Hardware Trojan Detection Using Path Delay Fingerprint

Yier Jin
Department of Electrical Engineering
Yale University
New Haven, CT 06520
Email: yier.jin@yale.edu

Yiorgos Makris
Department of Electrical Engineering
Yale University
New Haven, CT 06520
Email: yiorgos.makris@yale.edu

***Abstract*—Trusted IC design is a recently emerged topic since fabrication factories are moving worldwide in order to reduce cost. In order to get a low-cost but effective hardware Trojan detection method to complement traditional testing methods, a new behavior-oriented category method is proposed to divide Trojans into two categories: explicit payload Trojan and implicit payload Trojan. This categorization method makes it possible to construct Trojan models and then lower the cost of testing. Path delays of nominal chips are collected to construct a series of fingerprints, each one representing one aspect of the total characteristics of a genuine design. Chips are validated by comparing their delay parameters to the fingerprints. The comparison of path delays makes small Trojan circuits significant from a delay point of view. The experiment's results show that the detection rate on explicit payload Trojans is 100%, while this method should be developed further if used to detect implicit payload Trojans.**

I. Introduction

Trusted Integrated Circuit design is a newly proposed topic due to the progress of globalization and the fast improving IC manufacturing technology. Because of global economic pressures, the development and fabrication of advanced ICs are migrating offshore in order to lower the cost. As a result, the whole IC supply chain once located in one country can be spread globally now. To control all these manufacturing facilities is almost impossible while on the other hand, to compromise the IC supply chain for sensitive commercial and defense applications becomes easier. Also, under the pressure of market requirements, auto-placement and auto-routing tools are widely used in modern IC design to deal million-gate level circuits in order to reduce product developing cycle time. These tools, however, are not optimal and leave plenty of chip space unused. Based on the advanced IC manufacturing technology, it is much easier for attackers to embed some malicious circuits, so-called Trojan circuits, in the unused space, or other parameters without changing the area of the whole chip.

Traditional function testing is less effective in detecting Trojan circuit for the following reasons, 1) the trigger condition of a Trojan rarely appears, 2) Trojan inputs could be any patterns in the gap between the vast amount of exhaustive input patterns and the relatively small amount of testing patterns actually used, 3) the harm of Trojan circuits may emerge after a long time after chips are implemented. For example, the Trojan can be a series of XOR gates to compare some inner signals with a preset value, a value that will not appear under normal testing patterns. Only if the attacker loads a special test pattern could the Trojan be triggered to do harm to the circuit. It is also very difficult to construct fault models as there are many types of Trojans and it is difficult and unnecessary to construct a faulty model for each type of Trojan. Without the fault model, we cannot develop Trojan detection methods systematically leveraging the powerful EDA tools. In our work, we develop models which can represent most of the Trojan circuits and help us detect these Trojans and construct trusted ICs. Although the destructive reverse-engineering to check the integrity and genuineness of manufactured chips is a useful method to deal with any types of Trojan circuits, it can't guarantee those untested to be Trojan free [1]. Based on the reasons mentioned above, certain agencies have restricted circuit designs for military usage to the factories which have passed certain certifications. But not everyone can afford the high cost to put manufacturers under their control. Furthermore, as the trusted design idea emerges, vendors and consumers of commercial grade cryptographic and security critical hardware have started to pay attention on this topic. For them, cost is the most concerning aspect so they will be the main force to push for a cheap testing method in detecting Trojan circuits.

A lot of research has been done concerning the security of cryptographic IP cores and embedded systems with various design methods and hardware-based approaches. For example, in [2] a root-of-trust model together with a security policy was proposed. The authors paid attention on the security of ubiquitous embedded devices at the design methodology level to prevent the system from side-channel attacks. Also, another common approach to implement tamper-resistance is to use a separate secure co-processor module [3]. Other methods to counter probing attacks, side-channel attacks are proposed in [4], [5]. However, all of these methods rely on the trusted manufacture process, i.e., they all assumed that the fabrication process won't be compromised.

In the FPGA design flow, the sensitive design is not exposed to theft and tampering through the manufacturing process but only the base array must be verified. In [6], the author proposed the concept that manufacturing process could be unsafe

1-4244-2401-6/08/$20.00©2008 IEEE

but unable to give efficient methods to solve it except for the destructive way. [1] is a significant milestone in the Trojan detection field. It analyzed the common behavior of various types of Trojans and demonstrated the feasibility of building effective fingerprints for an IC family to detect Trojan ICs. Noise modeling was used to construct a fingerprint for an IC family. They inserted several types of Trojans in the genuine RSA circuits. The areas of Trojans are from 1.4% down to 0.01% of the size of the main circuit. Also, they modeled three sets of process variations by creating random variations in the cell libraries. Since the power consumed by Trojan circuit is insignificant compared to the total power consumed by the whole chip, simple side-channel analysis proved inadequate in detecting Trojans. Karhunen-Loeve (KL) expansion was then used to separate the randomness and the time-variation of a random process. This method is useful in detecting Trojan when the Trojan circuits is relatively large compared to the whole chip area and the process variation is relatively low. However, in the deep-micron process, the process variation could be as large as $\pm 7.5\%$ and in complicated designs, the area size percentage of Trojan occupied is low. These small Trojans can be too small to be detected through this method.

In [7], the authors gave an alternative view of triggering the Trojan by adding rare input patterns to the normal input patterns. But they assumed that attackers would only use the most rarely happened event. This assumption is not so convincing since the gap between large amount of exhaustive input patterns and limited test patterns still exists.

The limitation on detecting small-size Trojan by power fingerprint reflects that the power fingerprint is too vague to represent the whole characteristics of a large circuit design. In this paper, we propose a new fingerprint generating method using path delay information of the whole chip. There are many delay paths in a chip and each one represents one part of the whole characteristic of the entire chip. Totally there are a series of path delay fingerprints. No matter how small the Trojan is compared to the whole size of the chip, it can be significant in the path view and may be detected. Also, we define a Trojan fault model and make the detection process systematic.

The basic procedure of our Trojan detection method is similar to that in [1]. In our experiments, however, we concentrate on Trojan detection under manufacturing process with high process variation and the path fingerprints used to characterize genuine design are more elaborate than power traces, so the number of genuine chips that we need to generate our fingerprints are more than those in [1]. This whole testing procedure includes three steps:

Path delay gathering of nominal chips. In this step, many chips are selected from an IC design. High coverage input patterns are then run in these sample chips and high-dimension path delay information is collected. After that, these chips are checked under reverse-engineering to ensure they are genuine circuits.

Fingerprint generation. According to the path delays, a series of delay fingerprints are generated and mapped to low-dimension spaces.

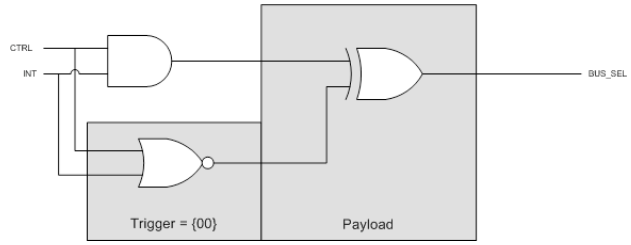

Fig. 1. Combinational Trojan Architecture

Trojan Detection. All other chips are operated under the same test patterns. Their delay information is reduced to low-dimension and compared to the delay fingerprints.

The rest of paper is organized as follows: Section 2 introduces the Trojan circuit architecture and gives a new categorization method to describe different types of Trojans. Based on this classification method and the testing procedure, the experimental setup steps are introduced in Section 3. Section 4 presents our experimental results. Finally conclusions are drawn in Section 5.

II. Trojan Examples and Category

There are many ways to categorize Trojans mostly based on their behaviors. Some Trojans are triggered by rarely occurring events, others are awaken after a preset time delay. Some Trojans propagate internal signals to output pins and these signals can disclose secret information to attackers. Others may make the design malfunction or, even worse, destroy the whole chip. Some Trojans only contain combinational circuits while others could have sequential circuits. In general, however, most Trojans contain two basic parts: trigger and payload. Figure 1 shows a simple combinational Trojan architecture. The trigger part monitors a control signal and an undefined instruction interrupt signal. When these two signals are both in low voltage levels, the payload part of the Trojan is activated and reverses the bus selection signal to select the wrong signals in the inner computation. Since undefined instruction interruption will not occur frequently if we use a well-designed compiler, especially in the testing mode, we will find that even a simple Trojan cannot be detected easily, not to mention other more sophisticated Trojans. Figure 2 shows another example of sequential Trojan whose trigger part contains a k-bit counter. The Trojan won't do harm to the design even if the rarely occurrence event is accidently fulfilled. The same rare event should occur for 2^k times until payload changes the read signal. Probably only attackers can construct the special input pattern sequence by purpose.

Here we give a new categorization method according to how the payload part of a Trojan works: explicit payload Trojan and implicit payload Trojan. The explicit payload Trojan works under a typical two-phase manner: trigger and propagated payload. The sample Trojans in Figure 1 and Figure 2 are both of this type. When the Trojan is triggered, the payload part will alter the value of internal control signals or data signals and cause the chip to perform erroneous behavior or propagate secret information, such as symmetric keys, to some

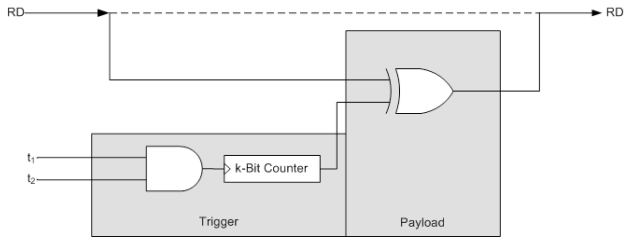

Fig. 2. Sequential Trojan Architecure

output pins. This type of Trojan will insert extra delay in some paths passing those signals.

The implicit payload Trojan has similar trigger part as the explicit payload Trojan but different payload working mechanics. The implicit payload Trojan does not compromise internal signals but only takes these signals as stimulus of the trigger. When the Trojan is triggered, the implicit payload part will behave in a different way than it does in explicit Trojans. The implicit payload may emit radio signals to leak secret information or may destroy the whole chip. It is easy to figure out that the implicit Trojans will make the circuit consume more power and make some path delay larger, since the signal to trigger the Trojan has larger capacity load and becomes slower than usual. But compared to the extra delay inserted by explicit payload Trojan, the added delay here can be smaller and harder to detect.

Also, we can differentiate these two Trojan types through a functional testing view. The explicit payload Trojan can probably be detected in traditional functional test if exhaustive input patterns are run. But functional test cannot detect implicit payload Trojan.

III. Experimental Setup

A. Target Circuit

In this paper, we use a synthesized DES IP core operating under CBC mode for analysis. DES is widely used as secret-key cipher algorithm and its derivative triple-DES is still quite popular in commercial security field nowadays.

The DES design is an area-optimized sequential design. It takes 16 clock cycles to finish a full encryption/descryption cycle based on a mode selection signal. The top module contains two sub-modules performing key generation and the Feistel function f in the DES algorithm. The top module handles initial permutation and inverse initial permutation. These permutations are hardwired with no logical circuits involved to reduce the path delays and lower the area usage. Both plaintext/ciphertext input and output are 64-bit wide so we can read out the ciphertext/plaintext once every 16 clock cycles.

The one-round DES circuit has equivalent area of 3147 NAND gates under 0.13μm, 1.0V technology library *smic13g* of Semiconductor Manufacturing International Corporation. The maximum clock frequency of the design is 180MHz. In the complexity view, the DES circuit used here is not comparable to million-gate SoC. But the Trojan test method can be migrated to high-complexity chips easily although new data analysis methods should be developed in order to reduce the testing cost.

B. Trojan Circuits

Based on our category method mentioned in Section 2, comparator based Trojans and counter based Trojans are generated to represent each type of Trojan. The combinational comparator is used here to compare two inner signals and alters the signal value if there is a match. In the payload phase, an extra XOR delay will be inserted in the path where the Trojan is located. The Trojan counter is simply a 4-bit counter and does no harm to genuine circuits, it only makes them consume more power and increase some path delays since the signal to trigger the counter has larger capacity loads and becomes slower than usual. However, this Trojan counter could be used to disclose secret information inside the chip to hackers, the secret keys, for example.

In our experiments, a total of four Trojans are used. Three of them are comparators and the fourth is a counter. The delay fingerprint, different from the power fingerprint, will be affected not only by the type of Trojan but also from its position in the nominal circuit. These three identical comparators are located near the input, in the middle stage and near the output of the DES circuit, respectively. All of them are 2-bit combinational comparators with an equivalent area of 4 NAND gates occupying only 0.13% of the total circuit area. The fourth Trojan is a 4-bit counter with an equivalent area of 24 NAND gates, which roughly occupies 0.76% of the total circuit size.

C. Circuit Population Generation

Many types of variations will affect the performance of manufactured ICs significantly. For example, across a die, device delays vary due to mask variations, also called the system component of delay variation. There are also random variations in dies across a wafer, and from wafer to wafer, due to process temperature and pressure variations during the various manufacturing steps. The magnitude of delay variation can be 5% in sub-micron processes and even worse in more advanced processes [8]. Even with the same input, the path delays could differ. In our experiments, in order to reflect the process variation, we randomly varied the delay parameters of the SMIC technology library in the range of $\pm$7.5%. This variation represents somewhat the upper bound for current manufacturing processes. We then compile the modified libraries through Synopsys Library Compiler [9] to get compiled libraries.

We use Synopsys Design Compiler Tool [10] to synthesize our DES circuit without Trojan circuits to get the gate level netlist. After that, we modify the netlist by inserting Trojan circuits. This is the step a hacker will probably do when he has access to the manufacturing process but not to the RTL code, which is not provided to the manufacturing factory. The netlists with and without Trojan circuits are then loaded into Synopsys PrimeTime [11] with technology libraries containing random variations. For each specific technology library

#Chip	Pattern 1			Pattern 2			…
	Out1	Out2	…	Out1	Out2	…	…
No.1	4.57	1.65	…	4.81	0.39	…	…
No.2	4.50	1.62	…	4.77	0.42	…	…
No.3	4.58	1.67	…	4.85	0.40	…	…
No.4	4.52	1.67	…	4.81	0.42	…	…
…	…	…	…	…	…	…	…
No.990	4.61	1.68	…	4.9	0.41	…	…

Fig. 3. Chip Path Delay (unit: ns)

(mapped to one chip being manufactured), one Standard Delay Format (SDF) file is generated to give detailed delay information of each gate in the design. The number of SDF files represents the number of chips under test. In all, we generated 990 genuine chips and 800 chips with Trojans, 200 chips for each of the four types of Trojan.

D. Delay Test

Test patterns are the key part in collecting path delay information for both nominal circuits and circuits with Trojans in our experiments. The goal of generating test patterns is to cover as many parts of the whole chip as possible, such that the delay fingerprints will represent the full chip's characteristics and any change in the chip can be detected easily. We used Synopsys TetraMAX ATPG Tool [12] to analyze the netlist and then generate the delay test patterns. Since the fingerprint reflects the nominal chips, only the genuine netlist is loaded into TetraMAX and a total of 163 test patterns are generated to be the basis of our testing of path delays and chip fingerprints. The coverage of the chip corner cases are 100%, i.e., these 163 test patterns can detect every corner case of the DES design.

As we mentioned before, the path delays are used to generate the fingerprint of nominal chips, so every path in the chip should be included to be part of the chip fingerprint. The SDF files generated by PrimeTime are back-annotated into testbenches. All 163 test patterns are simulated to generate the delay information. There are totally 64 outputs of the DES core, and we collect the delay information for each output under these 163 test patterns. Finally, $163 \times 64 = 10432$ path delays are gathered for each chip regardless of whether it is genuine or containing Trojans. Figure 3 shows part of the delay information for nominal chips that we obtained. In this table, each row represents a chip and each column represents the delay of one output under one test pattern. It has a total of 10432 columns.

E. Sample Analysis

1) Multivariate Trending: Since the delay information we collected for each chip contains a high number of dimension(10432), it is necessary to reduce the dimensionality before we deal with the data. One popular multivariate statistical technique is Principle Component Analysis (PCA). The main purpose of using PCA is to find factors that have much lower dimensionality than the original data set to reflect the major trends in the original data sets. The limitation, however, of PCA is that the first few dimensions may not cover adequate information to describe the properties of the whole data set. Also, the PCA method is based on linear correlation between the data of different dimensions. The more linear these variables are, the more efficient PCA will be. So before performing PCA in the whole data set, we divide the 10432-D parameter sets into different groups. Within each group, the linear correlations are higher than data between different groups.

2) Convex Hull: The convex hull of a set of points is the smallest convex set that contains the entire points. Quickhull algorithm is widely used for its computation speed and efficiency [13].

In the field of Trojan detection, numerous types of Trojans cause totally different behavior of the chip performances. Thus, we cannot figure out faulty models for each Trojan, due to their large numbers. The only thing we can rely on is the parameters of nominal chips, called the fingerprint. In our experiments the Quickhull algorithm is used to construct the convex hull of nominal chips and we believe that those chips whose parameter set is located in or near the surface of convex hull should be more reliable than others whose parameter set is located far away from the convex hull.

IV. Experiment Results

A. Construction of Convex hull

In this preparation step, we generated 990 new libraries by introducing $\pm 7.5\%$ random variations in the parameters on each library. Based on these libraries, 990 SDF files with delay information for each chip were generated and then back-annotated to the simulation platform. The path delay information is shown in Figure 3. In order to use PCA more efficiently, we first divide the Table into highly related groups, each group containing one output under different patterns. So, a total of 64 groups is considered in our experiments to generate the final chip fingerprints.

For the dimensionality selection, we observed that four or more dimensions will cause most of the points to migrate to the boundary of the generated convex hull, so we choose to reduce each group to three-dimension. Finally, for the parameter sets of nominal chips, 64 convex hulls are generated, each of which reflects one aspect of the whole fingerprint of a genuine chip.

B. Experiment 1: 2-bit comparator near input

In this experiment, we inserted the 2-bit comparator-based Trojan near the input of the DES core. That is, the Trojan is inserted in one of the input pin. As we mentioned before, the 2-bit comparator only occupies 0.13% of the whole chip area. We generated 200 new libraries by introducing $\pm 7.5\%$ random variations in the library parameters on each library. Then we generated 200 SDF files for circuits with Trojan. We conducted delay simulation and obtained 200 parameter sets which were transformed to principle components by multiplying with the loading matrix. The first three components were selected as coordinates in the 3-D fingerprint space.

Figure 4 shows one of the 64 fingerprint spaces where the convex hull and the test points (each test point represents a

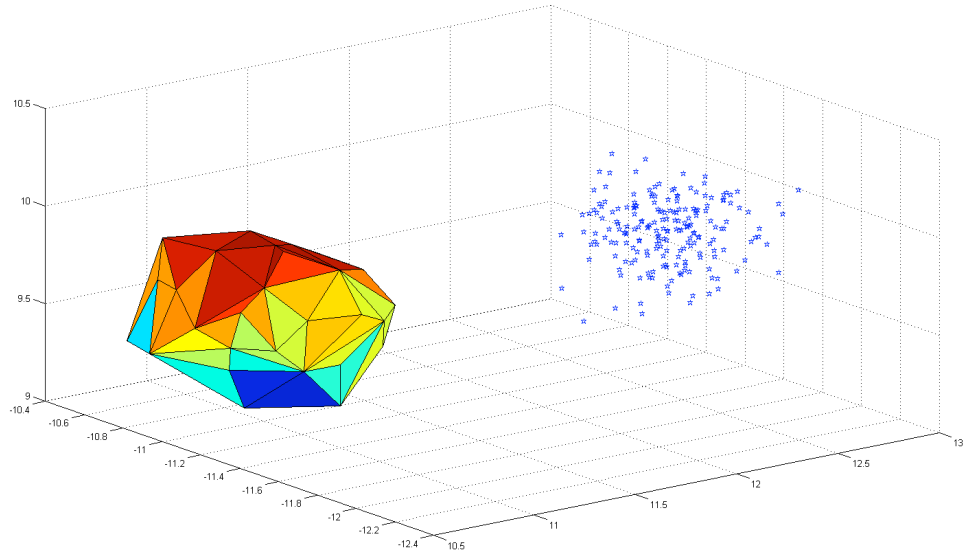

Fig. 4. Delay Comparison of Experiment 1

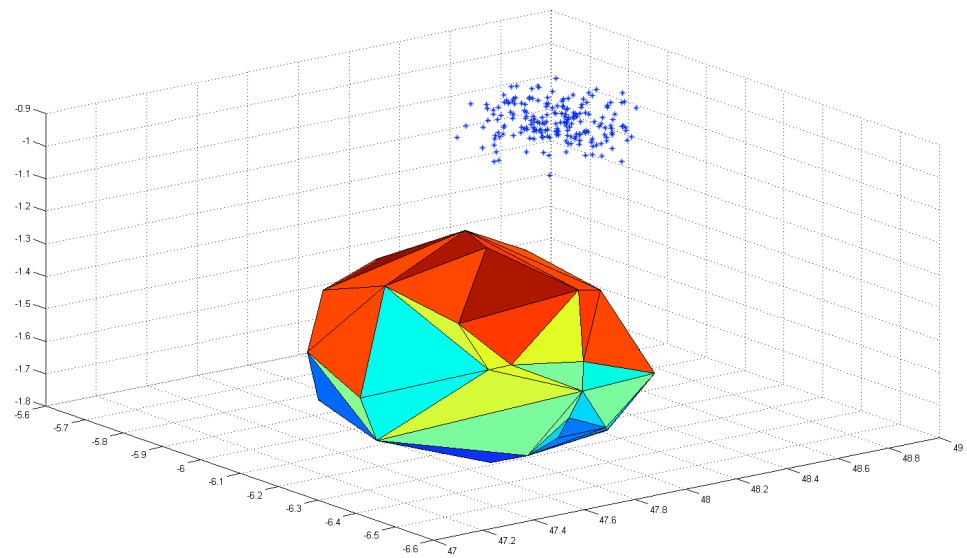

Fig. 5. Delay Comparison of Experiment 2

chip under test) are clearly separated. It is clear that in most of these 64 fingerprint spaces, the test points are near or in the surface of the convex hull. We can, however, denote chips with Trojans by one or several spaces where the sample points are far away from the convex hull. Only those chips whose sample points are in or near the convex hull in the entire 64 fingerprint spaces will be considered genuine.

From Figure 4, we find that it is easy to detect the Trojan through our method. The final analysis shows the detection rate is 100%.

C. Experiment 2: 2-bit comparator in middle stage

In this experiment, we added the 2-bit comparator-based Trojan in the middle stage of the DES chip. An inner signal wire was cut and replaced by the output of Trojan circuit. According to the same process as the first experiment, 200 new libraries by introducing $\pm 7.5\%$ random variations in the library parameters and 200 SDF files are generated. We then conducted delay simulation and obtained 200 parameter sets which are transformed to principle components by multiplying with the loading matrix. The first three components are

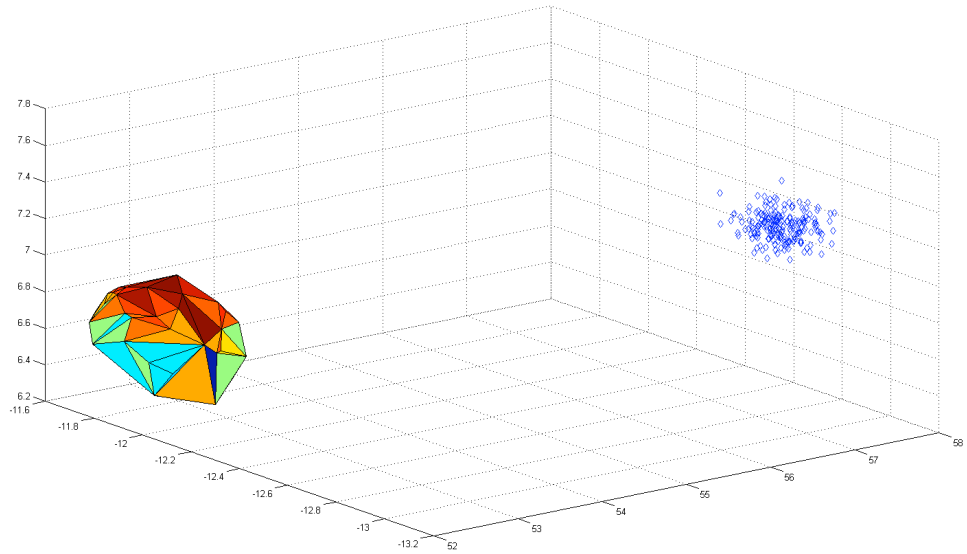

Fig. 6. Delay Comparison of Experiment 3

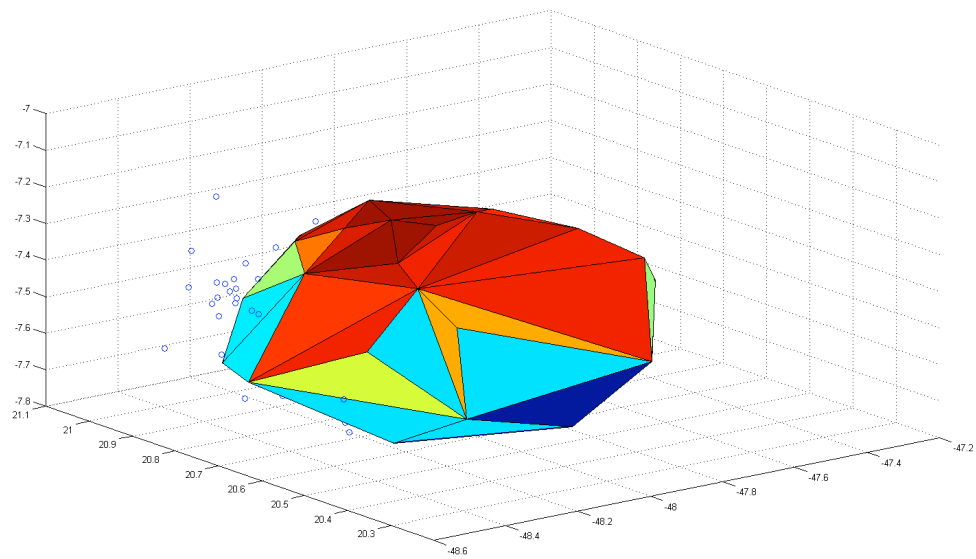

Fig. 7. Delay Comparison of Experiment 4

selected as coordinates in the 3-D space.

Figure 5 shows one of the 64 fingerprint spaces where the convex hull and the test points (each test point represent a chip under test) are clearly separated.

The ratio to detect Trojan circuits in our delay test method is 100% according to the results.

D. Experiment 3: 2-bit comparator near output

In this experiment, we added the 2-bit comparator based Trojan near the output of the DES core with one of the primary output pin compromised by the Trojan circuit. With a similar procedure, 64 fingerprint spaces containing both convex hulls and testing chip parameter sets are obtained.

Figure 6 shows one of the 64 fingerprint spaces where the convex hull and the test points (each test point represent a chip under test) are clearly separated.

From Figure 6, we find that it is easy to detect the Trojan through our method. The final analysis shows the detection rate is 100%.

E. Experiment 4: 4-bit counter

In this experiment, we added a 4-bit counter based Trojan in the DES core. As opposed to the comparator, this counter-based Trojan does not insert any explicit delay. It only uses one inner signal to trigger the counter. However, extra capacity load of that inner signal does finally affect the path delay. Based on the same procedure, 64 fingerprint spaces with 200 testing data points were generated.

Figure 7 shows one of the 64 fingerprint spaces of the convex hull and the test points (each test point represents a chip under test). The results show that although the counter based Trojan occupies about 0.76% of the whole chip area, its detection becomes relatively difficult compared to comparator-based Trojan since it doesn't insert any explicit delay.

Here the Trojan detection rate is only 36%, much lower than that in detecting explicit payload Trojan.

V. Conclusion

In this paper, we presented an effective way to construct the fingerprint for all delay paths to detect Trojan circuits.

The experimental results show that for explicit payload Trojans, our method is very effective. The detection rate is 100%, even though the Trojan circuits are quite small compared to the whole circuit and the process variation is high.

However, to detect implicit payload Trojans becomes difficult through our method. Other data analysis methods should be used and more chip fingerprints should be generated to detect these Trojans.

References

[1] Dakshi Agrawal, Selcuk Baktir, Deniz Karakoyunlu, Pankaj Rohatgi, and Berk Sunar, "Trojan detection using ic fingerprinting," in *Security and Privacy, 2007. SP '07. IEEE Symposium on*, 2007, pp. 296–310.

[2] I. Verbauwhede and P. Schaumont, "Design methods for security and trust," in *Design, Automation and Test in Europe Conference and Exhibition, 2007. DATE '07*, 2007, pp. 1–6.

[3] IBM Inc., *Secure Cryptographic Coprocessor. (http://www.research.ibm.com/ssd_scop)*.

[4] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smart-card security under the threat of power analysis attacks," *Computers, IEEE Transactions on*, vol. 51, no. 5, pp. 541–552, 2002.

[5] Paul Kocher, Joshua Jaffe, and Benjamin Jun, "Differential power analysis," in *Advances in Cryptology CRYPTO 99*, pp. 789–789.

[6] Steve Trimberger, "Trusted design in fpgas," in *DAC '07: Proceedings of the 44th annual conference on Design automation*, New York, NY, USA, 2007, pp. 5–8, ACM.

[7] Francis Wolff, Chris Papachristou, Swarup Bhunia, and Rajat S. A. Chakraborty Rajat S. Chakraborty, "Towards trojan-free trusted ics: Problem analysis and detection scheme," in *Design, Automation and Test in Europe, 2008. DATE '08*, Chris Papachristou, Ed., 2008, pp. 1362–1365.

[8] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Controlled physical random functions," in *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*, 2002, pp. 149–160.

[9] Synopsys Inc., *Library Compiler User Guide*, Version A-2007.12, December 2007.

[10] Synopsys Inc., *Design Compiler User Guide*, Version X-2005.09.

[11] Synopsys Inc., *PrimeTime User Guide*, Version X-2005.06, June 2005.

[12] Synopsys Inc., *TetraMAX ATPG User Guide*, Version A-2007.12-SP2.

[13] C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, no. 4, pp. 469–483, 1996.