work, the main design component is 128-bit AES encryption core. AES is an example of a private key crypto-systems. The cipher takes a plaintext block size of 128 bits, and the key length can be 16, 24, or 32 bytes (128, 192, or 256 bits). The algorithm is referred to as AES-128, AES-192, or AES-256, depending on the key length.

A. *Proposed design for path delay based HT detection*

Figure 3 shows the system block diagram including debug cores and Xilinx ChipScope Pro tools [13] for the proposed technique. We can place ICON, ILA, VIO, and ATC2 cores (as ChipScope Pro cores) into a design by generating the cores with the CORE Generator tool and instantiating them into the HDL source code. The bitstream can be downloaded into the device under test and the design is analyzed with Xilinx ChipScope Pro Analyzer tool.

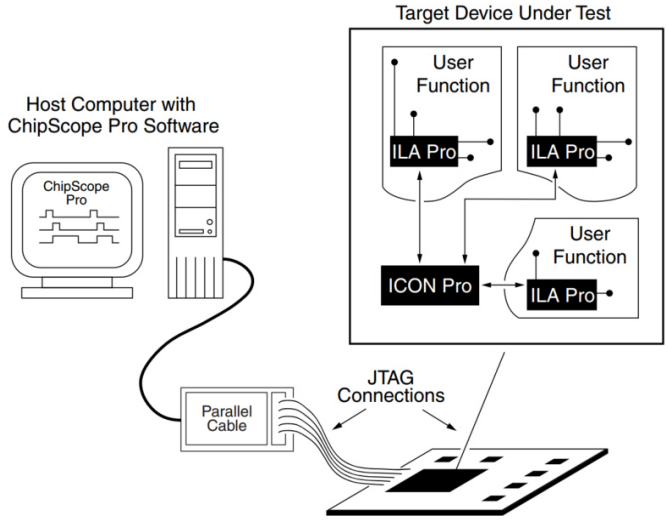

Fig. 3. ChipScope Pro system block diagram [13].

As one of ChipScope Pro cores, ILA (*Integrated Logic Analyzer component*) can be used to monitor any internal signal of a design. The ILA core includes many advanced features of modern logic analyzers, including Boolean trigger equations, trigger sequences, and storage qualification. There is a problem when using ILA with a script because not all the Chipscope Analyzer GUI behavior can be done with Tcl script. ChipScope Engine Tcl Interface provides Tcl scripting access to JTAG download cables using the communication library in the ChipScoppe logic analyzer engine. The purpose of the CSE/Tcl interface is to provide a simple scripting system to access basic JTAG, FPGA, and VIO (virtual input/output) core functions. The Tcl script can perform detecting the Cable, downloading the .bit file, submitting instructions through JTAG interface and VIO core function. But it cannot perform ILA function such as trigger condition setup, data capturing or exporting data. Fig. 4 shows the connections between devices in this method. The aim of this section is to design a new ILA called ILA_tiny with UART interface by VHDL language. ILA_tiny has simple features than original ILA on Xilinx's ChipScope. Block diagram of board-under-test is illustrated in Fig. 5.

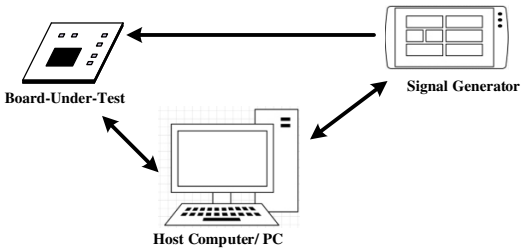

Fig. 4. Connections between devices.

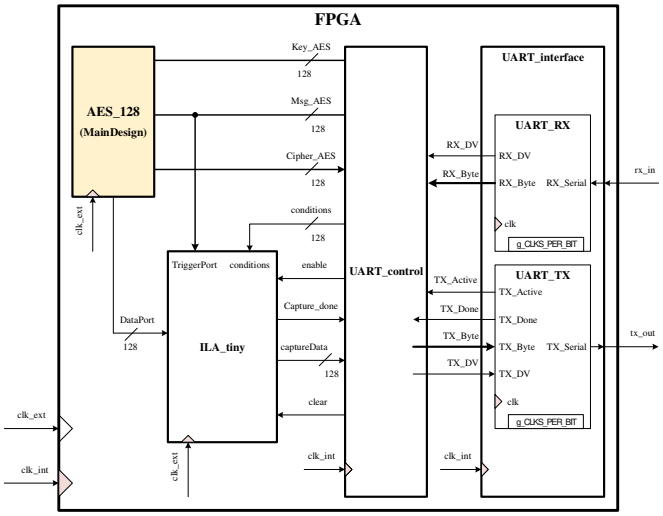

Fig. 5. Block diagram of the proposed HT detection method.

Signals in the FPGA design are connected to the inputs of ILA_tiny, and those signals can be captured at design speeds. Before the design is implemented, select the parameters of the core, including how many signals to capture and how many samples can be captured. Required input signals of ILA_tiny include:

- *Conditions*: $n-1$ bits;
- *TriggerPorts*: $n-1$ bits;
- *DataPorts*: $m-1$ bits ($m, n = 0, 1, ..., 127$).

The *TriggerPorts* input is compared against a set of expected values known as match units in *Conditions*. If the match equations evaluate to true, then a trigger event occurs and data is collected and stored into trace memory.

$$TriggerPorts = Condititons \quad (1)$$

Algorithm of the main program is illustrated in Fig. 6, it is divided into three subprograms, where:

m : total number of bits (or points) to check, in this research $m = 128$;

i : number of checked bits, default value $i = 0$;

j : number of bits is being checked, default value $j = 0$;

f_0 : initial frequency;

Δf_0 : maximum of step frequency, default value Δf=4.096 MHz;

f : instantaneous frequency;

Δf : instantaneous step frequency;

δf : minimum of step frequency, default value Δf=0.016 MHz

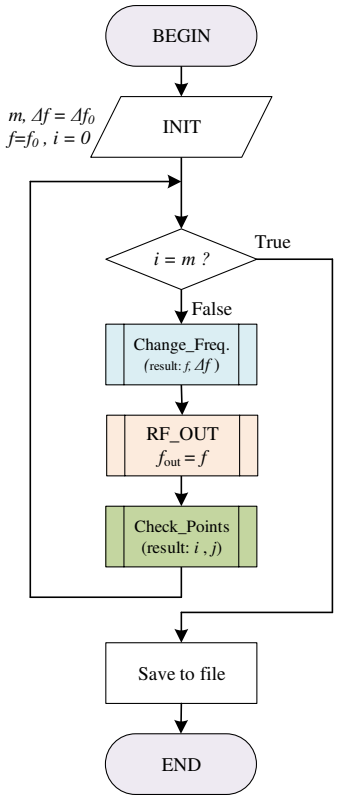

Fig. 6. Algorithm of the main program.

- *Change_Freq* is a subprogram to change the frequency of Signal Generator, determine the pair of values ($f, \Delta f$). It has two processes: *Corse_step* and *Fine_step.*

- *RF_OUT*: this is a program to connect and control parameters on Signal Generator. When the connection is successful, the required parameters from PC will be sent, such as frequency, state, signal level, and so on.

- *Check_Points*: at each frequency, PC sends *capture_en* command to Board_Under_Test, then receives 128 bits of the desired data. This operation is repeated 20 times. Comparing each bit of *capture_data* with reference data that was tested and stored in the database, if there are more than 10 different and the process in *Change_Freq* is *Fine_step*, the number of checked bits will increment. When m bits are checked, the measurement results are saved to the database that will be used for evaluation.

At each measurement, the critical values corresponding are saved. With a mathematical model, this result is represented in the form of a row vector, each element is the frequency corresponding to each bit of S1. To ensure statistical properties, the survey process was carried out in N trials. Finally, the data set of measurement results is presented in the form of a matrix with the size of N×128.

$$\mathbf{f} = \begin{bmatrix} \mathbf{f}_0 \\ \mathbf{f}_1 \\ \cdots \\ \mathbf{f}_{N-1} \end{bmatrix} = \begin{bmatrix} f_{0.0} & f_{0.1} & \cdots & f_{0.127} \\ f_{1.0} & f_{1.1} & \cdots & f_{1.127} \\ \cdots & \cdots & \cdots & \cdots \\ f_{N-1.0} & f_{N-1.1} & \cdots & f_{N-1.127} \end{bmatrix} \quad (2)$$

Where:

$\mathbf{f}_i$: Row vector, its size is 1×128 resulted in *i-th* trial;

$f_{i.j}$: Element in row *i*, column *j*, it is presented critical frequency corresponding to *j-th* bit of S1 in the *i-th* trial.

From (2), the HT can be detected based on the pair of values (μ_j, σ_j) for each bit as:

- Mean value:

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_0 & \mu_1 & \cdots & \mu_{127} \end{bmatrix} \quad (3)$$

$$\mu_j = \frac{1}{N}\sum_{i=0}^{N-1} f_{i.j} \quad (4)$$

- Variance:

$$\boldsymbol{\sigma}^2 = \begin{bmatrix} \sigma_0^2 & \sigma_1^2 & \cdots & \sigma_{127}^2 \end{bmatrix} \quad (5)$$

$$\sigma_j^2 = \frac{1}{N}\sum_{i=0}^{N-1}\left(f_{i.j} - \mu_j\right)^2 \quad (6)$$

The work evaluates the difference in distance between points in one of the rounds. The selected round is random and can be changed. In this research, the first round is evaluated, so input and output signals are S0 and S1, respectively. Msg is selected as the pair of values Msg_0 and Msg_1 corresponding to the output of S1 contains all of bits 0 or all of bits 1 (Table II). Msg_0 is used to set initial value for registers and signals inside AES. For ILA_tiny, the Conditions input has a value equal Msg_1. Thus, when changing Msg_0 to Msg_1, the condition in equation (1) is satisfied. After two periods of the clock, S1 will contain all of bits to 1 which is the desired data *capture_data*. The selected inputs of AES as follows:

```
Key   = "00112233445566778899aabbccddeeff"
Msg_0= "5aa6044e28ec2d1596cae34557eac82c"
Msg_1= "f8a89d615fe23b9a3ca0223df0615106"
```

B. HT Implementation

In order to evaluate the impact of HT in FPGAs, we need to keep the same placement and routing between the golden and HT infected circuits. Hence, the only difference between them is the logic utilized for implementing the HT logic. Chip Planner in Altera Quartus II and Xilinx FPGA Editor in Xilinx ISE/Vivado Suites are two basic tools that can insert HTs without modifying the designed routing. Fig. 6 presents the block diagram of 128-bit AES core [14]-[15] as the original circuit for the Trojan benchmark AES-T1500 in [16].

There are four main steps to implement HT with Xilinx FPGA Editor tool [2]:

1) Perform Synthesize, Translate, Map, Place & Route steps for the original circuit.

2) Extract the Native Circuit Description (NCD) file which contains the logic, placement & routing information of the original circuit as the golden model.

3) Using the FPGA Editor to insert HT in unused LUTs and slices of FPGA with the NCD file, manually or by a script.

4) Generate bit files for both original and HT infected designs with FPGA Editor.

With this method, we can ensure that the placement and routing of the original circuit are the same in both golden and HT-infected circuit. We explain how to add HT in the third step as follows:

* *Create Trigger component of HT:*

- Randomly select an unused LUT, denoted by LUT_A;

- Select signals related to Round 1, assume that two selected signals are net_1 and net_2. These nets are routed to in_1 and in_2 of LUT_A;

- Change the function of LUT_A so that HT is not activated.

TABLE II. VALUE OF EACH TRANSFORMATION IN ROUND 1.

State	Use Msg_0	Use Msg_0
Msg (Initial state)	5a 28 96 57 a6 ec ca ea 04 2d e3 c8 4e 15 45 2c	f8 5f 3c f0 a8 e2 a0 61 9d 3b 22 51 61 9a 3d 06
Key (Initial round key)	00 44 88 cc 11 55 99 dd 22 66 aa ee 33 77 bb ff	00 44 88 cc 11 55 99 dd 22 66 aa ee 33 77 bb ff
S0 (State at start of Round 1)	5a 6c 1e 9b b7 b9 53 37 26 4b 49 26 7d 62 fe d3	f8 1b b4 3c b9 b7 39 bc bf 5d 88 bf 52 ed 86 f9
After SubBytes	be 50 72 14 a9 56 ed 9a f7 b3 3b f7 ff aa bb 66	41 af 8d eb 56 a9 12 65 08 4c c4 08 00 55 44 99
After ShiftRows	be 50 72 14 56 ed 9a a9 3b f7 f7 b3 66 ff aa bb	41 af 8d eb a9 12 65 56 c4 08 08 4c 99 00 55 44
After MixColumns	c0 84 0c c0 39 6c f5 28 34 52 f8 16 78 0f b4 4b	3f 7b f3 3f c6 93 0a d7 cb ad 07 e9 87 f0 4b b4
AddRoundkey	c0 84 0c c0 39 6c f5 28 34 52 f8 16 78 0f b4 4b	c0 84 0c c0 39 6c f5 28 34 52 f8 16 78 0f b4 4b
S1 (State at start of Round 2)	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff

* *Create Payload component of HT:*

- Randomly select a used LUT in Round 1, denoted by LUT_B. Note that LUT_B has a free pin.

- Connect *out_A* to *in_B*, then changing LUT_B's function.

In this work, two selected nets are S0[126] và S0[125]. There is only an OR gate in LUT_A. From Table II, in_B is always "True" when MSG is either Msg_0 or Msg_1. LUT_B's function is given by:

$$out\_B = f(B) \quad (7)$$

When adding the in_B into LUT_B'pin, its function is modified so that the value of output is not changed. Here, an AND gate is used:

$$out\_B = f(B)\ AND\ in\_B \quad (8)$$

C. HT Detection Results

In this research, the board under test is Sakura-G board and the signal generator is Rohde&Schwarz SMBV100A. The control program is written in Python language. In our implementation, the size of the genuine and infected circuit is 626 and 627 slices, respectively. This information is presented in Xilinx's reports or the number of slices in FPGA Editor. So, we have an infected circuit with HT of size 0.2% of the

original one. Similar to other SCA based detection methods, the experiment's conditions are constant or negligibly changed, such as temperature, the accuracy of frequency, and so on. Figure 7 is the normal distributions of the critical frequencies corresponding to the benchmark circuits S1[0:3] and S1[124:127]. More details are presented in Table III with the case of the genuine (without HT) and infected (with HT) circuits. From (4) and (6), comparing (μ_j, σ_j) in the genuine and HT infected circuit, Trojan will be detected if there is any difference. We can see that a shift of the mean value is very small. In the future, we will improve the proposed method to achieve better results and more detail analysis.

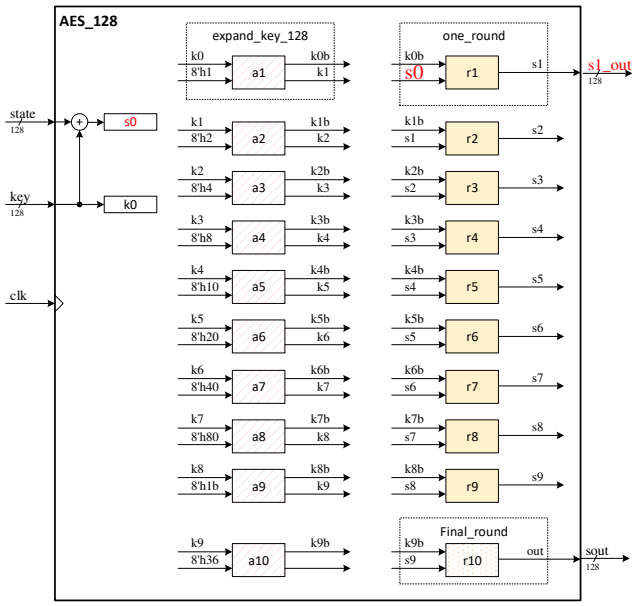

Fig. 6. Block diagram of 128-bit AES core.

TABLE III. CRITICAL FREQUENCIES OF S1[0:1], S1[126:127].

Trial No.	S1[0]		S1[1]		S1[126]		S1[127]	
	W/o HT	With HT	W/o HT	With HT	W/o HT	With HT	W/o HT	With HT
1	416.970	417.513	418.438	418.902	358.297	358.973	418.822	419.349
2	417.225	417.587	418.311	418.960	358.365	358.760	418.820	419.407
3	417.102	417.442	418.444	418.991	358.692	358.986	419.211	419.441
4	417.098	417.472	418.183	419.115	358.303	358.895	418.822	419.435
5	417.095	418.066	418.433	419.329	358.113	359.145	418.943	419.968
6	416.960	417.882	418.492	419.320	358.105	359.073	419.128	419.832
7	417.630	418.002	419.035	419.376	359.137	359.251	419.547	419.823
8	417.789	417.834	419.068	419.110	358.944	359.010	419.707	419.656
9	416.971	417.852	418.265	419.081	358.541	359.107	418.710	419.591
10	417.500	417.404	419.107	418.760	358.443	358.705	419.683	419.401
μ_j	417.234	417.705	418.577	419.094	358.494	358.990	419.139	419.590
σ_j	0.282	0.234	0.334	0.189	0.322	0.158	0.361	0.207

V. CONCLUSION

This paper has given an overview about HT detection methods using side channel analysis and presented a path delay based SCA method. The preliminary hardware implementation results in FPGA platform have clarified the feasibility of the proposed method. In the future work, we will improve the proposed method to achieve better results with more detail analysis and develop a reference free HT detection for a more applicable solution.

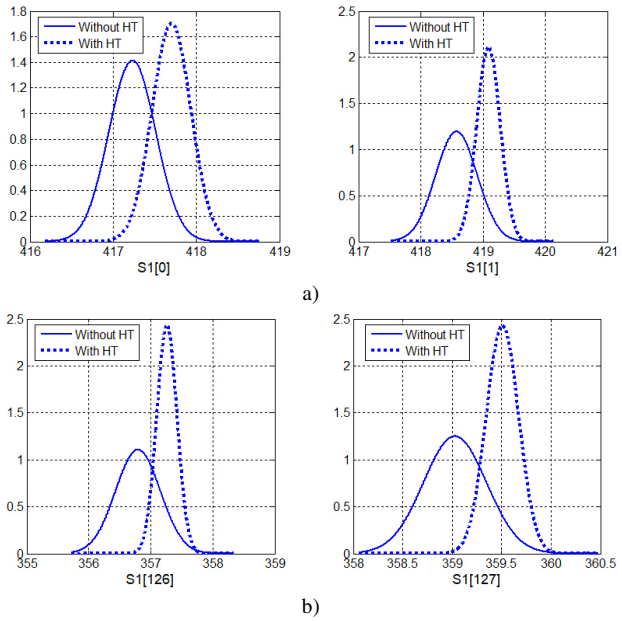

Fig. 7. Distributions of the critical frequencies corresponding to S1[0:1] (a) and S1[126:127] (b).

REFERENCES

[1] S.Bhunia et al., "Hardware Trojan Attacks: Threat Analysis and Countermeasures," Proceedings of IEEE, vol.102, pp.1229-1247, 2014.

[2] Xuan Thuy Ngo, *Prevention and Detection of Hardware Trojan in Integrated Circuits*, PhD Thesis, Telecom ParisTech, 2016.

[3] Jim Aarestad et al., "Detecting Trojans Through Leakage Current Analysis Using Multiple Supply Pad I_{DDQ}s," in IEEE Trans. Information Forensics and Security, vol. 5, no. 4, pp. 893-904, Dec. 2010.

[4] Seetharam Narasimhan, Dongdong Du et al., "Hardware Trojan Detection by Multiple-Parameter Side-Channel Analysis," IEEE Transactions on Computers, vol. 62, no. 11, pp. 2183-2195, Nov. 2013.

[5] X-T. Ngo et al., "Hardware Trojan detection by delay and electromagnetic measurements," 2015 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 782-787, Grenoble, 2015.

[6] Norimasa Yoshimizu, "Hardware trojan detection by symmetry breaking in path delays," 2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 107-111, Arlington, VA, 2014.

[7] Y. Jin and Y. Makris, "Hardware Trojan detection using path delay fingerprint," IEEE Int. Workshop Hardware-Oriented Security and Trust, 2008, pp. 51-57, IEEE, 2008.

[8] Dakshi Agrawal et al., "Trojan Detection using IC Fingerprinting," IEEE Symposium on Security and Privacy (SP '07), pp. 296-310, 2007.

[9] O. Soll et al., "EM-based detection of hardware trojans on FPGA," IEEE Int. Symp. On Hardware-Oriented Security and Trust, pp. 84-87, 2014.

[10] Fakir Sharif Hossain et al. "Detecting hardware Trojans without a Golden IC through clock-tree defined circuit partitions," 22nd IEEE European Test Symposium (ETS), pp. 1-6, Limassol, 2017.

[11] Jiaji He et al., "Hardware Trojan Detection Through Chip-Free Electromagnetic Side-Channel Statistical Analysis," in IEEE Trans. VLSI Systems, vol. 25, no. 10, pp. 2939-2948, Oct. 2017.

[12] A. Amelian and S.E. Borujeni, "A Side-Channel Analysis for Hardware Trojan detection based on Path Delay Measurement," Journal of Circuits, Systems, and Computers, vol. 27, no. 9, 2018.

[13] Xilinx, *ChipScope Pro software and Cores*, UG029 (v14.3). Oct. 2012.

[14] V.-P. Hoang, V.-L. Dao and C.-K. Pham, "Design of ultra-low power AES encryption cores with silicon demonstration in SOTB CMOS process," Electronics Letters, vol.53, no.23, pp.1512-1514, Nov. 2017.

[15] Van-Phuc Hoang, Van-Tinh Nguyen, Anh-Thai Nguyen and Cong-Kha Pham, "A Low Power AES-GCM Authenticated Encryption Core in 65nm SOTB CMOS Process," IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS2017), pp.112-115, Aug. 2017.

[16] Trojan Benchmarks, AES-T1500, https://www.trust-hub.org/resource/benchmarks/AES/AES-T1500.zip.