Fall 2021

A Comprehensive Study of the Hardware Trojan and Side-Channel Attacks in Three-Dimensional (3D) Integrated Circuits (ICs)

Zhiming Zhang
University of New Hampshire, Durham

A Comprehensive Study of the Hardware Trojan and Side-Channel Attacks in Three-Dimensional (3D) Integrated Circuits (ICs)

BY

Zhiming Zhang

MS, University of New Hampshire, 2018

DISSERTATION

Submitted to the University of New Hampshire

in Partial Fulfillment of

the Requirements for the Degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

September, 2021

This dissertation has been examined and approved in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical and Computer Engineering by:

Dissertation Director, Qiaoyan Yu, Ph.D.
Professor
Department of Electrical and Computer Engineering

Edward Song, Ph.D.
Associate Professor
Department of Electrical and Computer Engineering

Se Young Yoon, Ph.D.
Associate Professor
Department of Electrical and Computer Engineering

Bingxian Mu, Ph.D.
Assistant Professor
Department of Mechanical Engineering

Emre Salman, Ph.D.
Associate Professor
Department of Electrical and Computer Engineering
Stony Brook University

On August, 2021

Original approval signatures are on file with the University of New Hampshire Graduate School.

ACKNOWLEDGEMENTS

This dissertation is finished with the great support from so many people. I really appreciate all the help I received during this wonderful PhD journey.

First of all, I really want to say thank you to my advisor, Dr. Qiaoyan Yu. I could not complete my PhD program without her guidance and help. I was confused after graduating from my Master's program and it was her encouragement helping me build up the confidence of continuing my research and moving up to a higher level. She spent huge effort on every one of my course works and research publications, and gave me a lot of important career advice. It is a great honor and pleasure to work with her in pursuing my PhD degree.

This dissertation would not be completed without the great help from my dissertation committee: Dr. Qiaoyan Yu, Dr. Edward Song, Dr. Se Young Yoon, Dr. Bingxian Mu and Dr. Emre Salman. Their valuable comments and suggestions for my research works are the key of improving the quality of this dissertation. I really appreciate the effort that every professor spent on serving on my committee.

I also really appreciate my dear great school University of New Hampshire for providing such a lovely campus and expert platform for me to enjoy my study and accomplish my research. Being a wildcat is the most wonderful thing in my life!

I would also say thank you to all the funding sources of my research works: National Science Foundation (NSF), Semiconductor Research Corporation (SRC), and UNH Dissertation Year Fellowship (DYF). Their support is a great encouragement and help for me to accomplish this dissertation.

Last but not least, I really appreciate all my colleagues and classmates for their collab-

orations and suggestions in accomplishing this dissertation. They also give me a wonderful school lift as great friends!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

A Comprehensive Study of the Hardware Trojan and Side-Channel Attacks in Three-Dimensional (3D) Integrated Circuits (ICs)

by

Zhiming Zhang

University of New Hampshire, September, 2021

Three-dimensional (3D) integration is emerging as promising techniques for high-performance and low-power integrated circuit (IC, a.k.a. chip) design. As 3D chips require more manufacturing phases than conventional planar ICs, more fabrication foundries are involved in the supply chain of 3D ICs. Due to the globalized semiconductor business model, the extended IC supply chain could incur more security challenges on maintaining the integrity, confidentiality, and reliability of integrated circuits and systems. In this work, we analyze the potential security threats induced by the integration techniques for 3D ICs and propose effective attack detection and mitigation methods. More specifically, we first propose a comprehensive characterization for 3D hardware Trojans in the 3D stacking structure. Practical experiment based quantitative analyses have been performed to assess the impact of 3D Trojans on computing systems. Our analysis shows that advanced attackers could exploit the limitation of the most recent 3D IC testing standard IEEE Standard 1838 to bypass the tier-level testing and successfully implement a powerful TSV-Trojan in 3D chips. We propose an enhancement for IEEE Standard 1838 to facilitate the Trojan detection on two

neighboring tiers simultaneously. Next, we develop two 3D Trojan detection methods. The proposed frequency-based Trojan-activity identification (FTAI) method can differentiate the frequency changes induced by Trojans from those caused by process variation noise, outperforming the existing time-domain Trojan detection approaches by 38% in Trojan detection rate. Our invariance checking based Trojan detection method leverages the invariance among the 3D communication infrastructure, 3D network-on-chips (NoCs), to tackle the cross-tier 3D hardware Trojans, achieving a Trojan detection rate of over 94%. Furthermore, this work investigates another type of common security threat, side-channel attacks. We first propose to group the supply voltages of different 3D tiers temporally to drive the crypto unit implemented in 3D ICs such that the noise in power distribution network (PDN) can be induced to obfuscate the original power traces and thus mitigates correlation power analysis (CPA) attacks. Furthermore, we study the side-channel attack on the logic locking mechanism in monolithic 3D ICs and propose a logic-cone conjunction (LCC) method and a configuration guideline for the transistor-level logic locking to strengthen its resilience against CPA attacks.

CHAPTER 1

Introduction

1.1 Three-Dimensional (3D) Integration Circuit (IC) Is The Solution of Future Integration

As the semiconductor manufacturing process is approaching the physical limit of silicon, it is difficult to continue the Moore's Law [4]. Innovative integration is one of the ways to achieve "More than Moore" [5]. Three-dimensional (3D) integration emerges as a strong candidate [6], which vertically integrates multiple independently fabricated integrated circuits (ICs) as 3D tiers [7]. The stacked 3D structure can effectively increase the device density. Furthermore, the utilization of through-silicon vias (TSVs) as inter-tier connections reduces the global wire length, thus improving system performance and saving power consumption on global interconnect.

1.2 Security Threats of 3D ICs

However, 3D ICs may bring in unique and new security vulnerabilities [8]. The outsourcing fabrication of individual 3D tiers of the stacked 3D ICs provides malicious foundries a chance to perform malicious hardware modifications, such as hardware Trojan attacks. Moreover, side-channel attacks are a group of big threat to the security and integrity of 3D ICs, too.

1.2.1 Security Threats from Hardware Trojan

Besides the vulnerabilities in the fabrication process of stacked 3D ICs, the special stacking structure leaves attackers more exploration space to build new types of hardware Trojans [9].

Hardware Trojan can be defined as the malicious modifications to the original circuit or the malicious extra logic added to the circuit. It usually aims at either altering the logic function of the circuit or leaking important information. Those new types of hardware Trojans built in 3D ICs are the 3D Trojans that have never been explored so very dangerous. Split manufacturing techniques have been used as a protection mechanism against hardware Trojan attacks which separate a complete design into incomplete portions for multiple foundries, thus thwarting reverse engineering attacks and bringing difficulties to Trojan insertion. Unfortunately, the heuristics of electronic design automation tools could nullify the split manufacturing effort regarding the mitigation of either reverse engineering or hardware Trojan [10, 11].

1.2.2 Security Threats from Side-Channel Attacks

Side-channel attack (SCA) retrieves the secret key applied in a crypographic device by analyzing the side-channel signals (e.g., power, delay, and electromagnetic leakage) gained from the physical implementation of that device. Among various power-based SCAs, correlation power analysis (CPA) attack outperforms simple power analysis (SPA) attack and differential power analysis (DPA) attack [12], receiving more attentions [13, 14]. Existing efforts on CPA attacks and their counteracting techniques are primarily limited in the context of hardware implemented with two-dimensional (2D) ICs. Unfortunately, studies of CPA attack in context of the 3D ICs have not been widely explored yet. Although some surveys [15, 16] envision that SCA in 3D ICs may be more challenging than in 2D ICs, but neither physical experiment nor quantitative analysis is available. Moreover, SCA also becomes an emerging attack to logic locking [17], which can be used in monolithic 3D (M3D) ICs [3] to mitigate intellectual property (IP) piracy attacks, aiming at retrieving the locking key.

1.3 The Main Contributions of This Dissertation

In this dissertation, we present our works on securing 3D ICs from hardware Trojan attacks and side-channel attacks. More specifically, the main contributions of this dissertation are summarized as follows.

1. We propose a high-level 3D hardware Trojan characterization including four representative 3D Trojan models.

2. We implement the proposed cross-tier 3D Trojan model to create a TSV-based Trojan and propose a two-tier activation (T^2A) testing enhancement method for the most recent 3D IC testing standard to detect this new 3D Trojan.

3. A frequency-based Trojan-activity identification method (FTAI) is further proposed to detect the 3D Trojans designed based on our Trojan models.

4. An invariance checking based method, which leverages 3D network-on-chips (NoCs) to tackle 3D Trojans, is further introduced.

5. A temporally varied supply voltage (TVSV) method is proposed which utilizes the internal noise of 3D IC's power distribution network (PDN) to mitigate the CPA attacks in 3D ICs

6. A logic-cone conjunction (LCC) method and a configuration guideline is proposed for the logic locking mechanism used in M3D ICs to improve their CPA resilience.

The rest of the dissertation is organized as follows. Chapter 2 presents the proposed 3D Trojan characterization. The practical example for each Trojan case is provided. Chapter 3 introduces the proposed TSV-based 3D Trojan and the proposed T^2A testing scheme. Chapter 4 introduces the proposed FTAI method to detect 3D Trojans. Chapter 5 demonstrates the proposed invariance checking based 3D Trojan detection method. Chapter 6 introduces the TVSV CPA mitigation method for 3D ICs. Chapter 7 presents the proposed methods

for improving the CPA resilience of M3D IC's logic locking mechanisms. This dissertation is concluded in Chapter 8.

CHAPTER 2

Comprehensive Analysis on Hardware Trojans in 3D ICs: Characterization and Experimental Impact Assessment

2.1 Introduction

Since 2007, hardware Trojans inserted in 2D ICs have been well studied in the literature [18–22]. To facilitate Trojan detection, researchers categorize hardware Trojans based on their distribution, structure, size, and logic type. Depending on the activation mechanism, a hardware Trojan can be classified as internally or externally triggered. Based on how often hardware Trojans are triggered, the work [23] presents three types of Trojans: always-on, combinational condition triggered, and sequential condition triggered. Once the Trojan trigger condition arrives, the Trojan payload will execute the defined malicious operations, such as transmitting confidential information, modifying function, degrading performance, and consuming extra power.

Thanks to the mature models for 2D Trojans, various functional testing and side-channel analysis approaches have been proposed to detect different kinds of hardware Trojans in 2D ICs [19, 24–26]. However, Trojan detection methods for 3D Trojans have not been widely explored yet. One important reason for that is the lack of a well-established 3D Trojan model. Due to the vertical integration of multiple tiers, 3D Trojans appear with different characteristics than 2D Trojans [9]. Thus, the commonly used Trojan detection methods for 2D Trojans may not be effective to protect chips from 3D Trojans.

In this chapter, we introduce four 3D hardware Trojan models. Furthermore, we highlight the difference between 2D and 3D Trojans using architectural comparison and quantitative

assessment with practical implementations. More specifically, the main contributions are summarized as follows.

1. We made the first thorough survey on hardware Trojans in 3D ICs. Security threats and hardware Trojan models reported in existing literature are compared in this chapter.

2. Four representable high-level 3D hardware Trojan cases are characterized. Practical examples for each Trojan model are provided for quantitative analysis. The difference between 2D and 3D Trojans are highlighted in our study.

3. As the thermal issue is prominent in 3D ICs, we designed a thermal-induced 3D hardware Trojan and examined its triggering speed and resilience against Trojan detection in a 3D environment for a pass-code authentication.

4. Multiple FPGA boards were utilized to emulate the multi-tier collaborative hardware Trojans, through which attackers can manipulate the function of the target tier without direct tampering on the victim circuit.

5. We examined the success rate of an existing 2D hardware Trojan detection method in the context of 3D ICs. Our simulation results show that the 2D approach operated in 3D chips is not as effective as it works in the 2D scenario.

The rest of this chapter is organized as follows: Section 2.2 summarizes the security threats and hardware Trojan models for 3D ICs discussed in the existing literature. Section 2.3 proposes comprehensive characterization models for 3D Trojans and their practical implementations. Simulation and emulation results for the 3D Trojans are presented in Section 2.3, too. The effectiveness of a 2D hardware Trojan detection method applied in the scenario of 3D IC is examined in section 2.4. This chapter is concluded in Section 2.5.

Table 2.1: Existing Work on Hardware Trojan in 3D ICs

Existing Work	Threat Model		Trojan Model		
	Threat source	Attackers' access	Trigger	Payload	Location
[27]	Untrusted die foundries	GDSII files	Thermal effect caused transition glitches	No special requirement	Any tiers in 3D ICs
[28]	Untrusted die foundries	GDSII files	Thermal effect caused transition glitches	No special requirement	Middle tier in 3D ICs
[8]	Untrusted interconnect foundries Untrusted single die manufacturers	GDSII files	Thermal effect, Aging effect	Voids leading to DoS Partially filled TSVs	Interposer TSV
[29]	Untrusted interconnect foundries Untrusted single die manufacturers Untrusted unified foundries	GDSII files	Remote circuits, Distributed circuits	Impacts on target's power Impacts on target's delay	TSV Multiple tiers
[30]	Untrusted single die manufacturers	Least critical die	Low-activity nets	Leak key from encryption unit	Trojan in different tiers with encryption unit
[31]	Untrusted assemblers	No legitimate dies	No special requirement	Interrupt normal function, Leak information	Extra Trojan die in 3D ICs stack
[32]	Final bounding foundries	Entire layers	Internal nets	No special requirement	Any tiers in 3D ICs
[33]	Untrusted single die manufacturers	GDSII files	No special requirement	No special requirement	Any tiers in 3D ICs

2.2 Our Survey on Existing Hardware Trojans in 3D Integrated Circuits and Systems

The increased number of dies in 3D ICs and vertical-dimension integration potentially leave more attack surfaces open for adversaries to implement hardware Trojans. As multiple dies are vertically integrated into 3D systems, additional manufacturing steps are needed in 3D IC fabrication flow than in their 2D counterparts. Multiple foundries for dies and vertical interconnects will be involved in the 3D integration. In the current semiconductor business model, more and more chip designs are outsourced for fabrication. As a result, neither all single die fabrication foundries nor vertical interconnect manufacturers are trusted [8, 27–30, 33]. The die-to-die bonding may be performed in an untrusted foundry, too. In Fig. 2.1, we label the possible attack surfaces for 3D Trojan insertion. Trojans can be placed by the single-die manufacturing foundries, independently or cooperatively. Since the bonding foundries have access to all the single dies, they have a more likely-hood to implement a Trojan involving multiple dies.

Based on the existing literature, we categorize the 3D Trojans in Table 2.1, where we highlight the threat model with special emphasis on threat source and attack target. In addition to Trojan trigger and payload mechanisms, we also identify Trojan locations in 3D ICs. From Table 2.1 we can see, the nature of the 3D IC structure creates new opportunities for hardware Trojan design, for instance, thermal-based Trojans and cross-tier Trojans. In

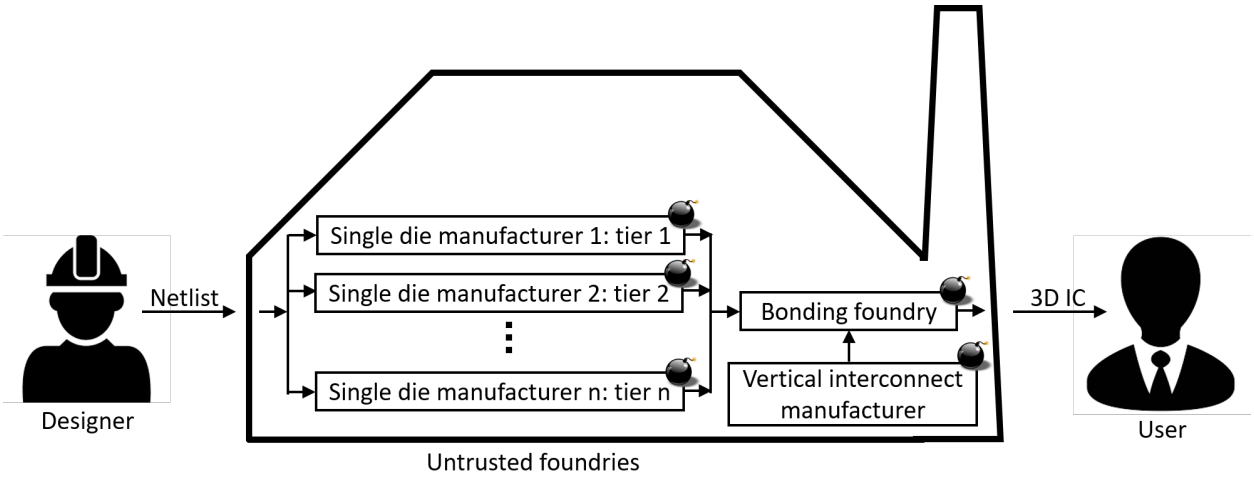


Figure 2.1: 3D hardware Trojan insertion in untrusted foundries.

the next three subsections, we discuss the existing literature listed in Table 2.1 according to their special trigger mechanisms and Trojan locations.

2.2.1 Thermal-triggered 3D Trojans

The fact of poor heat dissipation in a stacked 3D IC can be exploited to develop Trojan triggers. Although the techniques such as heat sink, liquid cooling, thermal-driven floorplanning and routing, and thermal TSV insertion [34] could address the thermal issue in 3D ICs at certain degree, the heat dissipation along a path could harm the tiers and degrade the chip performance [35]. The heat generated and accumulated in the chip will change the electrical parameters of transistors and the switching speed of logic gates. Thus, the system may have new (and unspecified) transition states. The unexpected transition glitches can be employed to design Trojan triggers.

As indicated in [27, 28], thermal-triggered Trojans can be inserted by any malicious foundries with access to the layout of designs. Those Trojans likely congregate near the middle tier, where heat dissipation is harder than in other tiers [28]. The work [8] demonstrates that a thermal triggered Trojan may be hidden in 3D interposers. Thermal Trojans can speed up circuit component aging and consequently lead to a Denial-of-Service (DoS) attack [8].

2.2.2 Cross-Tier 3D Trojans

The multiple-die structure of 3D ICs allows attackers to spread the circuit for a Trojan to multiple tiers. This type of Trojans could be inserted by untrusted die manufacturers, interconnect foundries, and unified foundries. The cross-tier concept means that either the trigger and payload circuits of cross-tier Trojans are separated into different tiers, or the trigger circuit split in multiple tiers is activated jointly to enable the payload [29]. The cross-tier Trojans may not be detected by functional testing performed on each individual die since the Trojan trigger condition is extremely rare. The work [30] demonstrates a Trojan located in a different tier than the encryption unit facilitates to leak the secret key. Even if the untrusted foundry only has partial knowledge of the 3D chip, they can launch cross-tier Trojan attacks.

2.2.3 Trojans Exploiting Other 3D Features

The work [31] envisions a new hardware Trojan in stacked 3D ICs: a malicious die is placed between other tiers in the 3D stack. That malicious die, carrying Trojan circuits, may interrupt normal operations in other 3D tiers or store secret information passing through the Trojan tier. Due to the prominent process variation in 3D chips, it is not easy to differentiate the extra delay induced by the 3D hardware Trojan. This type of Trojan can be inserted by untrusted die assemblers. For instance, the work [32] describes that attackers from the bonding foundry could leverage outsourced dies to implement 3D Trojans. In [33], the adversary is an untrusted die manufacturing foundry with access to GDSII files.

2.3 Proposed Comprehensive Characterization of 3D Hardware Trojans

The existing literature mentioned in Table 2.1 showcases diverse 3D Trojans, but they neither have a thorough discussion on the exact Trojan models nor provide quantitative impact assessment. We provide a solution by characterizing four representable 3D hardware Trojan

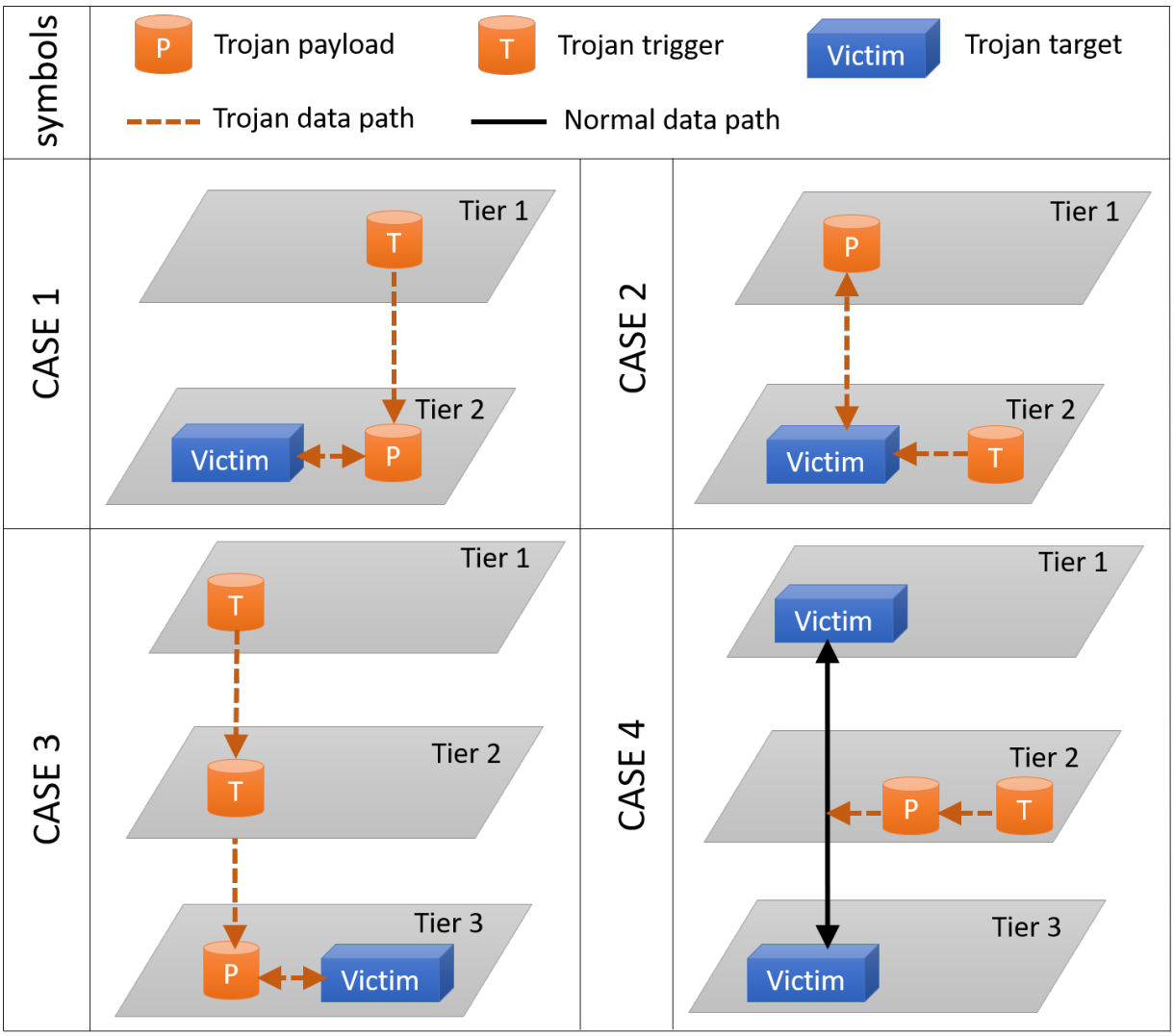


Figure 2.2: Proposed characterization of 3D hardware Trojans.

cases and quantitatively analyzing their practical examples in the following sections.

The major difference between 2D and 3D hardware Trojans is whether or not the Trojan trigger and payload circuits are located in the same tier where the target circuit resides. In 2D chips, the Trojan circuit co-exists with the victim in the same tier. One could perform testing or side-channel analysis to detect the presence of 2D Trojans. In contrast, conventional testing on 3D chips is typically done in a separate fashion. The die for each tier is tested individually before 3D integration. Once the good dies are stacked vertically, limited testing will be performed to detect the defects between die-to-die connections, rather than

extensively examining the correctness of the 3D system's behavior [36].

Based on our survey in Section 2.2, we characterize the 3D hardware Trojan with four cases shown in Fig. 2.2. To the best of our knowledge, this work is the first efforts that introduce comprehensive characterization for 3D hardware Trojans. The following subsections present four 3D Trojan cases in detail.

2.3.1 CASE 1: Cross-Tier Trojan Trigger

Characteristics

In case 1, the trigger circuit of the 3D Trojan is placed in tier 1 while the payload circuit is located near the Trojan target. This type of 3D Trojan is similar to the 2D Trojans that are triggered by an external signal [37], but it is more difficult to mitigate compared to the 2D Trojan. In 2D chips, the passive attack from the external trigger signals can be alleviated by adding shielding material or using unit isolation. In contrast, in 3D ICs, the external attack may be originated from the adjacent tiers, which are not removable after the 3D chip fabrication is completed. As heterogeneous 3D integration emerges, varieties of external trigger mechanisms could be implemented in the other 3D tiers, thus challenging the prevention of 3D Trojans. Moreover, since the payload circuit may never or rarely be enabled without the valid cross-tier trigger signal, the symptom of Trojan attacks will not be observed in typical functional testing. Thus, this type of Trojan is stealthy.

We illustrate the case 1 Trojan with an example shown in Fig. 2.3. The trigger circuit is a heat generator in the top tier. The payload circuit is a temperature-sensitive resistor, which is built in the authentication unit in the middle tier. When the heat from the top tier propagates to the middle tier, the temperature-sensitive resistor could alter the delay of the critical path or cause timing violations, thus resulting in a malfunction of the authentication unit. As reported in [28], the heat from the middle tier of a 3D vertical stacking structure is accumulated easily due to the relatively long dissipation path to the heat sink. Hence, the thermal triggered Trojans will be more likely deployed in 3D integrated circuits and systems

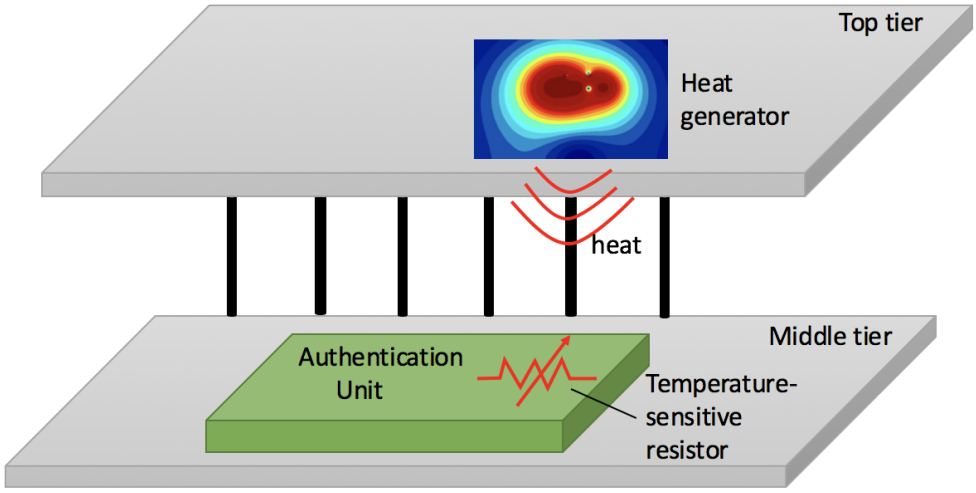
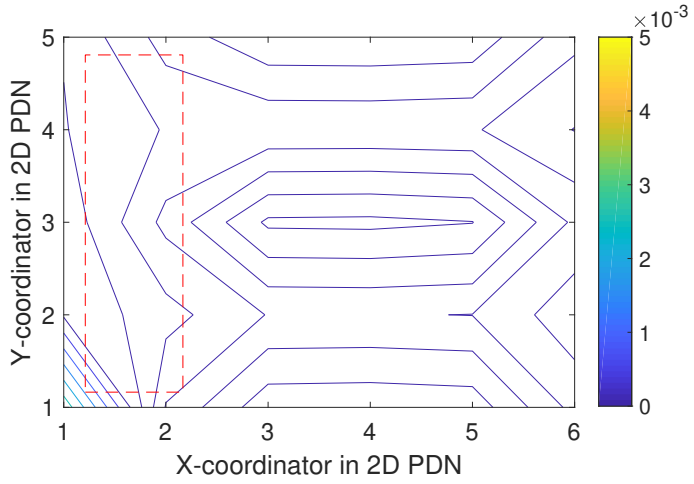


Figure 2.3: Thermal triggered cross-tier Trojan.

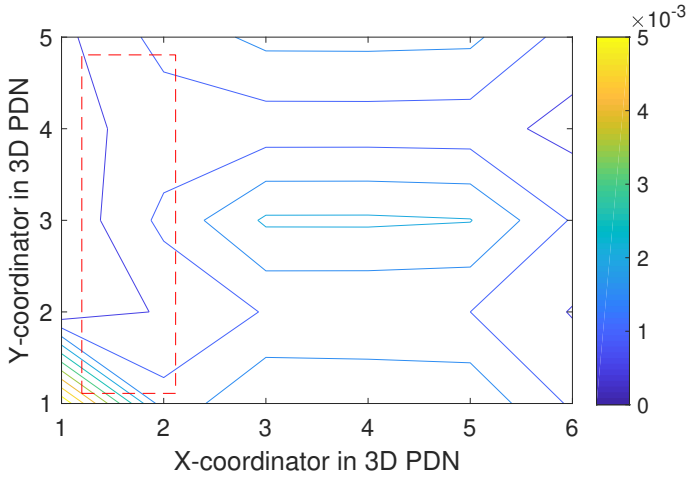
than its 2D counterpart.

We performed a transistor-level simulation in Cadence Virtuoso to demonstrate the impact of middle-tier heat dissipation on neighboring tiers. We collected the transient current of the nodes for load connection in the middle tier of our 3D power distribution network (PDN) model [38] to evaluate the thermal effect. Our target module for the thermal effect investigation is an 8-bit S-box module of AES. In the middle tier, we had 30 load nodes arranged as 5 rows by 6 columns and then captured the current of each node for 10 ns. The current collected in the 8^{th} ns is shown in the contour graphs Fig. 2.4. Generally, the 3D PDN carries greater currents than the 2D PDN. Although the highest current for both 2D and 3D cases appears in the bottom left area where the S-box is located, the current distribution near the S-box is different in the 3D PDN compared to the 2D PDN. We highlight the difference with red dashed rectangles in Figs. 2.4(a) and (b). Those observations make sense because any single tier in the 3D chip is not isolated but impacted by its neighboring tiers. Since the thermal dissipation of a circuit is proportional to its current, it is reasonable to believe that the temperature surrounding our target is influenced by its neighboring tiers.

To perform quantitative analysis for the cross-tier 3D hardware Trojan, we conducted a case study on a platform composed of Xilinx Nexys3 Spartan-6 FPGA, TI MSP430FR6989



(a)



(b)

Figure 2.4: Current contour maps of (a) 2D and (b) 3D PDNs.

LaunchPad board, IRF540 MOSFET transistor, and an NTC thermistor. The purpose of this case study is to verify the implementation feasibility of the thermal Trojan (similar to the one shown in Fig. 2.3) and compare its activation efficiency between the scenarios of 2D and 3D ICs. The overview of our experimental setup is depicted in Fig. 2.5. The main component of the heat generator circuit is a MOSFET driven by the FPGA board. The MOSFET could burn when its gate voltage exceeds a voltage threshold and the MOSFET temperature can be as high as 175°C. The sensor circuit composed of an NTC thermistor and multiple resistors in series is powered by the TI microcontroller. When the thermistor

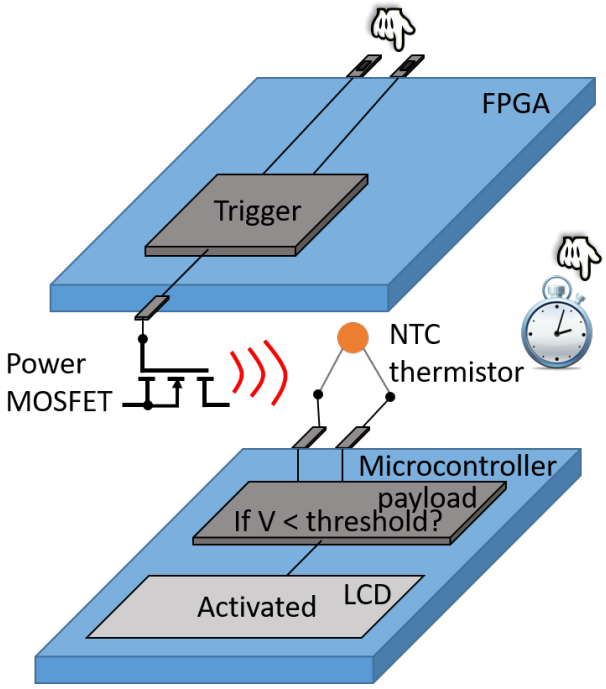


Figure 2.5: Experimental setup for the emulation of thermal-triggered hardware Trojan in 3D ICs.

senses an increase in the temperature in the surrounding air, its resistance starts to drop. This leads to a reduction in the voltage across the thermistor. To emulate the 2D scenario for comparison, we added a heat sink for the heat generator circuit, to provide a better heat dissipation which is commonly available in 2D ICs.

An authentication system is programmed in the microcontroller to examine the password provided externally. The microcontroller also detects the voltage level of the thermistor. A Trojan trigger logic is programmed in the FPGA to monitor the two input signals controlled by the two switches on the FPGA board. The triggered Trojan turns on the MOSFET (thus it starts to burn) to heat the temperature in the surrounding area. Once the thermistor senses the increased temperature, the microcontroller detects the change on voltage and then drives the authentication system to jump to the password reset status, which is usually only available to legal users. We successfully mimicked a 3D thermal-triggered hardware Trojan and overwrote the authentication password in our hardware demo [39].

Next, we compared the activation speed of the thermal-triggered Trojans for 2D and 3D

Table 2.2: Trojan Activation Efficiency.

Emulation scenarios	Time to trigger the Trojan (min)
2D	11:12
3D	6:52

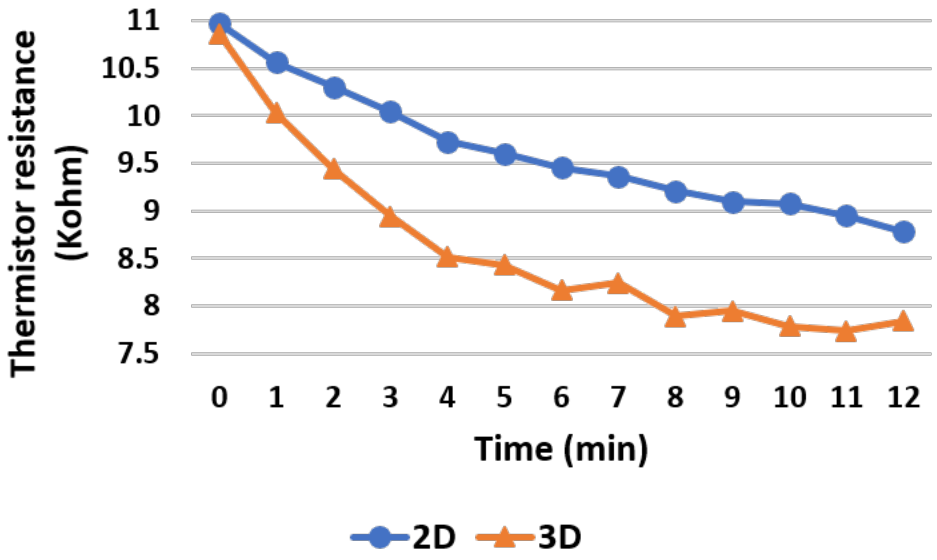


Figure 2.6: Resistance dropping of the thermistor used in Fig. 2.5.

scenarios. We used the microcontroller to implement a threshold comparator to examine the voltage level of the thermistor. If the voltage of a thermistor exceeds the threshold, the Trojan payload will reset the authentication password. We warmed the air surrounding the thermistor with and without the heat sink to mimic 2D and 3D scenarios, respectively. A timer is used to measure the time that the thermistor takes to drop the voltage below the threshold for each case. The results shown in Table 2.2 indicate that the Trojan activation time in the 2D scenario is almost twice compared to the 3D case. This means it is easier to implement thermal-triggered Trojans in 3D ICs than in 2D chips. We also measured the speed of temperature changing, which is reflected in the resistance of the thermistor. The dropping trend of the resistance in Fig. 2.6 implies that the NTC thermistor's resistance for the 3D case drops faster than the 2D. This fact further confirms that heat can be better accumulated in 3D than 2D. Thus, 3D ICs will provide a better environment to facilitate the implementation of thermal-based Trojans than 2D ICs.

2.3.2 CASE 2: Cross-Tier Trojan Payload

Characteristics

In the Trojan described in case 2, the payload is located in the top tier (tier 1), from where it is relatively easy to probe and measure side-channel signals than from the middle tier. The motivation of this type of 3D Trojan is to steal confidential information from the victim unit. Essentially, the stacked structure of 3D ICs provides a reliable medium for attackers to collect information from the middle and bottom tiers. In addition, as the payload resides in another tier, the effect of this kind of Trojans will not be observable while testing on the individual tiers. Here, we assume that the trigger circuit is small enough to hide its area, delay, and power overhead. This assumption is as reasonable as what we usually have in 2D ICs.

The cross-tier Trojan can facilitate the development of a covert channel to leak information. The victim unit could be an encryption engine, such as the one shown in Fig. 2.7. The crypto key is loaded from the volatile memory in the top tier. To prevent the leaked key from being visible during the middle tier testing, the pilfered key is first transformed into another format (i.e., obfuscated key), and then the Trojan passes the obfuscated key to the rarely used main memory in the top tier. When we test the top tier, the main memory functions normally. The separated testing on the middle tier will not reveal the presence of the 3D Trojan because the key is obfuscated. However, the key will be leaked by the covert channel built by the cross-tier 3D Trojan since the attacker knows how to de-obfuscate the key.

Example Analysis

In this subsection, we use a combination of transistor-level simulation and FPGA emulation to demonstrate the feasibility of leaking the AES secret key via cross-tier Trojans. We implemented the cross-tier hardware Trojan and the 3D system shown in Fig. 2.8 in Cadence

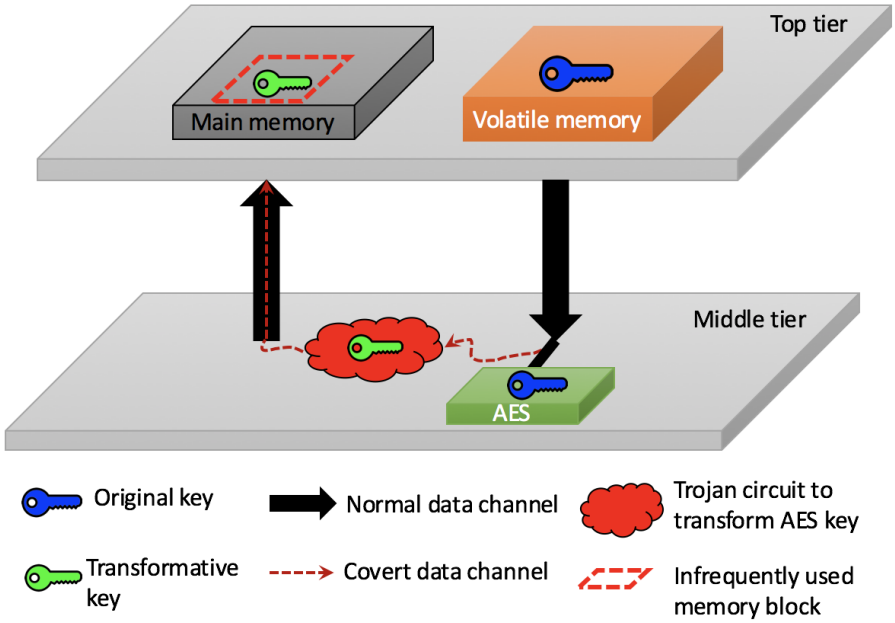


Figure 2.7: An example of key leaking via the covert channel formed by a hardware Trojan in a stacked 3D IC.

Virtuoso with a 45nm NCSU FreePDK technology [40]. The PDN in each tier of the stacked 3D structure is mainly composed of a global power grid and a virtual grid. TSVs connect the global power grids in nearby tiers. The parameters for the TSV and wire model are listed in Table 2.3. The parameters are verified by [40, 41]. Our transistor-level 3D circuit nearly matches the practical 3D IC. The crypto unit adopted here is a transistor-level AES S-box. To ensure the unipolarity of the channel between key and TSV, a buffer is located in the middle of the channel (not shown in the diagram) so that we can prevent the power data from being transmitted back to the S-box to hinder normal operation. The hardware Trojan shown in Fig. 2.8 stealthily passes the secret key to a nearby 3D tier. The main component of the Trojan is a capacitor connected with the PDN. Each key is assigned to one Trojan capacitor. The Trojan capacitors are charged or discharged based on the key bits transmitted through TSVs. The charges stored in the Trojan capacitor C_T will facilitate the side-channel analysis for the crypto key retrieval. The capacitor C_T acts like a decoupling capacitor, which can keep the supply power stable. In this way, the normal function of the nearby tier will not be affected so that the stealthiness of the inserted Trojan can be

Table 2.3: Parameters for TSV and Wire Model

TSV Model (per TSV) [42]					
Diameter	Height	Pitch	Resistance	Inductance	Capacitance
10 μm	60 μm	20 μm	20 mΩ	34.94 pH	283 fF
RC Model for Local Wire Interconnect (per mm) [41]					
Resistance			Capacitance		
3.31 kΩ			170.59 fF		

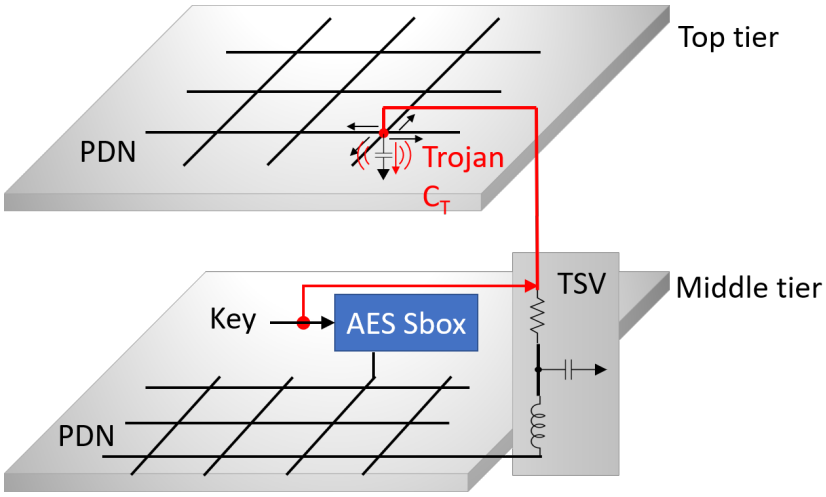
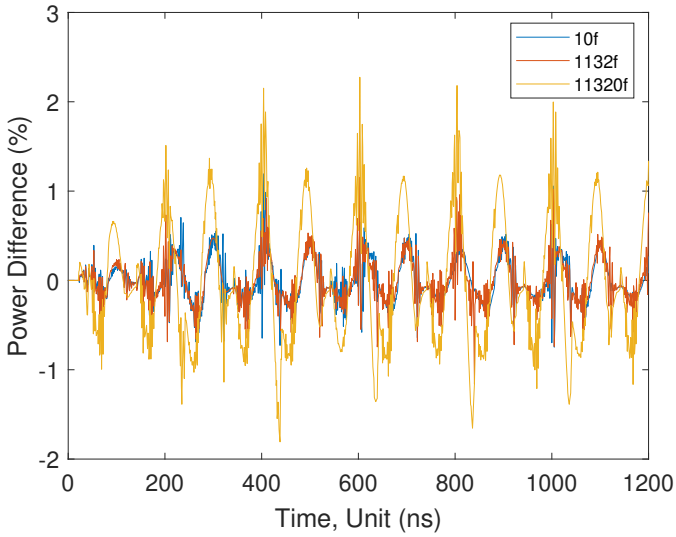


Figure 2.8: Experimental setup of key leaking via a cross-tier Trojan.

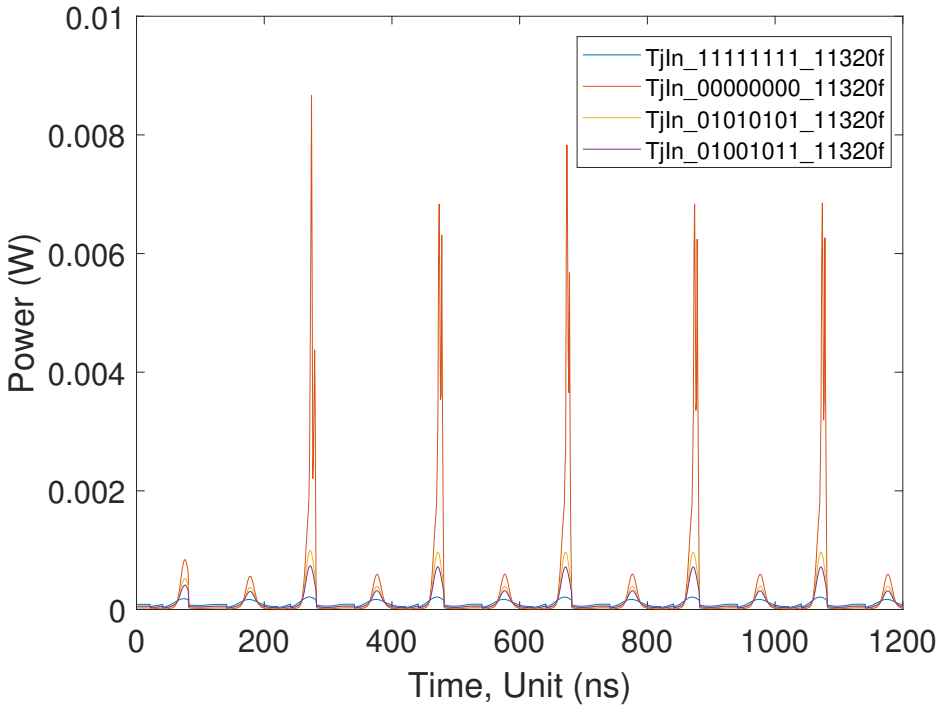
achieved.

In our experiment, we set the key bits to "11111111", and varied C_T from 10fF, 1132fF, to 11320fF. The power consumption of the S-box without Trojan or with different Trojan loads was measured and compared. As shown in Fig. 2.9(a), a smaller Trojan capacitor leads to a smaller power change, but the power difference induced by the Trojan is still less than 2.5% even though we increase C_T to 11320fF. However, the power profiles for different Trojan capacitors are consistent. The slight but consistent variation on the power profile is an important quality to ensure the stealthiness of the cross-tier Trojan. We kept the capacitance of the Trojan as 11320fF but changed the key bits from "11111111", "00000000", "01010101", to "01001011". The power consumption for these four cases is shown in Fig. 2.9(b). It can be observed that the power consumption for each key is unique. Thus, we can correlate the new power profile with the key used in the crypto unit.

Next, we used a SAKURA-G FPGA assessment kit to conduct a side-channel analysis



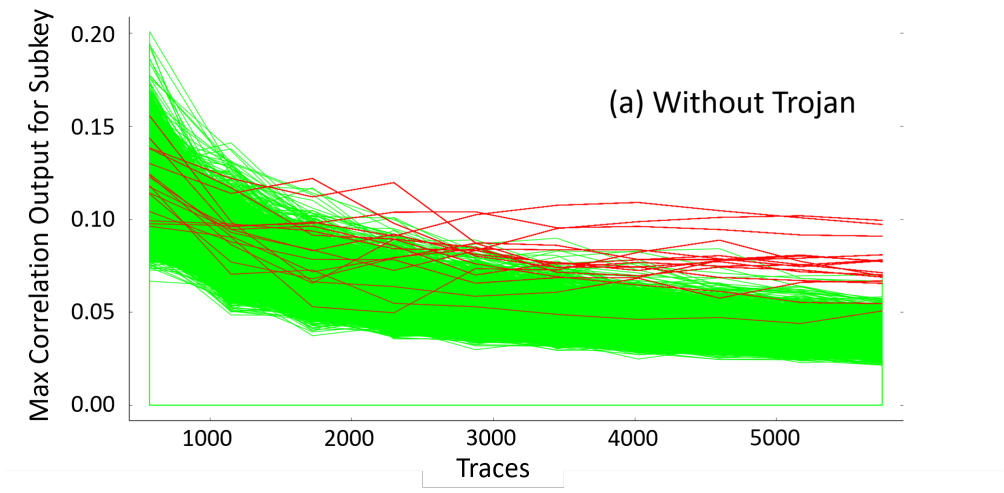
(a)



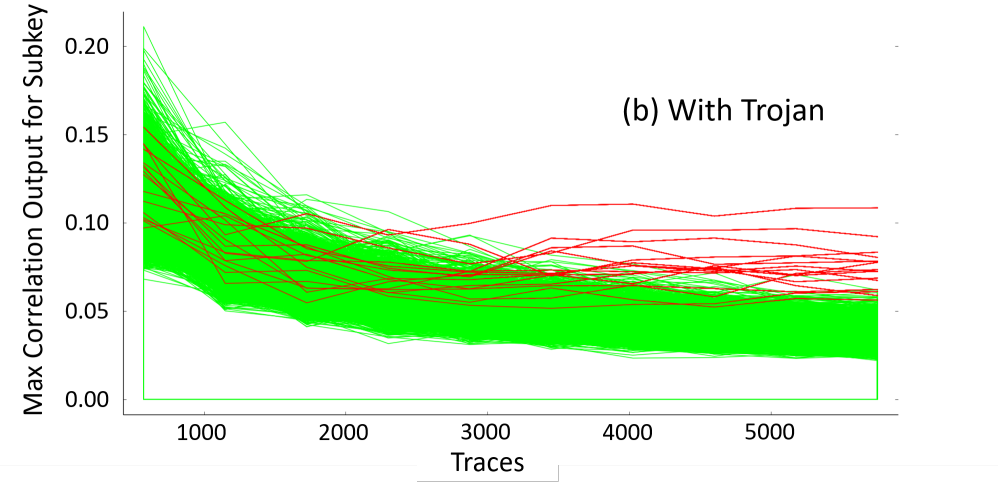
(b)

Figure 2.9: Impact of cross-tier Trojans on the power consumption of an AES S-box. (a) Power differences caused by the Trojans implemented with different Trojan capacitors, and (b) unique power profiles induced by the same Trojan that snoops the AES S-box with different keys.

on an AES affected by the cross-tier Trojan. The Trojan model AES-T1000 published on Trust-hub was modified to mimic the 3D Trojan described in Fig. 2.8. The main difference



(a)



(b)

Figure 2.10: Correlation power analysis for the AES (a) without Trojan and (b) with Trojan.

is, we used FPGA pins to mimic the Trojan capacitors. Each key bit additionally drives eight FPGA pins. Due to the capacitor induced by the Trojan, the total power consumption of the AES module is slightly changed. However, the power difference due to the Trojan accelerates the correlation power analysis (CPA) attack. The key retrieval processes for cases of without Trojan and with Trojan are shown in Fig. 2.10. The red lines represent the 16 key bytes of AES. As the number of analyzed traces increases, the red lines are getting out of the green zone, which means the key bytes are being retrieved. As a result, the CPA attack on the AES with Trojan is able to retrieve all the key bytes within the use of 6000 power

traces. Given the same amount of power traces, the CPA attack without Trojan retrieves only 14 key bytes out of 16 since two lines are still buried in the green zone. This indicates that the Trojan implemented in this example could ease the CPA attack.

2.3.3 CASE 3: Multi-Tier Collaborative Trojan

Characteristics

There may emerge another kind of 3D Trojan, multi-tier collaborative Trojan, which is more sophisticated than the cross-tier Trojan trigger and payload. The multi-tier Trojan in case 3 shown in Fig. 2.2 is activated by the two trigger circuits from tiers 1 and 2, respectively. Compared to hardware Trojans in 2D ICs, the multi-tier Trojan trigger has significantly lower Trojan triggering probability due to a larger pool of trigger signals. Moreover, the collaborative Trojan trigger could be a combination of different trigger mechanisms (e.g., temperature, voltage level, and electromagnetic flux). Multi-tier collaborative Trojans represent the scenario that attackers exploit the security weaknesses of other tiers in the 3D system to breach the target tier with strong security mechanisms, instead of compromising the target tier directly. In terms of cost and effectiveness, multi-tier Trojans are more likely to appear in 3D chips than a single-tier Trojan.

We implemented an example of a multi-tier collaborative Trojan in a 3D system with 4 tiers. Two FPGA boards, each including two FPGA chips, were utilized to emulate the 3D system. The schematic diagram and FPGA setup are shown in Figs. 2.11(a) and (b), respectively. Tiers 1 and 2 are weak in the sense of resistance against hardware Trojan insertion. Thus, two hardware Trojan triggers were placed in those two tiers. The 3D Trojan manipulates the signals passing images from tiers 1 and 2 to tier 3. Due to their low trigger probability, sequential hardware Trojan (SHT) triggers were applied in this example. When the SHT trigger is active, the vertical data communication is compromised such that the valid indication signals vd_a and vd_b will allow improper operands a and b to propagate to tier 3. Consequently, the compromised inputs $\widetilde{vd_a}$ and $\widetilde{vd_b}$ lead the Trojan target circuit to behave

differently ($\widetilde{g}$) than the normal specification (g). Once the valid signals are compromised by the 3D Trojans, the integrity of the images received by tier 3 will be sabotaged. As a result, image-based authentication will fail.

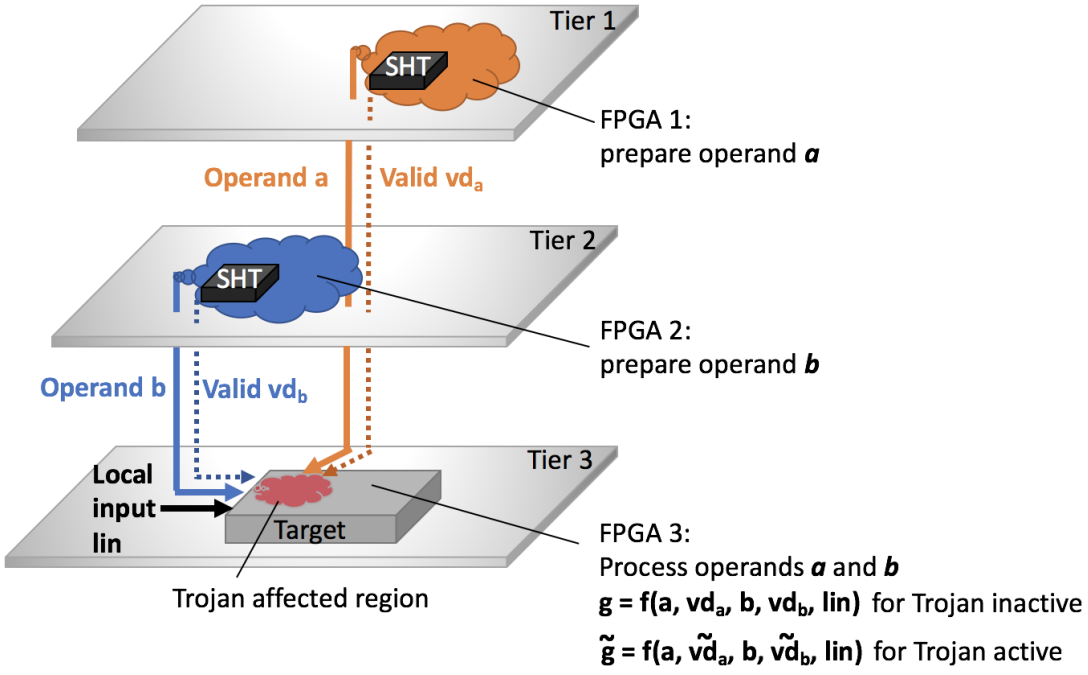
Example Analysis

In the FPGA platform, we connected those FPGA chips with external wires so that the tier-to-tier communication can be manipulated and observed via the oscilloscope. Figure 2.11(c) illustrates that the square-wave signal from tier 1 (the yellow line on the top) is not passed to tier 3 (as the blue signal on the bottom is flat). When the Trojan is triggered, a portion of the yellow line is copied to the blue signal as shown in Fig. 2.11(d). This indicates that the multi-tier collaborative Trojan manipulates the signal filter, which is controlled by the valid signal, and transfers invalid or even malicious data to the target tier. Assume tier 3 in the 3D system examines whether the images from the top two tiers are highly correlated and then enables the critical mission programmed in tier 3. If the valid signals vd_a and vd_b are tampered by the multi-tier collaborative Trojan, dummy image rows will be dumped to tier 3. Five images shown in Fig. 2.12 are adopted for correlation analysis in the 3D system mentioned above. Clearly, Figs. 2.12(b)-(e) are different than Fig. 2.12(a), thus the image correlation cannot get close to 0.9. However, when the valid signals for enabling image transfer between tiers are compromised, the image correlation could approach to 0.9 if the hardware Trojan is able to manipulate vd_a and vd_b for a time period long enough to dump 100 dummy image rows.

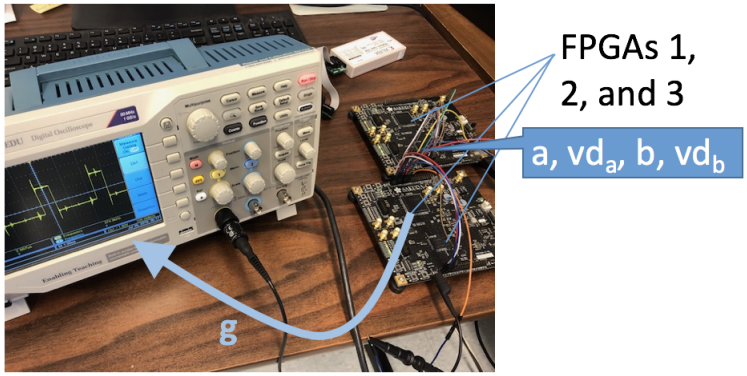
2.3.4 CASE 4: Multi-Tier Synergic Trojan Payload

Characteristics

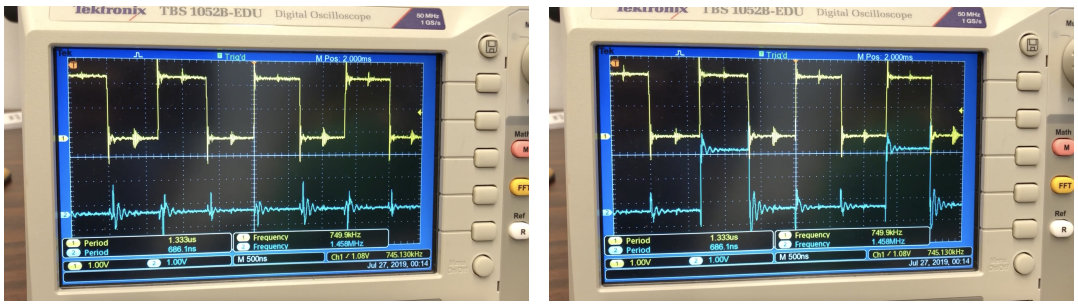
When an IC is expended from planar to vertical dimension, the corresponding Trojan payload will be distributed to multiple tiers as well. In case 4 shown in Fig. 2.2, the Trojan circuit snoops the data (or even the side-channel signal) available in tier 2. As a result, the



(a)

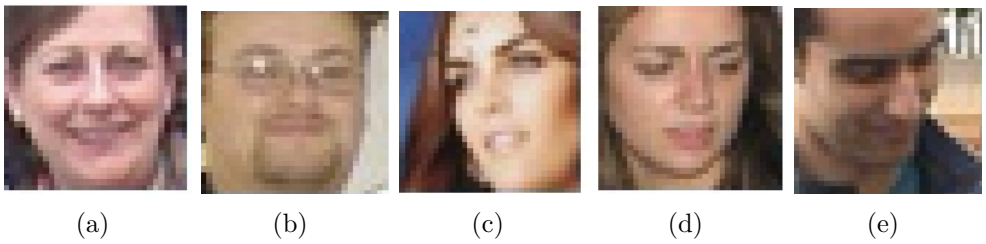


(b)

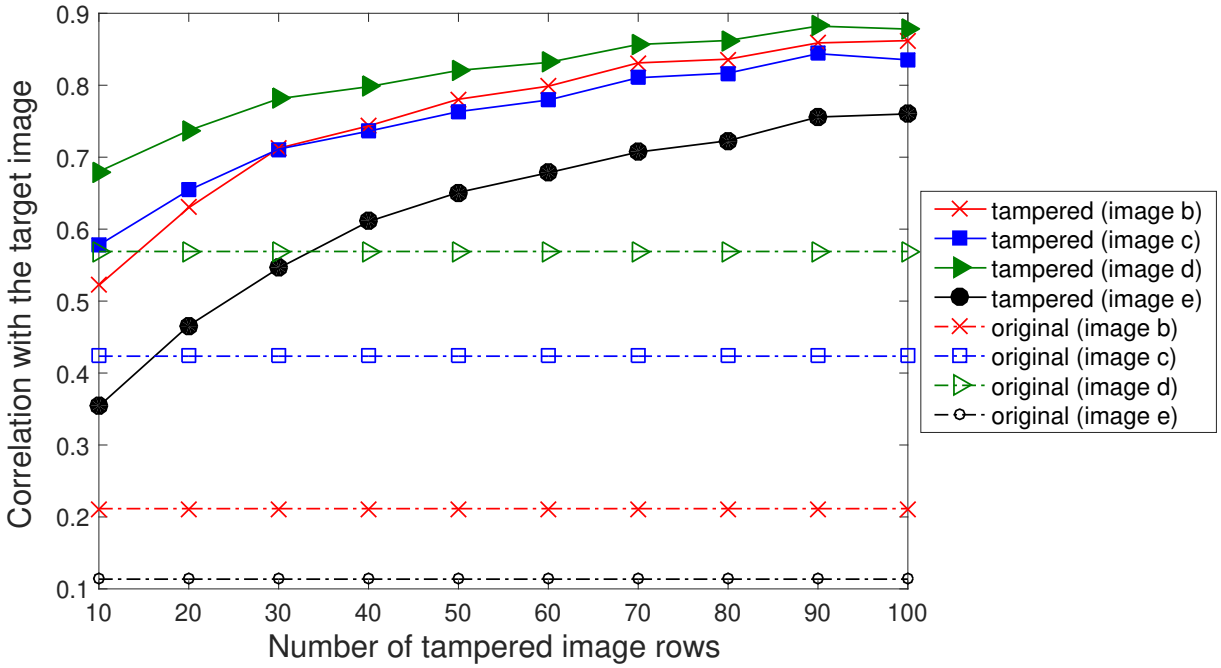


(c) (d)

Figure 2.11: Multi-tier collaborative hardware Trojan. (a) Conceptual diagram, (b) multi-FPGAs experimental setup, (c) normal output, and (d) Trojan affected output.



(a) (b) (c) (d) (e)



(f)

Figure 2.12: Impact of multi-tier collaborative hardware Trojans in an image authentication application. (a) Image generated in tier 1, (b)-(e) Images for comparison provided by tier 2, and (f) correlation analysis results obtained from tier 3.

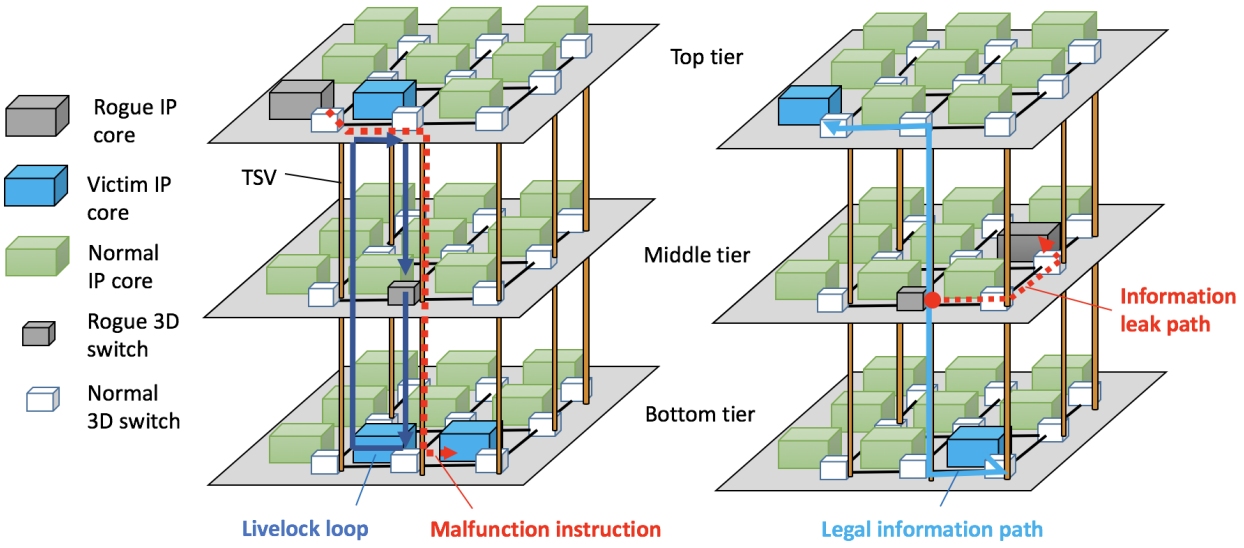


Figure 2.13: Multi-tier synergic hardware Trojan payload causing malfunction, communication livelock, and information leaking.

confidential information is leaked from tier 2 to other tiers. Often time, both the Trojan trigger and payload are located in the different tiers than the target one. Alternatively, a thin Trojan tier can be integrated into the 3D stack structure to provide flexible and precise control on the snooped information without incurring noticeable delay overhead [31]. We further envision that a 3D Trojan payload could achieve a synergic attack effect in multiple tiers, rather than influencing each tier independently. In summary, a multi-tier synergic Trojan has the potential to impact a bigger area than a 2D Trojan. It will be challenging for module-level testing for a subsystem to identify the underlying security threat in the 3D system. The symptom of a synergic Trojan may seem benign from the viewpoint of a small local area. More importantly, the increased impact area of the synergic Trojan payload will make the technique of isolating malicious hardware ineffective or unrealistic since multiple tiers are involved.

Example Analysis

3D network-on-chip (NoC) [43, 44] has been demonstrated as a promising infrastructure to integrate increasing transistors in multiple tiers. 3D NoC eliminates the need for long global

interconnects and reduces the voltage droop and power consumption on long wires. A rogue 2D NoC leads to information leaking and bandwidth depletion [45]. If NoC-based 3D ICs have a synergic Trojan placed in some IP cores or 3D switches, that Trojan leads to a similar consequence, as shown in Fig. 2.13. The rogue IP core sends an NoC instruction packet to the rogue switch. Next, the rogue switch passes that malicious packet to the victim IP core in the bottom tier. As a result, the multi-tier synergic Trojan eventually causes the victim IP core to have malfunctions. Or, the rogue switch in the middle tier could trigger a livelock between the middle and bottom tiers. The proposed multi-tier synergic Trojan is stealthy because the hardware of the rogue IP core and switch has high similarity with the normal ones and the 'rogue' feature is only visible at the arrival time of special NoC packets. Figure 2.13 illustrates another practical example of the case 4 Trojan model. The rogue switch and IP core tampered by a hardware Trojan monitor the special packet transferring through the middle tier and the packet of interest in the rogue IP core is stored for future use and analysis. In the case of passing malicious packets in NoCs, the rogue IP core is the Trojan trigger to initialize the attack by issuing the malicious instructions. The rogue 3D switch is the payload, which causes malfunction by delivering malicious instructions to the victim IP cores. The trigger and payload are from different tiers but none of them is in the same tier where the victim locates. In the case of information leaking, the payload formed by a rogue 3D switch is responsible for leaking NoC packets. Although the trigger and payload for this case are in the same tier, they remotely control the victims in other tiers. The Trojan type proposed in this subsection is non-invasive. Moreover, the snooping attack is hidden in the normal data transmission of the middle tier. Side-channel analysis of the entire system may not be able to detect the presence of such hardware Trojans.

2.4 Examination of A 2D Trojan Detection Approach in 3D IC

The existing Trojan detection methods are mainly designed for the Trojans in 2D ICs. Due to the unique characteristics of 3D Trojans, as analyzed in Section 2.3, they may not work

well in 3D scenarios. Split manufacturing may impact the hardware Trojan insertion in 3D ICs at some level. However, the adversaries in untrusted foundries with partial design details might be able to reverse engineer the whole design. Once the design is recovered, attackers can continue to insert Trojans. On the other hand, split manufacturing is not for securing the stacked 3D ICs in which every single tier is complete. This type of 3D IC is addressed in this work. New countermeasures specifically for 3D Trojans are needed.

In this section, we applied an existing approach [19], originally designed for 2D Trojans, to a 3D system and compared the effectiveness of Trojan detection in 2D and 3D ICs. As 3D chips have severe internal noise, we suspect that Trojan detection using side-channel signals will lose its detection accuracy. Thus, we chose a current based Trojan detection method.

2.4.1 Description of Trojan Detection Method for 2D ICs

The Trojan detection method we examined is Temporal Self-Referencing (TeSR) [19]. In TeSR, a special test vector generator offers the input sequence to ensure the system go through the identical state transitions in a period of time. A Trojan free system should obtain identical current signatures in two consecutive time windows when it goes through the same state transitions. Any mismatch between the two current signatures will indicate the presence of a hardware Trojan. This method may not work well in 3D scenarios because of the greater internal noise in 3D ICs.

2.4.2 Targeted Hardware Trojan

In the following experiment, we inserted the same MOLES Trojan mentioned in [46] to the 2D and 3D circuits. The MOLES Trojan is composed of a set of registers as a ring generator to generate a series of random numbers, which will be XORed with the key information. The XOR outputs will drive a set of capacitors. Attackers who know the implementation details of the ring generator can decode the obfuscated key information via power analysis. However, the power consumed in the load capacitors seems like noise if the random sequence

Table 2.4: Trojan Detection Confidence for Different Victim Sizes.

	1 S-box	2 S-boxes	4 S-boxes	6 S-boxes	8 S-boxes
2D	+31.07%	+11.84%	+12.80%	+80.06%	+48.00%
3D	-21.99%	+12.61%	-61.30%	-24.32%	-28.74%

is unknown. In the 2D case, MOLES was implemented as an external circuit on the same tier of the target circuit. In the 3D scenario, MOLES and the victim circuit were placed in two different tiers.

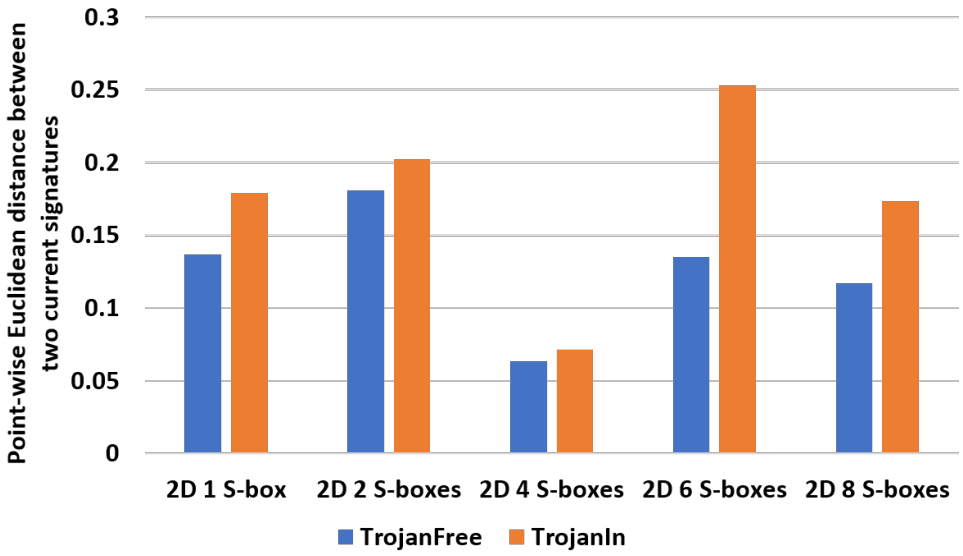
2.4.3 Efficiency of TeSR Trojan Detection Method in 2D and 3D ICs

We adopt the metric *point-wise Euclidean distance (PWED)* between the two current signatures to assess Trojan detection efficiency, following the similar process used in the work [19]. The PWED for the Trojan free case (i.e. TrojanFree) is considered as the noise threshold. If the PWED measured from the Trojan injected case (i.e., $PWED_{_{TrojanIn}}$) is higher than that measured from the Trojan free case (i.e., $PWED_{_{TrojanFree}}$), the hardware Trojan is detected.

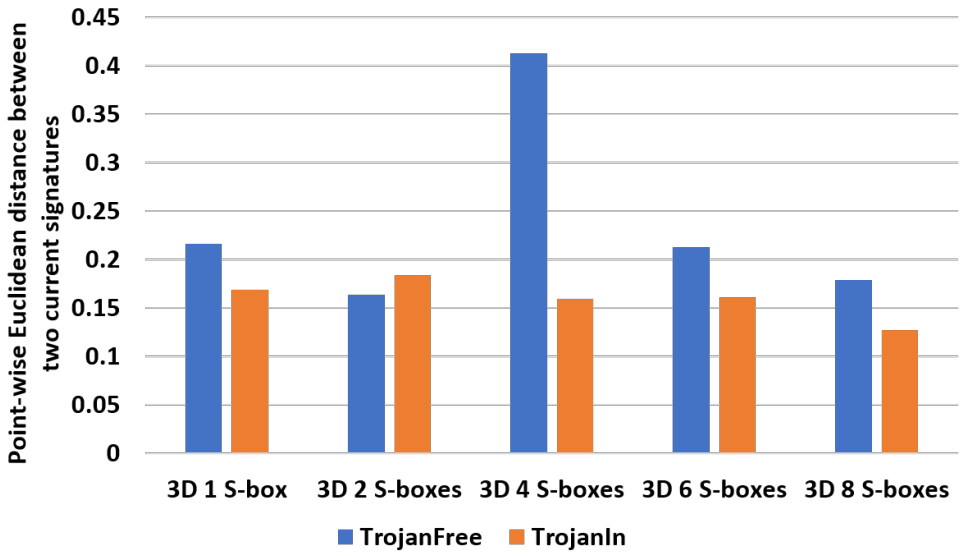
We implemented the TeSR Trojan detection method in the transistor-level 3D IC model built with a 45nm NCSU FreePDK technology [40]. The detailed setting is as same as what described in Section 2.3.2). One, two, four, six, and eight S-boxes were applied for the purpose of sweeping the size of the victim circuit. The number of registers in the MOLES ring generator was varied to observe the impact of Trojan size on Trojan detection efficiency.

Our simulation results shown in Fig. 2.14 confirm that the TeSR Trojan detection method is generally less effective in the 3D scenarios than in the 2D cases. The inserted MOLES Trojan can be successfully detected in the 2D environment for all victim sizes tested in the experiment. In contrast, the Trojan in the 3D scenario is not detected in most of the cases because the 3D $PWED_{_{TrojanIn}}$ is lower than $PWED_{_{TrojanFree}}$. We further zoom in the PWEDs for different test cases and define the confidence level of Trojan detection $Confidence_{HTD}$ as the expression shown in Eq. (2.1).

$$Confidence_{HTD} = \frac{PWED_{_{TrojanIn}} - PWED_{_{TrojanFree}}}{PWED_{_{TrojanFree}}} \tag{2.1}$$



(a)



(b)

Figure 2.14: Trojan detection results achieved by the TeSR approach applied in (a) 2D and (b) 3D ICs with different sizes of victim circuits.

Table 2.4 shows $Confidence_{HTD}$ for all the test cases reported in Fig. 2.14. A positive percentage means that the Trojan is detected. A higher percentage stands for better confidence in the detection result. If the positive percentage is too small, our detection conclusion may be changed by the interruption from some internal noise or process variations. Although TeSR achieves a positive confidence value in the 3D TrojanIn with 2 S-boxes case, the per-

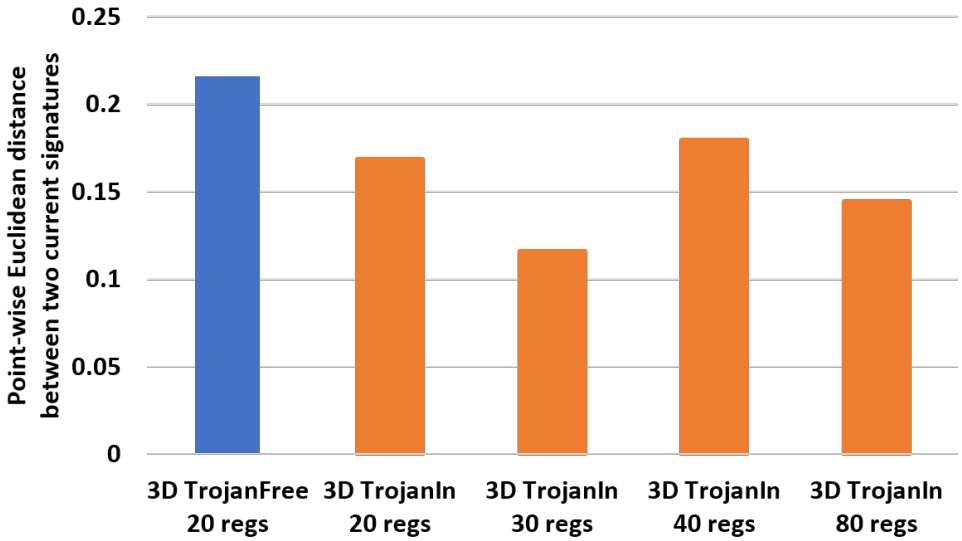


Figure 2.15: Trojan detection efficiency of TeSR against 3D MOLES Trojans with different sizes.

Table 2.5: Trojan Detection Confidence for Different Trojan Sizes.

3D TrojanIn 20 regs	3D TrojanIn 30 regs	3D TrojanIn 40 regs	3D TrojanIn 80 regs
-21.99%	-46.34%	-16.94%	-33.15%

centage of 12.61% is not as high as that in most of the 2D cases. A negative percentage in Table 2.4 indicates that the TeSR fails to capture the Trojan. To conclude, the MOLES Trojans in most of the 3D scenarios are not recognized by the TeSR approach.

Next, we swept the size of the MOLES Trojans from 20 to 80 registers and obtained the corresponding PWED shown in Fig. 2.15. As can be seen, the PWED for all 3D TrojanIn cases is less than the TrojanFree case. This indicates that the TeSR approach fails to detect the MOLES Trojans inserted in the 3D circuits even if the Trojan size increases. Another observation we had from our case study is, the PWED does not monotonically increase or decrease with the Trojan size. This is summarized in Table 2.5.

2.5 Conclusion

Three-dimensional integration techniques for integrated circuits leverage vertical-dimension space to increase the chip density and provide better performance than two-dimensional

chips. However, the increased number of transistors in a small footprint leaves more exploration space for attackers to insert stealthy hardware Trojans. Trojans in planar integrated circuits are well modeled and understood, but there is limited work available to investigate hardware Trojans specifically in 3D ICs. This chapter summarizes the existing effort on 3D hardware Trojans. To improve the awareness of potential attacks that could succeed in 3D ICs, this chapter characterizes four representable 3D hardware Trojan cases and provides practical simulation/emulation examples for each model. To the best of our knowledge, this is the first comprehensive work that analyzes the 3D Trojan models, especially for cross-tier and multi-tier Trojans, and demonstrates their impact with the quantitative assessment. Our experimental results show that 3D Trojans are feasible to be implemented in 3D integrated circuits and systems. We advocate the research community to investigate unique Trojan detection methods for 3D hardware Trojans.

CHAPTER 3

Two-Tier Activation (T^2A) Testing Scheme for 3D ICs to Detect TSV-Based Hardware Trojans

3.1 Introduction

Three-dimensional integrated circuits (3D ICs) offers higher device density, better chip performance and lower power consumption than planar chips. However, the vertical communication channels, through-silicon vias (TSVs), in 3D chips bring in new concerns of reliability and security issues. For example, the heat expansion of TSVs induces serious stress to 3D ICs under the condition of the poor heat dissipation, which may cause damage to the device layer of each die [47]. Moreover, the coupling effect between TSVs may impact the integrity of inter-tier communications and could even be used to form a crosstalk-based attack [48,49]. The recent literature reveals that TSVs could be exploited by attackers to build new hardware Trojans, which are more difficult to detect than the ones inserted in planar chips [9, 50].

The testing for 3D ICs is more complicated than that for traditional 2D ICs. The multi-tier structure in 3D ICs requires to perform multiple phase testing, including pre-bond, mid-bond and post-bond testings [6]. The pre-bond and mid/post-bond testing check the circuitry of individual dies and the TSV defects, respectively. To survive from the tier-level testing, the attackers from the bonding foundry could implant the Trojans in the interface of tier-to-tier connection. That type of Trojan will not modify the circuitry of any tiers or tamper with any TSVs, but only alter the signals transmitted through TSVs when the Trojan is activated. Due to the press of time-to-market, the limited functional testing performed

after the 3D IC is packaged does not provide sufficient test coverage to detect the 3D hardware Trojans.

The most recent 3D IC testing standard IEEE Std 1838 [51,52] provides a comprehensive tier-level testing mechanism after the chip is packaged. Similar to the testing of multi-core systems in 2D ICs, each die in a 3D stack is wrapped using die wrapper registers (DWRs). The test data inputs and DWRs will control the bypass registers of dies to directly test any single die's logic while bypass the uninterested dies. However, the standard does not provide a detailed testing scheme for the logic outside of DWRs but still in the die. Such logic is known as *shore logic*, which are extremely important to the integrity and security of 3D ICs because they are the interface to the TSVs. Any malicious modifications on the TSV signals could lead to the corruption of important instructions from neighboring tiers and cause system malfunction consequently.

In this chapter, we investigate the potential hardware Trojan insertion in the shore logic to manipulate TSV signals. More specifically, the main contributions of this chapter are as follows.

- We propose a TSV-based hardware Trojan that is inserted to the shore logic of 3D tiers. More specifically, the Trojan spreads its trigger and payload logic to two neighboring tiers and induces an extra Trojan TSV to transmit the trigger signal.

- We analysis and compare the Trojan detection probability for the proposed TSV-Trojan with the case of conventional hardware Trojans under the testing guided by IEEE Std 1838.

- Furthermore, we propose a two-tier activation (T^2A) enhancement method for IEEE Std 1838 to examine two neighboring tiers at a time in the testing and thus facilitates the detection of TSV-based Trojans.

3.2 Attack Model

IEEE Std 1838 normalizes the testing on the logic function of each die of a 3D stack. As shown in Fig 3.1, the die logic of 3D tiers is fully wrapped by DWRs which are controlled by bypass registers. Each tier can be tested individually by setting its bypass register value. However, the standard does not give a detailed testing approach for the shore logic. Hardware Trojans (HTs) may be inserted to those logic to exploit the relatively weak testing feature. Moreover, we found that IEEE Std 1838 only supports the testing mode, in which only one tier is tested at a time. A Trojan may bypass this tier-level testing if it has Trojan components spread among multiple tiers. The Trojan is not complete in the testing such that the tier will not have any abnormal functions.

In our attack model, we assume that attackers have the access of the fabrication of two neighboring tiers of a 3D IC. They can perform a hardware Trojan attack by inserting the Trojan trigger and payload to the tiers separately. The inter-tier communication between the Trojan components can be conducted using an extra induced TSV, labeled with Trojan TSV in Fig. 3.1. Once the Trojan is triggered, it will corrupt the TSV signals transmitting between the two tiers.

3.3 Proposed TSV-Based 3D Trojan

3.3.1 Description of Proposed Trojan

We assume that some attackers will exploit the fact that the IEEE Std 1838 is not capable of testing two neighboring tiers to design a hardware Trojan, which will not be triggered in the single-tier testing. For simplicity, we zoom in a TSV-based 3D Trojan across two tiers as shown in Fig. 3.2. Locating in the shore logic, the Trojan trigger and payload have direct interaction with some TSVs between two 3D tiers. The trigger circuit monitors the incoming TSV signals to Tier 2 and generates a trigger signal for the payload when a specific input pattern arrives. The trigger signal is propagated to the payload circuit via an extra Trojan

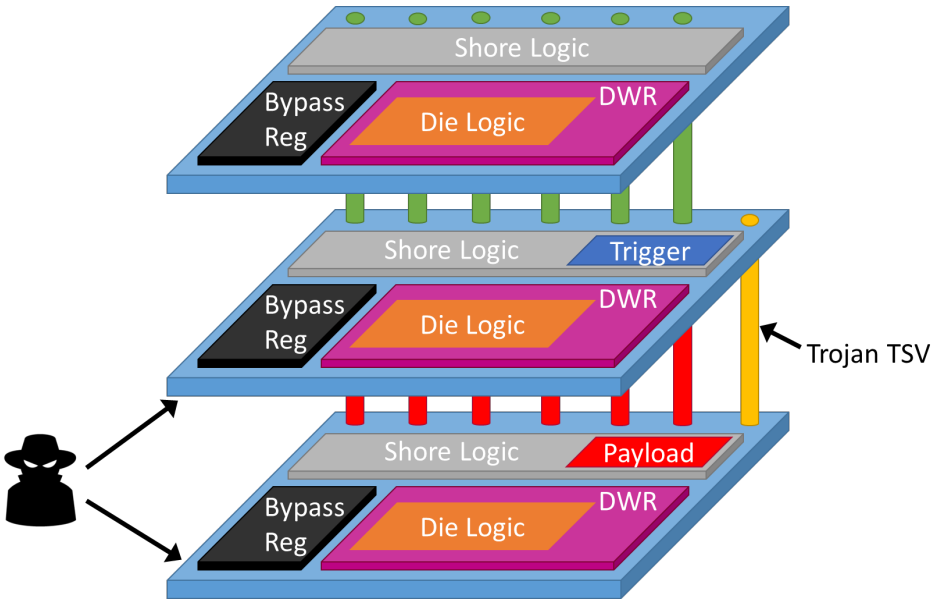


Figure 3.1: Diagram of testing structure for 3D ICs.

TSV, labeled with HT_TSV in Fig. 3.2. The payload will, for example, flip the incoming TSV signals from Tier 2 if the trigger signal is logic high.

If an advanced attacker can manipulate the bypass registers, s/he could implement a pre-activation scheme to further increase the complicity of the TSV-based hardware Trojan. In this type of Trojan, the trigger and payload will make use of the bypass register value to survive from the conventional single-tier testing. The stars in Fig. 3.2 highlight the data paths that are associated with the pre-activation scheme. The trigger and payload will stay dormant without the proper pre-activation signals. More specifically, the trigger circuit does not generate an active trigger signal until the bypass register of Tier 2 is logic 0, regardless of the input pattern designed for Trojan triggering has arrived or not. Likewise, the payload circuit cannot perform any modifications if the bypass register in Tier 1 is logic high, no matter the trigger signal from Tier 2 is active or not. In summary, the pre-activation condition is mandatory to enable the trigger and payload circuit to function properly. For instance, both the trigger and payload are pre-activated only if the 3D tier they are located in are not bypassed in the testing.

We name the TSV-based Trojans without the access to the bypass register as **TSV-**

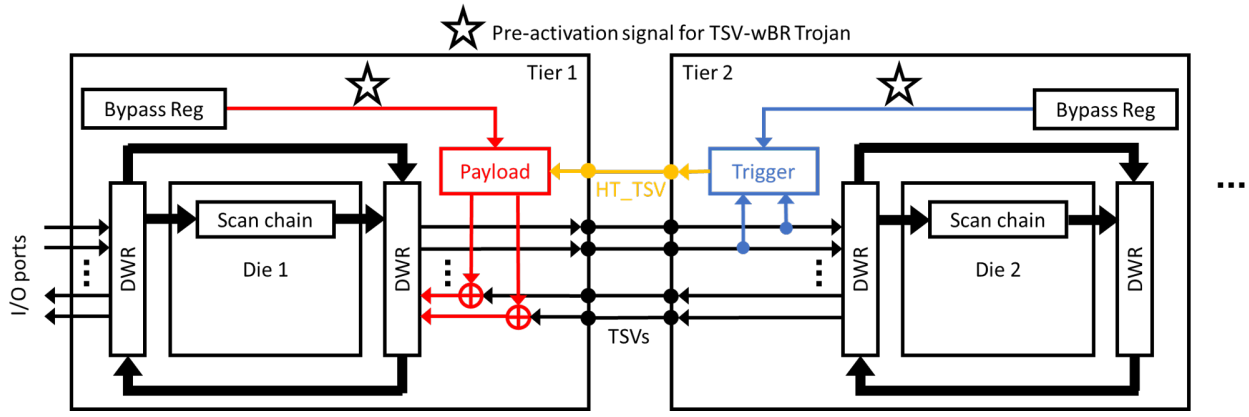


Figure 3.2: TSV-based 3D hardware Trojan attack in 3D tiers.

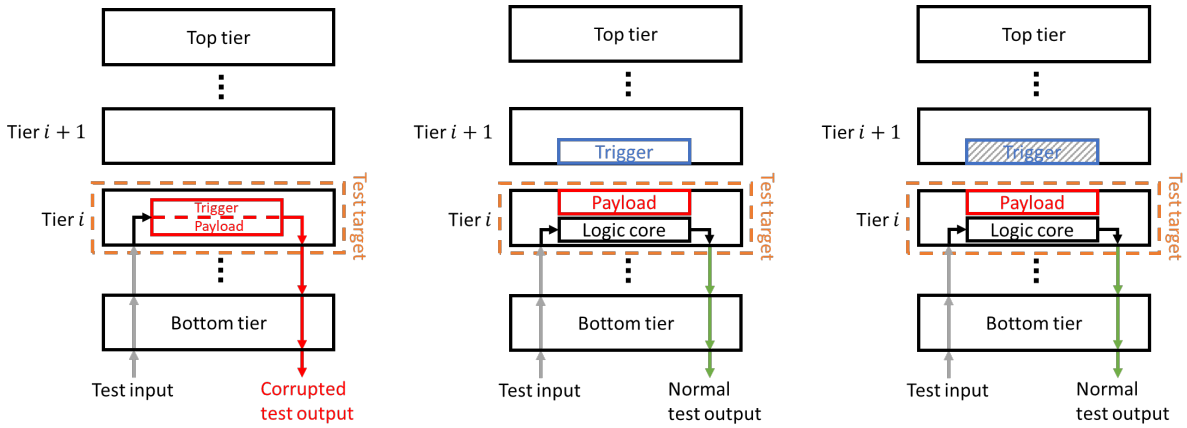
woBR and the more advanced Trojans that can monitor the bypass register as **TSV-wBR**. In the next section, we will analyze the stealthiness of the TSV-based Trojans by deriving the probability of successfully detecting TSV-woBR and TSV-wBR Trojans achieved by performing the single- and two-tier testing.

3.3.2 Theoretical Probability of Successful Trojan Detection

We use the probability of successful Trojan detection (PSTD) as a metric to compare the efficiency of different testing schemes. We assume the 3D IC in our following discussions have N tiers and the triggering probability for the Trojan is P for a given set of input stimuli.

PSTD Obtained by Single-Tier Testing

If the single-tier testing is performed through the frame defined by the IEEE Std 1838, it is less likely to successfully detect a TSV-based 3D Trojan than a conventional 2D Trojan. Without loss of generality, we assume a conventional Trojan is inserted in Tier i of a 3D IC and a TSV-based 3D Trojan involves Tiers i and $i+1$. As shown in shown in Fig. 3.3(a), the single-tier testing on Tier i can facilitate the conventional Trojan detection as long as the Trojan is triggered by the test input and the corrupted test output is observable from the bottom tier. In contrast, the single-tier testing on Tier i will not be able to activate the trigger circuit of the TSV-based Trojan on Tier $i+1$ (since the test input will not reach Tier



(a) (b) (c)

Figure 3.3: Single-tier testing based (a) conventional Trojan, (b) TSV-woBR Trojan and (c) TSV-wBR Trojan detection.

Table 3.1: Comparison of probability for successful Trojan detection.

Target Trojan	Test frame location	Detection output		PSTD	
		Single-tier testing	Two-tier testing	Single-tier testing	Two-tier testing
TSV-woBR HT	Before Tier i	✗	✗	$\frac{N-i}{N}P$	$\frac{N-i}{N-1}P$
	At Tier i	✗	✓		
	After Tier i	✓	✓		
TSV-wBR HT	Before Tier i	✗	✗	0	$\frac{1}{N-1}P$
	At Tier i	✗	✓		
	After Tier i	✗	✗		
Conventional HT	Before Tier i	✗	—	$\frac{N-i+1}{N}P$	—
	At Tier i	✓	—		
	After Tier i	✓	—		

$i+1$) and thus the Trojan effect will not affect the normal test output collected through the bottom tier, as shown in Figs. 3.3(b) and (c).

As the Trojan payload is located on Tier i, we examine whether the conventional 2D and TSV-based 3D Trojans can be detected by the single-tier testing performed before, at and after Tier i. As compared in the column 3 of Table 3.1, the single-tier testing cannot detect most of the TSV-based Trojan and can successful identify the conventional Trojan if the testing tier is at or after Tier i. We further derive the PSTD for each category. To visualize our theoretical derivation of PSTD, we set N to 4 and swept the location of the

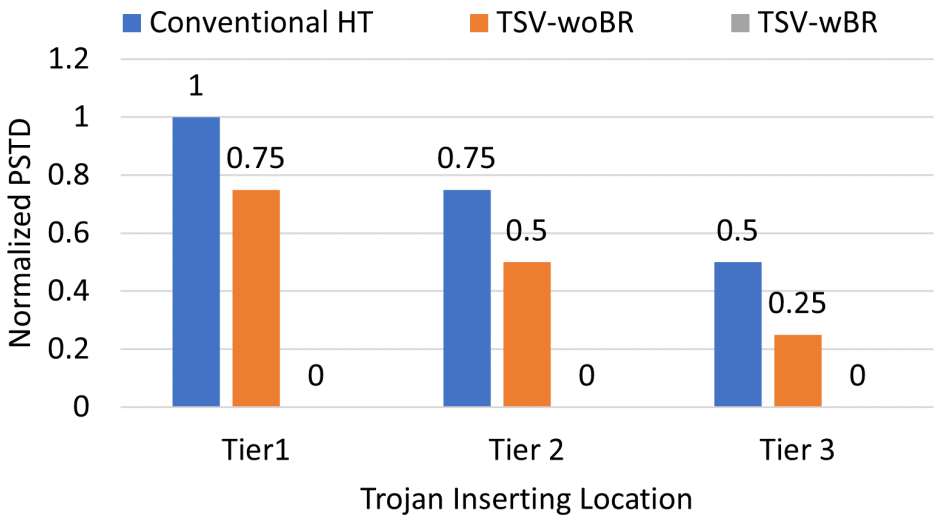
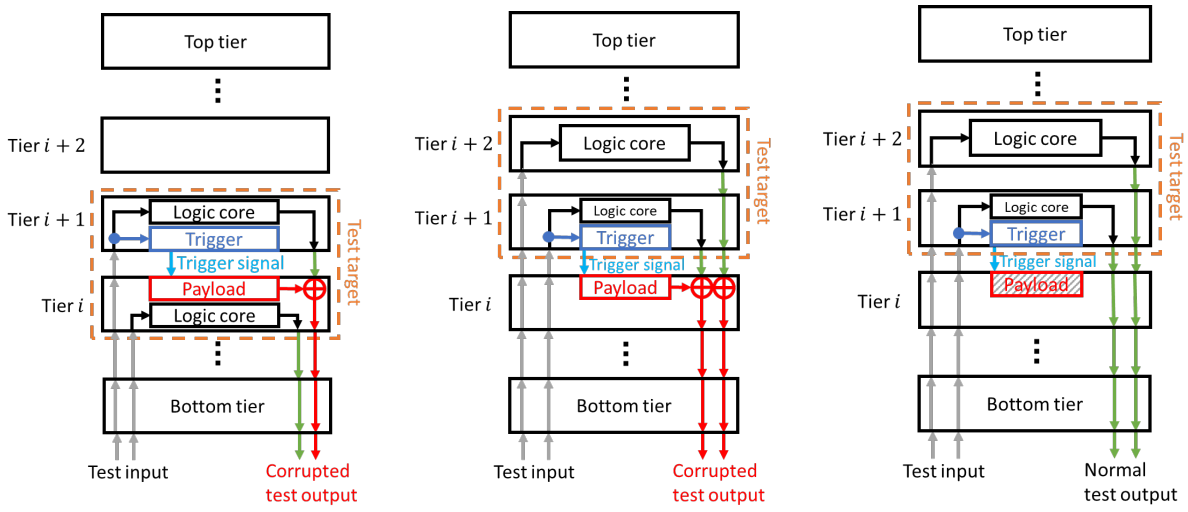


Figure 3.4: Normalized Trojan detection probability comparison between conventional and TSV-based Trojans.

conventional and TSV-based Trojans in the 3D structure. We assume the set of input stimuli in each testing is the same. After normalizing the factor P, we plot the PSTD for each case in Fig. 3.4. As can be seen, the normalized PSTD for the proposed TSV-based Trojan is always lower than that for the conventional Trojan. This theoretical analysis result confirms that TSV-woBR and TSV-wBR HTs are more difficult to detect if we execute the single-tier testing.

PSTD Obtained by Two-Tier Testing

To address the limitation of the single-tier testing on the TSV-based Trojan detection, we analyze the Trojan detection efficiency of two-tier testing, which activate two 3D tiers simultaneously during testing. As shown in Fig. 3.5(a), a test target frame (denoted by the orange dash line) covering Tiers i and $i+1$ can detect the TSV-based Trojan across Tiers i and $i+1$. When we set the test target frame at a higher tier location $i+1$ and $i+2$, the two-tier testing detects the TSV-woBR Trojan, as shown in Fig. 3.5(b). However, for the TSV-wBR Trojan in Fig. 3.5(c), setting the test target frame higher will fail the Trojan



(a) (b) (c)

Figure 3.5: Two-tier testing on TSV-based Trojans. (a) Setting test target at Tiers i and $i+1$ detects TSV-based Trojan. Setting test target at Tiers $i+1$ and $i+2$ (b) detects TSV-woBR Trojan but (c) fails in detecting TSV-wBR Trojan.

detection because the payload cannot be activated. The conceptual comparison shown in Fig. 3.5 indicates that TSV-wBR Trojan is more difficult to detect than TSV-woBR Trojan in two-tier testing.

We summarize the detection outputs for different test target frame locations in the column 4 of Table 3.1. The corresponding PSTD for each Trojan type is listed in the column 6. Since conventional Trojans are not our focus in this paper, we do not omit the detection output or PSTD for the conventional Trojan case. We vary the number of tiers N from 2 to 10 and plot the theoretical PSTD improvement achieved by the two-tier testing in Fig. 3.6. As shown, the improved PSTD for TSV-woBR Trojan will drop when N increases. This indicates that the advantage of two-tier testing in TSV-woBR Trojan detection may be more obvious for the 3D chips that have relatively fewer tiers. However, because the single tier testing cannot detect TSV-wBR Trojans, the improved PSTD for TSV-wBR Trojan will always be 100%.

We further investigate the impact of Trojan insertion location on PSTD. We assume a 3D IC has 6 tiers, the PSTD results for all the possible Trojan insertion locations are listed in Table 3.2. We assumed three scenarios where attackers insert Trojans differently. In

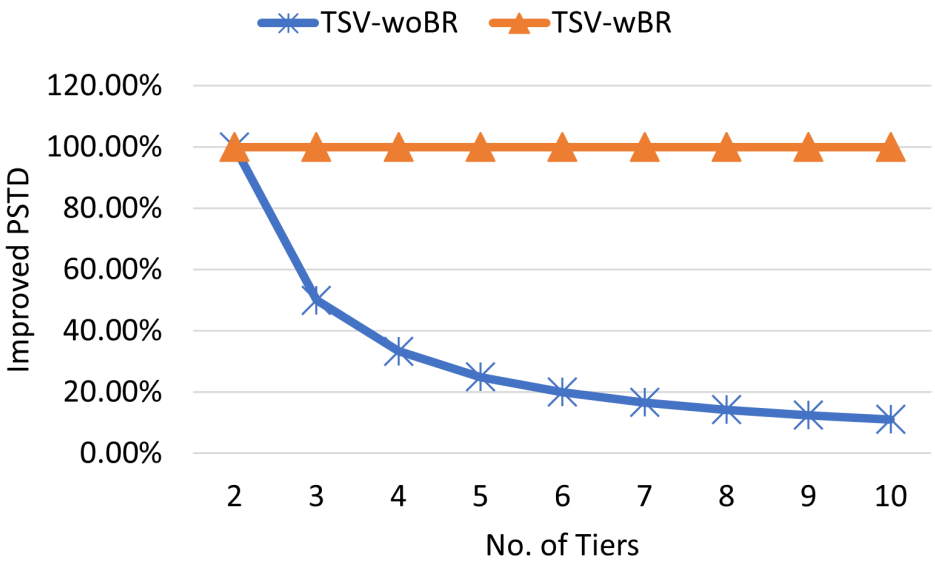


Figure 3.6: Improved Trojan detection probability by two-tier activation scheme.

Table 3.2: Comparison of PSTD Achieved by Single- and Two-tier Testing.

Trojan type	Testing	i=1	i=2	i=3	i=4	i=5	Overall		
							Even	*Linear*	*Normal*
TSV-woBR	Single-tier testing	$0.83P$	$0.67P$	$0.5P$	$0.33P$	$0.17P$	$0.5P$	$0.42P$	$0.59P$
	Two-tier testing	P	$0.8P$	$0.6P$	$0.4P$	$0.2P$	$0.6P$	$0.5P$	$0.71P$
TSV-wBR	Single-tier testing	0	0	0	0	0	0	0	0
	Two-tier testing	$0.2P$	$0.2P$	$0.2P$	$0.2P$	$0.2P$	$0.2P$	$0.2P$	$0.2P$

the scenario of *Even*, we assume attackers randomly choose a Trojan insertion location. In *Linear*, attackers prefer to insert a Trojan in the higher tiers. In *Normal*, attackers prefer to place a Trojan in the middle tiers. In different scenarios, the probabilities of inserting the Trojan in each location (i=1, 2, 3, 4, and 5) are different. As shown in Fig. 3.7, the two-tier testing improves the PSTD in all scenarios. In summary, our analysis confirms that two-tier testing is superior to single-tier testing in the sense of TSV-based Trojan detection, if the same test input stimuli is applied in the testing.

3.4 Proposed Two-Tier Activation (T^2A) Testing Enhancement Method

The theoretical analysis in Section 3.3.2 indicates that activating two neighboring tiers simultaneously in the testing can significantly improve the probability of successfully detecting

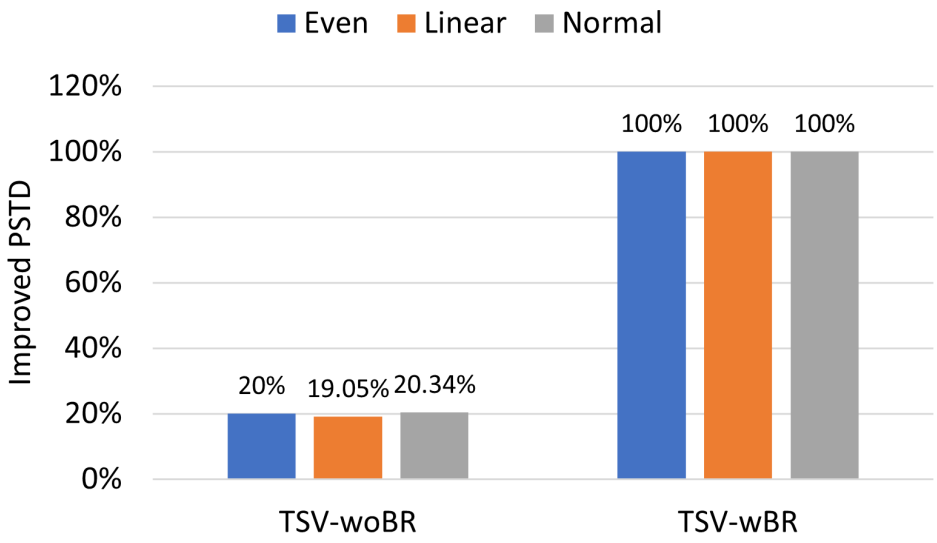


Figure 3.7: Improved Trojan detection probability by two-tier activation scheme.

TSV-based Trojans. Thus, we propose a *two-tier activation (T^2A) testing architecture* to enable the detection of both TSV-woBR and TSV-wBR hardware Trojans.

3.4.1 Overview of T^2A Method

As shown in Fig. 3.8, the testing frame for our Trojan detection method is an enhancement over what IEEE Std 1838 uses (as shown in Fig. 3.1). In T^2A, we add a position determination module (PDM) to each tier. It takes the bypass register values from its current tier and its previous tier as inputs and reports the current tier's position in the test frame. We further induce an extra TSV, BR-TSV, between every two neighboring tiers to transmit bypass register values. Based on the different positions determined by PDMs, the tiers will have different data paths during a test. In this way, PDM takes over the role of the bypass register in IEEE Std 1838 to decide which tiers are bypassed or activated in the ongoing testing. More specifically, the data flow inside test frame ensures the test input is given to the top-half tier (Tier 2 in Fig. 3.8) in the frame and the test output is from the top-half tier as well. This is because the payload in the bottom-half tier (Tier 1 in Fig. 3.8) will flip the incoming TSV signals from Tier 2, once the test input given to Tier 2 satisfies the triggering

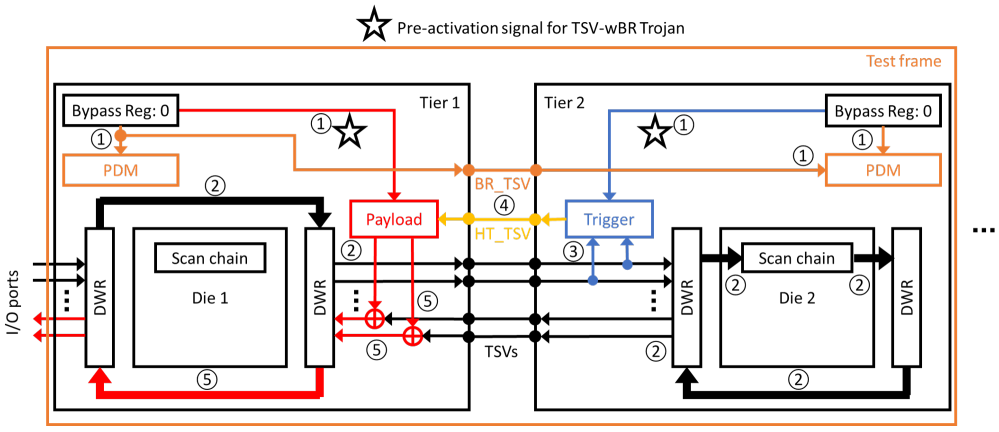


Figure 3.8: Proposed T^{2}A method in detecting TSV-based Trojans.

condition of the trigger logic there. The corrupted test output from Tier 2 will be passed to I/O ports and the Trojan is detected. There is no need to test the logic function of the bottom-half tier even though it is in the test frame.

Figure 3.8 demonstrates an example of the proposed T^{2}A method successfully detecting the TSV-based Trojan. The test frame currently includes the bottom two tiers of a 3D IC. When a test begins, the PDM in each tier of the 3D IC reads in the bypass register values and determines the tier's position. Note, the PDM in bottom tier only needs the bypass register value from its current to determine the position. The test input coming into the test frame will bypass Tier 1 and directly reach Tier 2, the top-half tier of test frame. The logic of Tier 2 will be tested and the test output is then passed back to Tier 1. During this process, the Trojan trigger in Tier 2 will be activated by the test input and the triggered payload in Tier 1 will flip the text output from Tier 2. The corrupted test output then can be observed in I/O ports. The data flow in the test described above follows the order labeled in Fig. 3.8.

3.4.2 Place Determination Module (PDM)

The logic defined in the PDM can derive which tier is being tested. A PDM works as a look up table (LUT) which takes the bypass register values from the current tier (BR_Pre) and the previous tier (BR_Curr) as inputs and controls the DWRs to conduct different data

Table 3.3: The Number of Clock Cycles Needed in TSV-woBR Trojan detection.

			Trojan1	Trojan2	Trojan3	Trojan4	Trojan5	Average	Reduced detection time
2 tiers	IEEE Std 1838		56	34	41	43	55	45.8	68%
	T^2A		25	3	10	12	24	14.8	
3 tiers	IEEE Std 1838	Trojan in tiers 1 and 2	56	34	41	43	55	61.3	51%
		Trojan in tiers 2 and 3	87	65	72	74	86		
	T^2A	Trojan in tiers 1 and 2	25	3	10	12	24	30.3	
		Trojan in tiers 2 and 3	56	34	41	43	55		
4 tiers	IEEE Std 1838	Trojan in tiers 1 and 2	56	34	41	43	55	76.8	40%
		Trojan in tiers 2 and 3	87	65	72	74	86		
		Trojan in tiers 3 and 4	118	96	103	105	117		
	T^2A	Trojan in tiers 1 and 2	25	3	10	12	24	45.8	
		Trojan in tiers 2 and 3	56	34	41	43	55		
		Trojan in tiers 3 and 4	87	65	72	74	86		

Table 3.4: The Number of Clock Cycles Needed in TSV-wBR Trojan Detection.

		Trojan 1	Trojan 2	Trojan 3	Trojan 4	Trojan 5	Average
c17	IEEE Std 1838	n/a	n/a	n/a	n/a	n/a	n/a
	T^2A	25	3	10	12	24	14.8
c432	IEEE Std 1838	n/a	n/a	n/a	n/a	n/a	n/a
	T^2A	122	16	61	158	195	110.4
c880	IEEE Std 1838	n/a	n/a	n/a	n/a	n/a	n/a
	T^2A	31	43	116	28	30	49.6

paths based on its output. If we assume the bypass register value of a tier is zero if the tier is under test currently. The key logic of the PDM is specified in Table 3.5. If the a tier is determined as the top-half tier of test frame, the DWRs of it will direct the test input to the logic core and send the test output from the logic core to the bottom-half tier. If a tier is determined as the bottom-half tier, its DWRs will directly relay the test input to the top-half tier instead of letting it entering the logic core. The DWRs will also pass the test output from top-half tier to their next neighboring tier. For the tiers that are outside of the test frame, they only play a role of relaying test input and output data.

3.5 Experimental Results

3.5.1 Experimental Setup

We implemented the proposed T^2A method and use a Xilinx Spartan-6 FPGA platform to emulate the 3D structure. We assume each tier of the 3D ICs tested in our experiments has the same circuitry. ISCAS'85 circuits c17, c432 and c880 are used as benchmarks to configure

the 3D tiers. For example, one emulated 3D IC in our experiments has 3 tiers and each one is configured using a benchmark circuit. Two levels of TSV-based Trojans are inserted to the 3D ICs. Because our TSV-based Trojan focuses on modifying the behavior-level function of 3D ICs, it is not necessary to create a device-level 3D IC for the emulation. In this way, we create a logic module for each 3D tier and map them all in one FPGA device. The modules are connected using wires to simulate TSVs. The Trojans are inserted to two of the modules and an extra wire is induced to simulate the Trojan TSV. The effectiveness and efficiency of the proposed T^2A method is examined and compared with the baseline version of IEEE Std 1838. Furthermore, the area and delay overhead induced by T^2A method is evaluated in the Xilinx ISE and PlanAhead software, respectively.

3.5.2 Trojan Detection Speed

TSV-woBR Trojans

As discussed in Section 3.3.2, the improvement on the PSTD for TSV-woBR Trojans depends on the number of tiers in the 3D IC, we examine the number of clock cycles needed to detect the Trojan that is inserted in one of the 2-tier, 3-tier, and 4-tier stacked 3D ICs. The original IEEE Std 1838 and the proposed T^2A method were adopted in the testing. For the cases where the 3D ICs have 3 or 4 tiers, the Trojan could be inserted to different locations within a 3D stack. We considered all possible Trojan locations and ran the Trojan detection on each case with 5 trials. Each trial represents a unique triggering condition, which will consume a unique period of time for Trojan triggering. As shown in Table 3.3, our T^2A method reduces the time spent on the TSV-woBR Trojan detection by 68%, 51%, and 40% for the 2-tier, 3-tier, and 4-tier 3D cases, respectively, compared to IEEE Std 1838. The improvement will be slightly degraded with the increase of the number of tiers, which is consistent with the predicted improvement on the Trojan detection probability shown in Fig. 3.6. If we increase the number of tiers in the test frame, this limitation can be addressed.

Table 3.5: Logic Table Defined in PDM

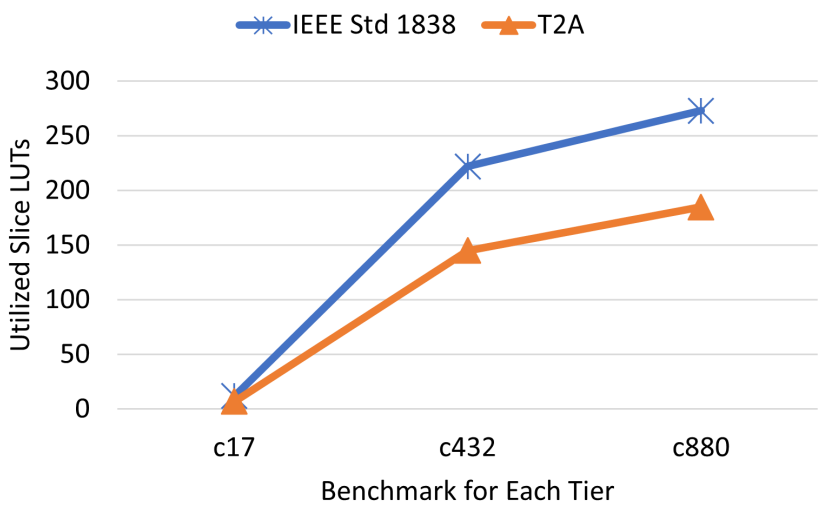
BR_Pre	BR_Curr	Location in two-tier testing unit
0	0	Top-half
0	1	Out of frame
1	0	Bottom-half
1	1	Out of frame

TSV-wBR Trojans

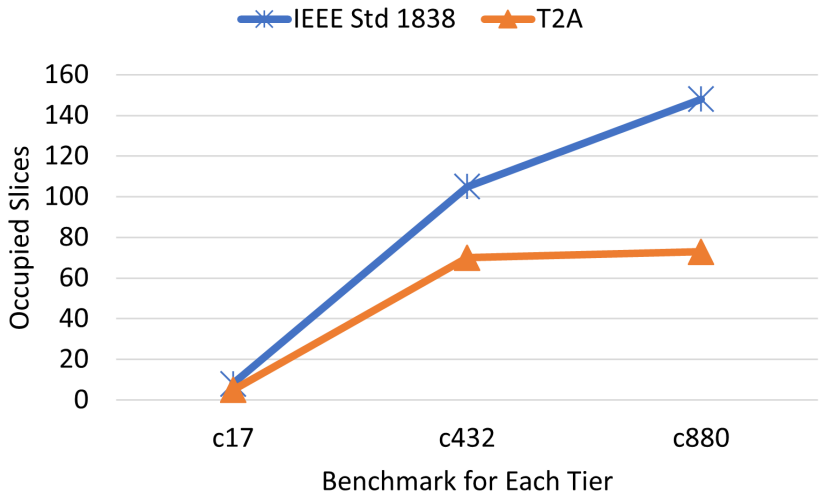
Our proposed T^{2}A method outperforms the original IEEE Std 1838 as T^{2}A can successfully detect TSV-wBR Trojans. In the experiments presented in this subsection, we fixed the Trojan insertion location and varied the benchmark circuit (c17, c432, and c880) in our testing. The time consumed by different Trojan detection methods is shown in Table 3.4. As the original IEEE Std 1838 fails in detecting any Trojans inserted in the experiments, the execution time is denoted as 'n/a' in Table 3.4. In contrast, the proposed method succeeds in all the test cases, and the number of clock cycles needed for Trojan detection depends on the size of the benchmark circuit deployed in the 3D tiers.

3.5.3 Overhead Evaluation

The proposed T^{2}A method modifies the original testing algorithm defined in the IEEE Std 1838. Instead of performing the testing on each tier, our method is able to activate two neighboring tiers per testing. More specifically, the method always returns the test output from the tier that is located at the top-half of the test frame. This means the bottom tier of a 3D stack will never need to be checked for test output. Comparing to the original IEEE Std 1838, the proposed T^{2}A is expected to induce less overhead. The overhead on hardware cost is evaluated in Xilinx ISE. As shown in Fig. 3.9, T^{2}A indeed consumes less FPGA resources than IEEE Std 1838. The delay overhead is measured by Xilinx PlanAhead software and compared in Fig. 3.10. As we can see, our method reduces the delay overhead by 12.03%, 10.8%, and 7.94% in the case of using benchmark circuit c17, c432, and c880, respectively.



(a)



(b)

Figure 3.9: Hardware cost comparison in (a) utilized slice LUTs and (b) occupied slices in FPGA.

3.6 Conclusion

Due to the unique structure and complicated testing environment of 3D ICs, it is more complicated and difficult to detect 3D hardware Trojans than the conventional Trojans in 2D chips. This chapter proposes TSV-based hardware Trojans in 3D ICs. We perform theoretical analysis on the probability of successful Trojan detection under the testing frame

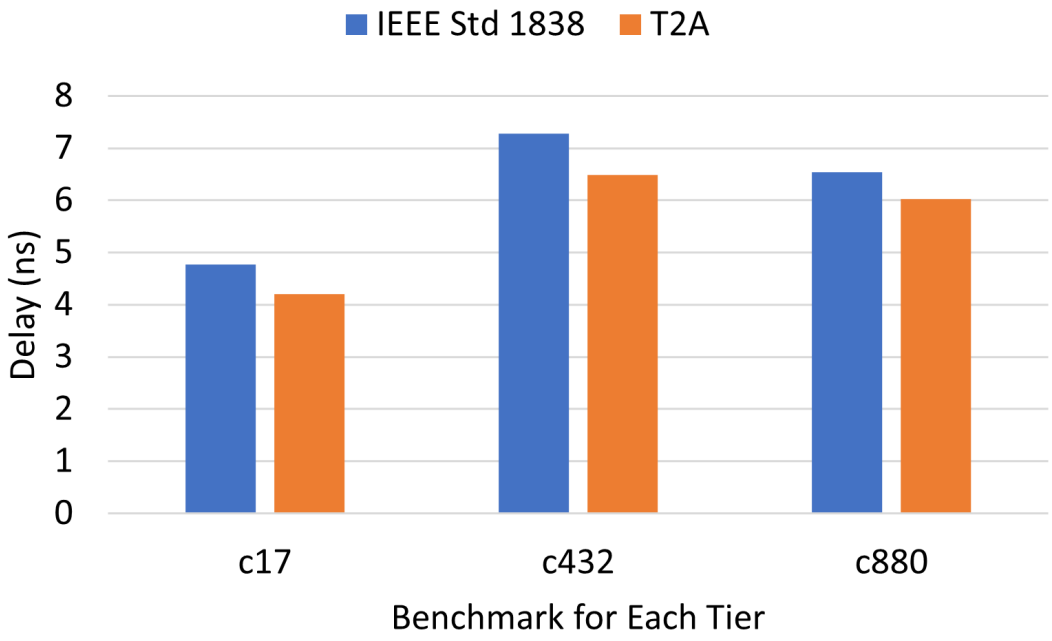


Figure 3.10: Delay overhead comparison.

defined by IEEE Std 1838, and reveal the limitation of single-tier testing on 3D Trojan detection. Furthermore, we further propose a T^2A enhancement method to improve the test frame of the IEEE Std 1838, thus achieving better 3D Trojan detection rate and consuming less Trojan detection time. Our experimental results show that the proposed method can successfully detect the TSV-based Trojans and reduce up to 68% of detection time compared to the original IEEE Std 1838.

CHAPTER 4

FTAI: Frequency-based Trojan-Activity Identification Method for 3D Integrated Circuits

4.1 Introduction

Existing Trojan-detection methods are mostly proposed for conventional 2D ICs. Their detection effect might be degraded in a 3D environment. For example, functional-verification based methods may not work well in 3D scenarios. First, the larger number of transistors integrated into the 3D package makes the exhaustive functional verification more sophisticated and time-consuming. Moreover, the limited probing capabilities do not allow us to simultaneously access all die-to-die vertical communication channels for thorough testing neither. Side-channel based Trojan detection is commonly used in securing 2D ICs. However, because 3D ICs usually have more internal noise than 2D ICs, the signal-to-noise ratio (SNR) of the side-channel signals for detection will be reduced noticeably. Larger variations on the process, voltage, and temperature in 3D ICs further lead to a higher false-positive rate. Thus, it is more challenging to precisely extract Trojan's impact on side-channel indicators or/and functional behaviors in the 3D scenarios [6].

To facilitate side-channel based Trojan detection in 3D ICs, it is imperative to develop a new method to tolerate the interference from 3D noise. In this chapter, we propose a Frequency-based Trojan-Activity Identification (FTAI) to detect 3D Trojans. Our FTAI method tolerates 3D noise and achieves a high Trojan detection rate. Comparing to the existing frequency-based detection methods, such as [53], FTAI takes process variation into consideration and provides a new way of threshold generation without using a fabricated

golden chip. Our theoretical analyses verify that the Trojan effect is more differentiable in the frequency spectrum than in timing waveform, no matter it acts as an additive or multiplicative noise. The experimental results further show that FTAI increases the Trojan detection rate by 38.1% compared to the time-domain detection method.

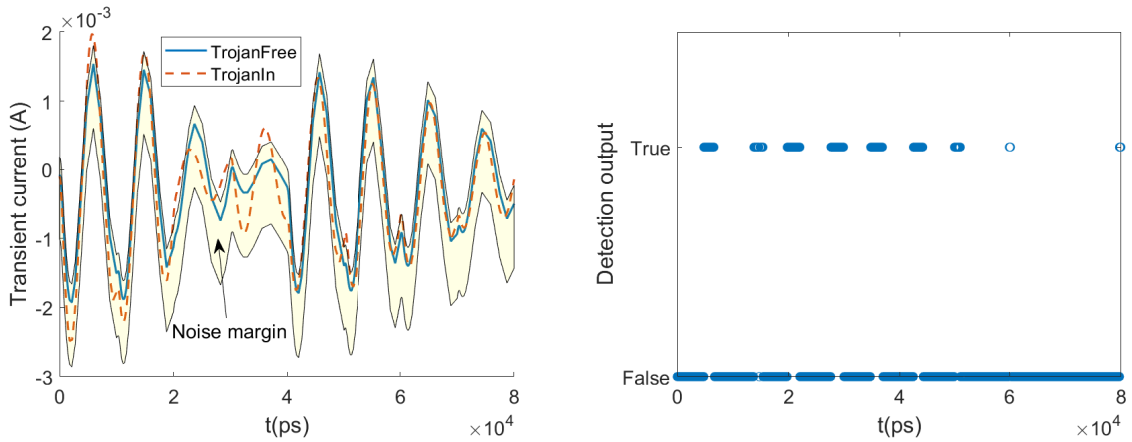
4.2 Trojan Model

3D hardware Trojans are characterized in Chapter 2 and the recent work [9]. In this chapter, we aim to detect the cross-tier hardware Trojan in 3D ICs. The goal of the 3D Trojan is to leak the secret key of a crypto unit implemented in the middle tier. The trigger is located in the same tier as the crypto unit while the payload is in the top tier. The trigger and payload circuits are inserted in two different single-die fabrication phases. According to the cross-tier hardware Trojan model in Chapter 2 and [9], the Trojan is not functioning during the single-tier testing stage but will be triggered after all 3D tiers are assembled.

We extend the MOLES Trojan [46], which is modeled for 2D ICs, to a 3D version. MOLES is composed of a set of registers as a ring generator to produce a series of random numbers, which will be XORed with the crypto key. The XOR outputs drive a set of capacitors as the Trojan payload. Attackers who know the implementation details of the ring generator can decode the obfuscated key information via power analysis. However, the power consumed in the load capacitors seems like noise if the random sequence is unknown. To form a cross-tier Trojan, the trigger and the ring generator of MOLES are inserted in the middle tier of our transistor-level 3D chip. The crypto unit, an AES Sbox, is located in the middle tier as well. The crypto key for AES will be leaked with eight capacitors. More details are available in Section 4.5.1.

4.3 Limitation of Time-domain Analysis based Trojan Detection

Time-domain analysis on the transient current of the circuit under Trojan attacks could reveal the presence of hardware Trojans, which contribute to more/less current. The effi-



(a) (b)

Figure 4.1: Time-domain analysis for Trojan detection. (a) Transient currents for three test cases, (b) Success/Failure of Trojan detection.

ciency of time-domain analysis heavily depends on the difference between the Trojan-induced current change and pre-existing inherent noise. A smaller difference leads to a higher false-positive/negative detection rate. Figure 4.1(a) shows the timing waveform for the transient current measured from our transistor-level 3D chip. The current was collected from the power-supply pin of the chip. The *TrojanFree* line in the graph represents a basic 3D chip. The *TrojanIn* line indicates the current after the injected Trojan is triggered. We further introduced process variation to the TrojanFree case to create noise margins, which are highlighted by the yellow area. As shown in Fig. 4.1(a), the impact of Trojans on the transient current does not exceed the boundaries defined by the noise for most of the time. If we consider the cases in which the TrojanIn line goes beyond the noise margin as the success of Trojan detection, the detection output is shown in Fig. 4.1(b). A very small portion of the line reaches *True* (i.e., detected) and the overall success detection rate is only 16.98%.

4.4 Proposed Frequency-based Trojan-activity Identification (FTAI) for 3D Hardware Trojans

4.4.1 Overview of Proposed FTAI Method

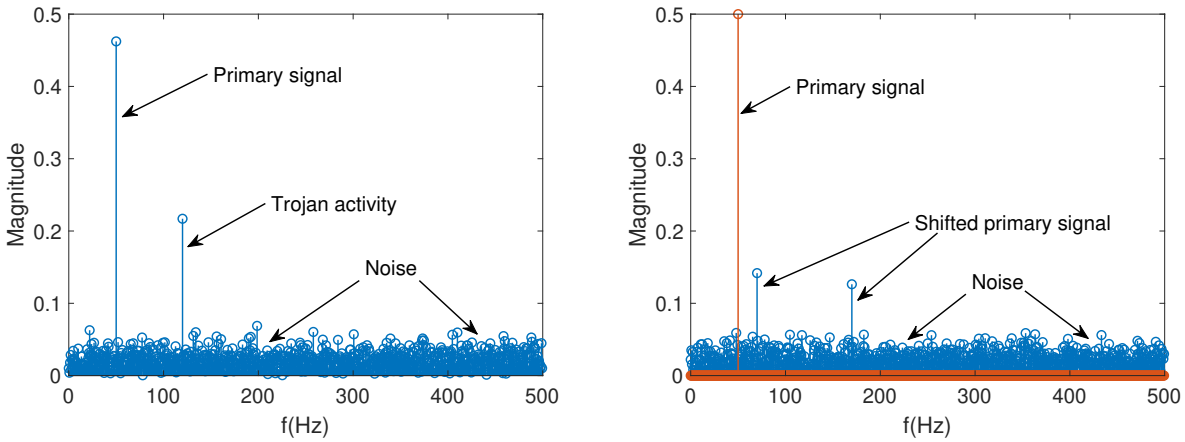
As 3D integration techniques bring in new security threats to ICs, it is imperative to develop effective Trojan detection methods for 3D chips. Since time-domain Trojan analysis methods suffer from noise interference, we explore new methods performed in the frequency domain. In this chapter, we propose a frequency-based Trojan-activity identification (FTAI) method, which exploits the frequency spectrum of the transient current of a 3D chip under Trojan attacks to detect hardware Trojans. We follow the footprint of the work [53] but specifically tailor the detection method for 3D ICs, which are known to have more variation on process/voltage/temperature and internal noise. Different than the work [53], our method waives the assumption on the frequency band of potential Trojans and the independence between primary circuits and Trojans. Furthermore, we propose a new threshold generation algorithm to achieve a high Trojan detection rate and reduce the false-positive rate over the existing work.

4.4.2 Theoretical Analysis

First, we assume that I_{Prime} and I_{HT} represent for the transient current contributed by the primary circuit and hardware Trojan, respectively. If the Trojan is an extra circuit that is independent of the primary circuit (i.e., victim module), we can model the total current for the circuit suffering from the Trojan attack with the expression shown in Eq. (4.1).

$$\begin{aligned} I_{tot} &= I_{Prime} + I_{HT} + n(t) \\ &= A_{Prime} sin(2\pi f_{Prime} t) + A_{HT} sin(2\pi f_{HT} t) + n(t) \end{aligned} \tag{4.1}$$

In which, A_{Prime} and f_{Prime} represent the amplitude and frequency for I_{Prime}. Similarly, A_{HT} and f_{HT} are the amplitude and frequency of the Trojan current I_{HT}. We use sinusoidal



(a) (b)

Figure 4.2: Frequency spectrum of (a)I_{tot} and (b) $I_{tot_{mul}}$.

functions to model the current components since most kinds of signals in nature can be modeled with a format of sinusoids [54]. The term $n(t)$ is white noise.

After *Fourier Transformation*, we will observe that the frequency spectrum $\mathcal{F}(I_{tot})$ includes three kinds of frequency components as shown in Eq. (4.2). Because the frequency response of the white noise is a constant value approximately, we use C to substitute $\mathcal{F}(n(t))$. The corresponding spectrum for $\mathcal{F}(I_{tot})$ is shown in Fig. 4.2(a). Different than the frequency response of noise, which is flat at the bottom of the entire spectrum, the Trojan activity will result in unique and substantial frequency response.

$$\begin{aligned}\mathcal{F}(I_{tot}) \approx \frac{A_{Prime}}{2i}[\delta(f - f_{Prime}) + \delta(f + f_{Prime})] + \\ \frac{A_{HT}}{2i}[\delta(f - f_{HT}) + \delta(f + f_{HT})] + C\end{aligned} \tag{4.2}$$

In addition to the additive influence on the total current, the current contribution from the hardware Trojan can be modeled as a multiplicative component if the Trojan is inserted by performing malicious modifications to the primary circuit. We formulate the total current $I_{tot_{mul}}$ in Eq. (4.3). After performing the Fourier transformation on $I_{tot_{mul}}$, we can obtain the frequency-domain expression for the total current $\mathcal{F}(I_{tot_{mul}})$, which is expressed in Eq. (4.4).

$$I_{tot_{mul}} = (I_{Prime} \times I_{HT}) + n(t) \tag{4.3}$$

$$\begin{aligned}\mathcal{F}(I_{tot_{mul}}) \approx \frac{A_{Prime}A_{HT}}{-4}\{&\delta[f - (f_{Prime} + f_{HT})]+ \\ &\delta[f - (f_{Prime} - f_{HT})] + \delta[f - (-f_{Prime} + f_{HT})]+ \\ &\delta[f - (-f_{Prime} - f_{HT})]\} + C\end{aligned} \tag{4.4}$$

Because multiplication in the time domain is transformed to convolution in the frequency domain, the frequency of the primary current will have an offset induced by the Trojan. Figure 4.2(b) shows the spectra of $\mathcal{F}(I_{tot_{mul}})$ and primary signal together. We can see the frequency of the primary signal is shifted by the Trojan. In conclusion, our theoretical analysis indicates that the impact of Trojans on the frequency spectrum can be easily differentiated from white noise. This motivates us to propose a frequency-based detection method for 3D Trojans.

4.4.3 Detection Flow

The detection flow for the proposed FTAI method is composed of three phases: *preliminary Trojan inspection*, *reference threshold generation*, and *final scrutinization*. Figure 4.3 depicts the detailed detection flow.

In the phase of the preliminary Trojan inspection, one needs to collect the total transient current of the 3D chip from the power-supply pin. Then, *Fourier transformation* is utilized to convert the time-domain current trace to its frequency-domain representation $\mathcal{F}_{real}$. Next, the same process is repeated on the transistor-level 3D model for the same 3D chip to obtain $\mathcal{F}_{sim}$. The two frequency spectra $\mathcal{F}_{real}$ and $\mathcal{F}_{sim}$ are compared to identify the suspicious frequency band $\mathcal{F}_{HT}$, in which the Trojan may be located. This process will minimize the noise interference on Trojan detection, as discussed in Section 4.4.2. We performed a simulation to compare the frequency spectra for the current trace of the golden model (i.e.,

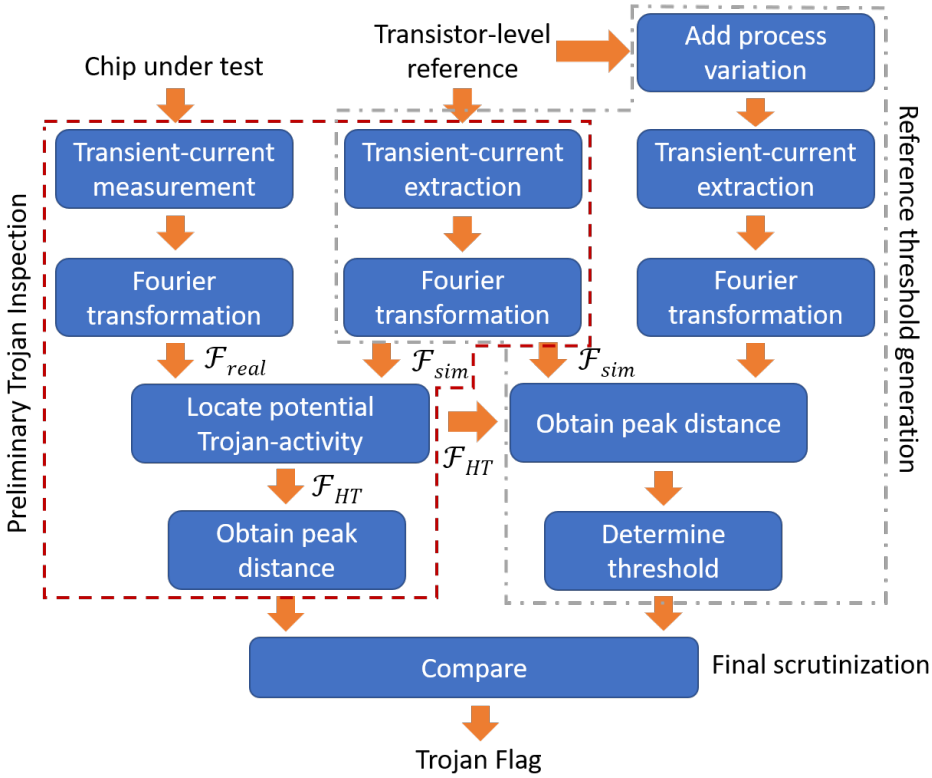


Figure 4.3: Trojan detection flow proposed in our FTAI method.

clean without noise and Trojan), noisy model (i.e., noise induced by process variation is considered), and Trojan-infected model (i.e., the triggered Trojan leaks information). As shown in Fig. 4.4, the Trojan results in a new frequency peak on the lower frequency band than the primary signal. The zoom-in view of that frequency peak indicates that the Trojan introduces a more substantial magnitude difference than the process variation noise. To facilitate Trojan scrutinization in the following phases, we define a metric, named *peak distance (PD)*, to quantify the difference in frequency magnitude between the first frequency peak induced by the Trojan and the corresponding response from the reference model.

After the preliminary Trojan inspection, a reference threshold will be applied to further examine the suspicious frequency band. As golden chips are often unavailable in practical situations, the reference threshold is provided based on simulations [55]. In this work, we apply different process variations to our transistor-level reference and obtain the corresponding frequency spectra. In each spectrum, we measure the peak distance against $\mathcal{F}_{sim}$ in the

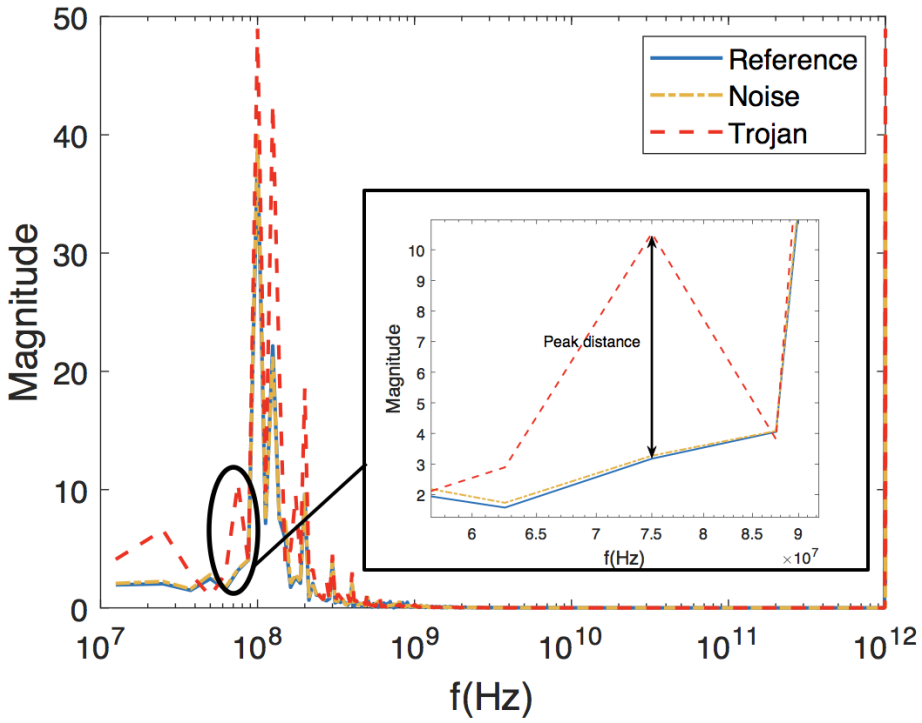


Figure 4.4: Comparison of frequency spectra for baseline, noisy, and Trojan impacted cases.

frequency band $\mathcal{F}_{HT}$. We denote the group of the peak-distance values for all the cases as PD_{noise}. It is used to evaluate the magnitude changes induced by noise on $\mathcal{F}_{HT}$. To achieve a high confidence, we apply the 3σ value of the signal PD_{noise} as the threshold PD_{th} to our Trojan detection method. The closed-form expression for PD_{th} is available in Eq. (5), where μ and σ are the mean and the standard deviation of PD_{noise}.

$$PD_{th} = \mu + 3\sigma = Mean(PD_{noise}) + 3Std(PD_{noise}) \tag{4.5}$$

In the phase of final scrutinization, the *peak distance* of the chip under examination is compared with the threshold generated in the previous phase. If the peak distance exceeds the given threshold, we conclude that there is a Trojan inserted in the chip.

4.5 Experimental Results

4.5.1 Experimental Setup

We evaluated the proposed method by using transistor-level simulations. A stacked 3D IC with three tiers is implemented in a 45nm NCSU FreePDK technology [56]. The PDN in each tier is mainly composed of a global power grid and a virtual grid. The local load circuits in each tier are multiple inverters. In our experiments, the target of the MOLES Trojan (described in Section 4.2) is an AES Sbox implemented at transistor level. We provided the input vectors satisfying the triggering condition of 3D MOLES Trojans to leak the crypto key during our experiments. We collected the transient current trace for a period of 80ns from the power-supply pin of the transistor-level 3D chip and converted the time-domain current traces to frequency spectra. We repeated the same procedure for the models of Trojan-free (i.e., *reference*), Trojan-free but considering different process variation noise (i.e., *noise*), Trojan-injected at the nominal process variation (i.e. *Trojan*), and Trojan-injected in different process variation cases.

4.5.2 Impact of Trojan size on Trojan Detection

All the experiments in this subsection were based on the nominal process variation. We first compared the proposed detection metric *peak distance* in the frequency domain, with *Euclidean distance* in the time domain. We performed the proposed spectrum analysis and identified the Trojan-related frequency peak in 75MHz. Peak distance for three Trojan sizes (MOLES20, MOLES40, and MOLES80) was measured. MOLES20 means that there are 20 registers in the MOLES ring generator. As shown in Table 4.1, the proposed peak distance is always 30× higher than Euclidean distance. This means the proposed frequency-domain analysis method can better tolerate the measurement errors and noise interference than the time-domain Trojan detection. We applied the seven process corners to the reference chip and collected their corresponding *peak distance* to form the group PD_{noise}. After following

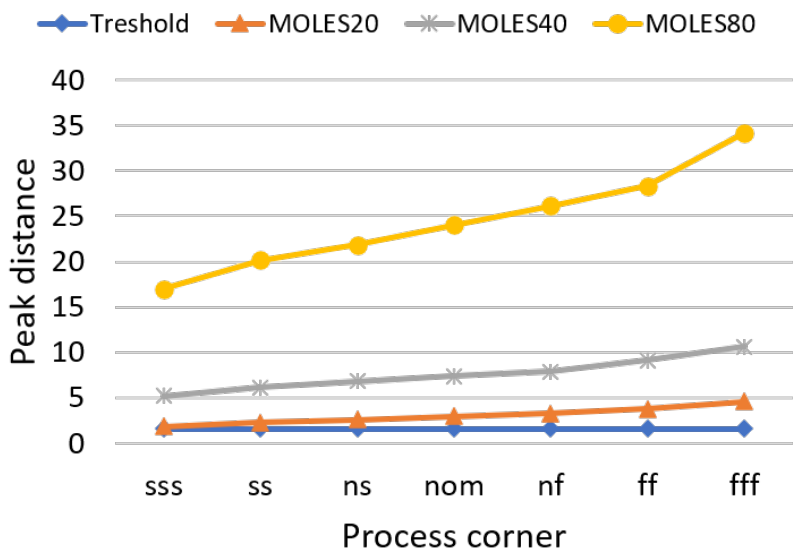
Table 4.1: Trojan detection metrics used in frequency-domain and time-domain analysis methods.

Metrics w.r.t Trojan size	MOLES20	MOLES40	MOLES80
Euclidean distance	0.0899	0.1831	0.6049
Proposed peak distance	2.986	7.367	24.007
Improved distance	33.2×	40.2×	39.7×

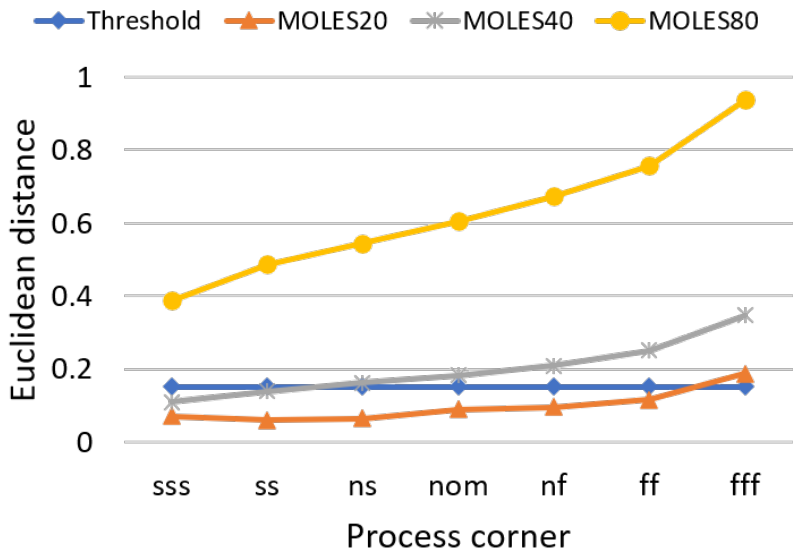
the procedure introduced in Section 4.4.3, we obtained its 3σ value of 1.578 for our frequency-domain Trojan analysis. As all measured peak distance values are greater than 1.578, our method can detect all three Trojans. In contrast, the 3σ value for the time-domain Trojan analysis is 0.1521, which is higher than the Euclidean distance for MOLES20. Thus, the time-domain Trojan detection fails in the MOLES20 case.

4.5.3 Impact of Process Variation on Trojan Detection

We further evaluated the impact of process variation on the Trojan detection success rate of our method. We conducted different test cases by applying seven process variation configurations to our 3D structure. The seven corners are sss (the slowest), ss, ns, nom, nf, ff, and fff (the fastest). The *sss* (*fff*) case doubles the progress variation from *nom* to *ss* (*ff*). The *ns* (*nf*) case is the half variation step from *nom* to *ss* (*ff*). The main variations include the long channel threshold voltage, gate oxide thickness, channel length offset, first-order body effect coefficient, and low-field mobility. As shown in Fig. 4.5(a), the peak distance of the 3D circuit tampered by Trojans with different sizes is consistently larger than the threshold, which means all the Trojans can be detected and the Trojan detection rate achieved by our method is 100%. The results shown in Fig. 4.5(b) represent the Euclidean distance obtained by the time-domain analysis method. As can be seen, the Euclidean distance for the case of MOLES20 is always below the threshold (except the fff corner). The time-domain analysis based Trojan detection also fails to detect MOLES40 in the sss and ss cases. We calculated that the time-domain method yields a 61.9% of Trojan detection rate. Thus, our proposed FTAI increases the Trojan detection rate by 38.1%.



(a)



(b)

Figure 4.5: Trojan detection effectiveness comparison between (a) frequency-domain method and (b) time-domain method at different noise levels.

4.6 Conclusion

3D IC is considered as a promising solution for future integration. However, the stacking structure and complicated fabrication process give adversaries a chance to perform malicious

attacks. Unexplored 3D Hardware Trojans can be inserted in the supply chain. Very limited works about 3D Trojan's detection and mitigation can be found in the current literature. We proposed an FTAI, which can better tolerate 3D noise than the time-domain detection method to provide a better detection rate on 3D hardware Trojans. The experimental results show that FTAI achieved a 100% detection rate on the 3D-version of MOLES. Comparing to the time-domain method, FTAI improved the detection rate by 38.1%.

CHAPTER 5

Invariance Checking based Trojan Detection Method for Three-Dimensional Integrated Circuits

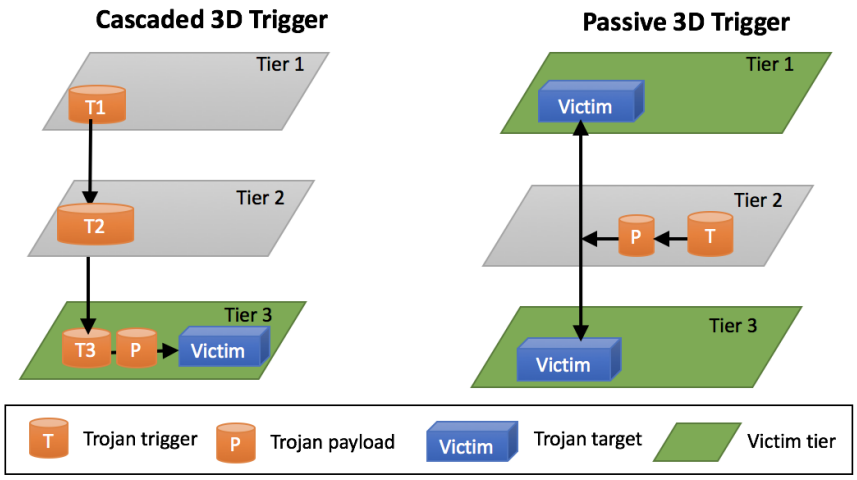
5.1 Introduction

In this chapter, we propose a run-time Trojan detection and mitigation method to complement the existing countermeasures against 2D and 3D hardware Trojans. Our main contributions are as follows: (1) our method proposes to leverage the 3D communication infrastructure, 3D-Network-on-chips (3D-NoCs), to tackle the cross-tier hardware Trojans in stacked multi-tier chips, and (2) an invariance checking method is proposed to detect Trojans, which introduce malicious NoC packets or facilitate information leak among 3D tiers.

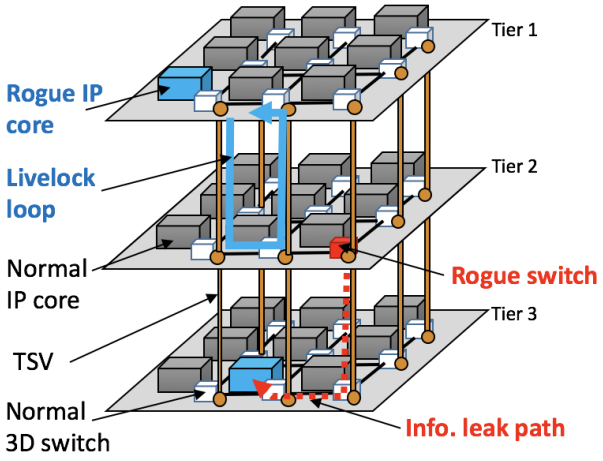
The rest of this chapter is organized as below. Section 5.2 presents the attack model interested in this work. Section 5.3 proposes a novel invariance checking method to thwart 3D Trojan insertion attacks. Simulation and synthesis results are provided in Section 5.4. This work is concluded in Section 5.5.

5.2 Attack Model

The models for representable 3D hardware Trojans are introduced in Chapter 2 and the recent work [9]. The most significant difference between 3D hardware Trojans and conventional 2D hardware Trojans is that the trigger and payload circuits of 3D Trojans are not located in the same 3D tier. Figure 5.1(a) shows examples of *Cross-Tier Trojans*. In the left case, the trigger circuits are distributed in the top and middle tiers, and they jointly trigger the



(a)



(b)

Figure 5.1: Attack scenarios considered in this work. (a) Characterization of 3D hardware Trojans, and (b) an example of the activated 3D Trojan effect [1].

payload in the bottom tier. This triggering mechanism can have a much lower triggering probability than the Trojan trigger in a single tier. In the right case, neither the trigger nor the payload circuit is located in the same tier where the victim remains; the data transmission between victims is leaked due to the Trojan in the middle tier. This type of Trojans does not interrupt normal data communication. If 3D hardware Trojans described in Fig. 5.1(a) are placed in a 3D-NoC system shown in Fig. 5.1(b), that system may suffer from livelock and information leak [1]. In this chapter, we analyze the characteristics of these two types

of 3D Trojans and propose a mitigation method accordingly.

Our 3D Trojan detection and mitigation method is based on the following assumptions: (1) each tier is a completed die (rather than a die appeared in the middle of split manufacturing), (2) the communication between tiers is at IP core level, rather than functional block level, and (3) the routing rule used in 3D routers is public to attackers.

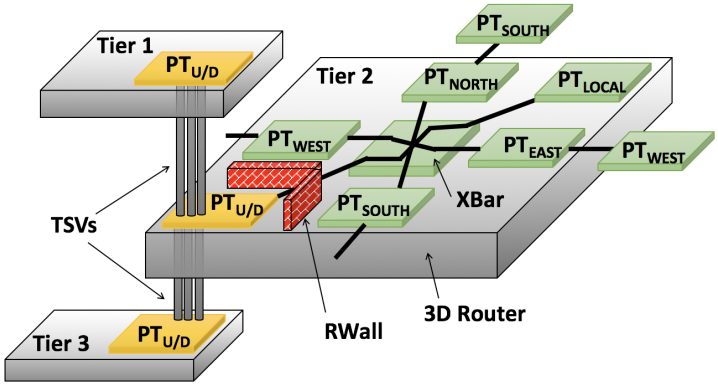
5.3 Proposed Invariance Checking Based 3D Hardware Trojan Detection and Mitigation

5.3.1 Proposed Hardened Router Architecture for 3D-NoCs

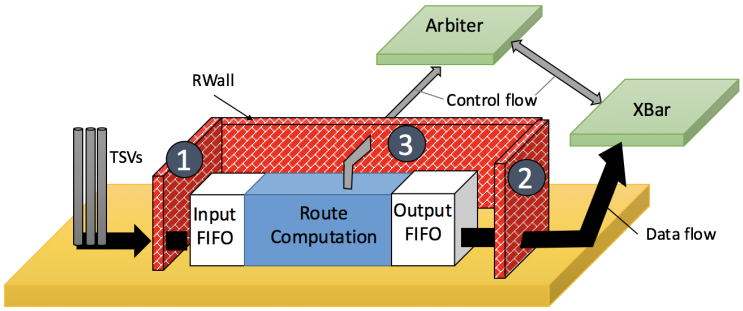
Cross-tier hardware Trojans (or multi-tier collaborative Trojans) emerge as a new hardware Trojan model for 3D ICs. Due to 3D hardware Trojans' unique threat characterizations, it is a pressing need to investigate new detection and mitigation methods specifically for 3D hardware Trojans. Moreover, the countermeasures against 3D hardware Trojans are expected to be compatible with the architecture of 3D systems. The defense mechanism should be integrated into the system specification, rather than an add-on component patched afterwards.

We propose to tackle cross-tier Trojans with a *router-hardened 3D-NoC*, which is the communication backbone for 3D integrated circuits and systems. The proposed security mechanism complements to the investigation on other 3D-NoC aspects (thermal issue [57], architecture [58], and usage in computationally intensive applications [59]). As 3D IC testing is not as thorough as 2D IC verification, there will be residual hardware Trojans, especially cross-tier Trojans, harming 3D systems after testing [9]. To address this issue, we propose a run-time Trojan detection and mitigation method against cross-tier hardware Trojans.

Figure 5.2(a) shows the architecture of proposed 3D router, in which the five ports PT_{NORTH}, PT_{SOUTH}, PT_{WEST}, PT_{EAST} and PT_{LOCAL} are used for the intra-tier communication, and PT_U and PT_D are responsible for transferring data to the upper and lower tiers, respectively. To detect and mitigate potential 3D Trojan intrusion, we propose a *RWall*,



(a)



(b)

Figure 5.2: Proposed cross-tier Trojan detection. (a) Proposed router architecture for 3D-NoC, and (b) block diagram of vertical port $PT_{U/D}$ protected with invariance checking based hardware firewall.

an invariance checking based hardware firewall, to thwart unauthorized access to the other router ports and prevent 3D-NoCs from sniffing attacks. The zoom-in view for the proposed RWall is illustrated in Fig. 5.2(b). The RWall ① examines whether a NoC flit (i.e., a basic flow unit in NoCs [45]) is tampered during its propagation from other tiers. Such tampering could be induced due to malicious through-silicon-vias (TSVs) or compromised input FIFOs. The RWall ② terminates the requests from $PT_{U/D}$ to use other ports. The RWall ③ monitors the duplication of malicious NoC packets among multiple output ports. The combined effect of ② and ③ blocks the illegal information leak and prevents the 3D communication infrastructure from being tampered at the router level.

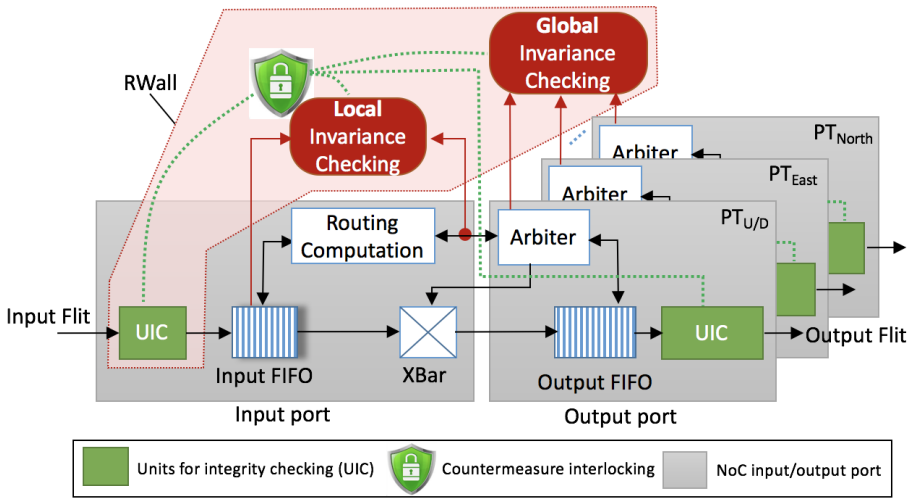


Figure 5.3: Proposed invariance checking in NoC router.

5.3.2 Proposed Invariance Checking within NoC Router

Invariance checking is a cost-effective method for fault tolerance within NoC [60]. Following that footprint, we propose to leverage the invariance within 3D-NoCs to tackle cross-tier hardware Trojans. In this subsection, we first examine the suitable invariance at the flit, port, and router levels and then develop a practical implementation algorithm. Figure 5.3 provides the detailed view of a hardened NoC router. A typical router for 2D-NoCs consists of five bi-directional routing ports, each of which is composed of input/output FIFOs, a routing computation, a crossbar (XBar), and an arbiter. For 3D-NoCs, up-stream and down-stream ports ($PT_{U/D}$) are added to access other 3D tiers. Global invariance checking examines any violation of port access among seven bi-directional NoC router ports. As our defense target is cross-tier Trojans, we pay extra attention to $PT_{U/D}$ by adding local invariance checking. The complete Trojan detection and mitigation algorithm is shown in Algorithm 2. More precisely, the proposed algorithm is implemented at the flit, port, and router levels.

At flit level, tampered flits (router inputs) will be detected by the *Units for Integrity Check* (UIC). Error control coding (ECC) is a common low-cost approach for data integrity detection. We propose to use two-level ECC based integrity check as expressed in Equations (5.1) and (5.2). In addition to encode/decode the entire flit, limited configurations of

critical flit fields will be encoded for another level integrity check.

$$UIC_{alert1} = DecFun_1(Flit) \tag{5.1}$$

In which, $Flit$ is a tuple of {flit type, flit source, flit destination, hopping path, routing priority, parity check}.

$$UIC_{alert2} = DecFun_2(flit_{type}, field_{sel}, parity_{2nd}) \tag{5.2}$$

Where, $flit_{type}$ indicates whether the flit is a header or payload, $field_{sel}$ is several selected fields for second-level integrity check (e.g., flit source and destination), $parity_{2nd}$ is the second level coding algorithm for integrity check. The two alert signals UIC_{alert1} and UIC_{alert2} from UIC will stop the malicious flit from entering or leaving the suspicious router port.

At port level, the invariance for Trojan detection includes illegal port requests and mismatched control-data flows. Only a header flit can request to reserve port-to-port connection. Any port-requests issued from other flits indicate Trojan intrusion. Since port-to-port communication is exclusive, each output port can accept one and only one request from all other input ports in the same router. Likewise, an input port cannot simultaneously issue multiple requests to access more than two output ports. Another invariance is originated from the routing history. The incoming and outgoing port request (RC_{req}) should match to packet source ($SRID$), destination ($DRID$) and the current router IDs ($CRID$). The routing inverse function expressed in Eq.(5.3) facilitates the detection of tampered routing history.

$$Local_{invar} = RInverse(SRID, DRID, CRID, RC_{req}) \tag{5.3}$$

As the information regarding the complete routing path varies with different NoC applications, the hardware Trojans inserted in 3D-NoC design time is not able to bypass all of the routing consistency check. Moreover, the invariance rules mentioned above are not mutable

Algorithm 1: Proposed multi-level invariance check.

```
Data: Packets through a 3D-NoC router
Result: Alert for 3D Trojan intrusion
UIC_alert1 (Input flits);
UIC_alert2 (Selective flit breakdown fields);
while Cross-tier packets being transferred do
    //Local invariance checking;
    if Σ(RC_req from PT_U/D) > 1 then
        Information leak detected;
    else
        if Σ(PT_U/D ⇌ PortFIFOs) > 1 then
            Intrusion attack detected;
            Terminate cross-tier communication;
        else
            //Global invariance checking;
            if RInverse outputs mismatch RC_req then
                Intrusion attack detected;
                Drop malicious flits;
            else
                Pass local invariance check;
            end
        end
    end
    Use encryption key to unlock arbiter tables;
end
```

once the router is power up. Thus, our invariance checking does not only detect malfunctions but also monitors illegal behaviors triggered by 3D hardware Trojans.

At router level, our method examines the invariance available among arbiters. In the baseline, the arbiter grants one of the port requests based on even opportunity (i.e., round robin rule). Updating on the round-robin register tables has to satisfy the priority rule. Any interrupts appeared in the middle of packet transmission indicates the occurrence of an attack. Logic encryption [61] is adopted to harden the round-robin tables. In our case study, we use a 7-bit key to unlock the updating logic for arbiters in 3D routers. The incorrect encryption key will terminate the arbiter's normal function.

Table 5.1: Comparison of Area, Delay and Power

Metric under comparison	Baseline [8]	Proposed	Overhead
Area (μm^2)	19731	21005	6.46%
Delay (ns)	0.86	0.94	9.30%
Dynamic power (mW)	13.0733	13.5646	3.76%
Leakage power (μW)	108.0194	115.6355	7.05%

5.4 Experimental Results

5.4.1 Area, Power, and Delay

We implemented the proposed 3D NoC router in Verilog HDL and synthesized the HDL code in Synopsys Design Compiler with a 45nm NCSU openPDK technology. The flit width for the NoC is 32 bits. The input and output FIFOs are 32-bit single-depth buffers. Round-robin arbitration was used in the router arbiter. We set the clock frequency to 1 GHz. The area, delay and power consumption for the baseline [8] and our method are compared in Table 5.1. As shown, our method is a lightweight countermeasure. The area is only increased by 6.49%. The overhead on dynamic power and leakage power are 3.76% and 7.05%, respectively. As we add invariance checking in the cross-bar unit, the worst-case delay of our router is 9.3% higher than that of the baseline.

5.4.2 Trojan Detection Rate

The proposed invariance checking examines the consistency between the port requests and the routing history to detect 3D hardware Trojans. We randomly altered the port request to access upper and lower tiers (i.e., attack on router port requests) or the destination router ID carried in the NoC header flit (i.e. attack on destination router ID). Each Trojan detection rate was obtained from 10,000 simulations. As shown in Fig. 5.4, the Trojan detection rate of our method is above 94%, no matter the Trojan attack is on the port request signals or the destination router ID.

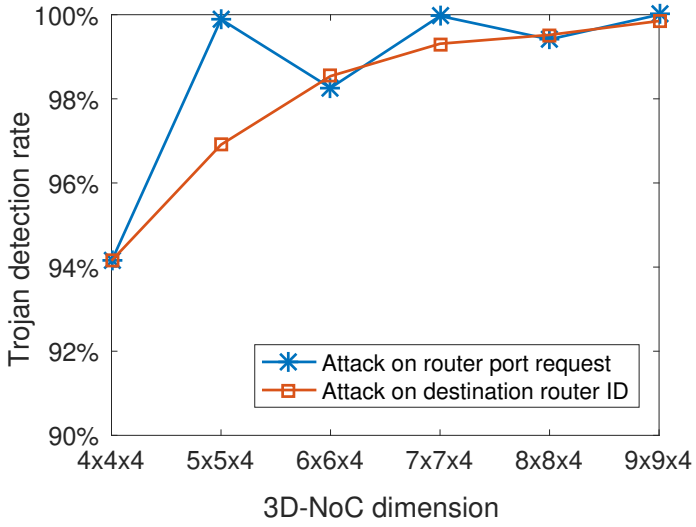
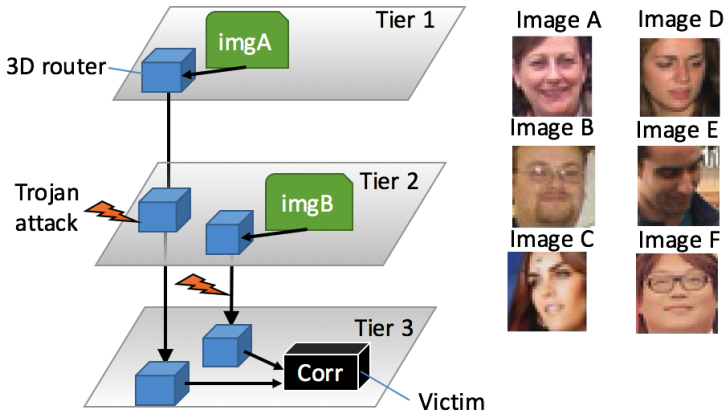


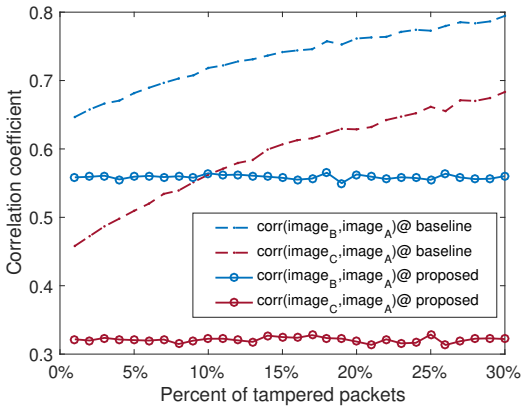
Figure 5.4: Trojan detection rate of proposed method.

5.4.3 Impact of Cross-tier Trojan Mitigation on Image Authentication in a 3D system

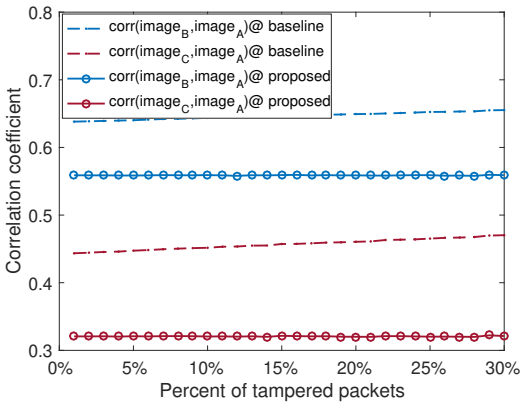
In our case study, we used a 3D-NoC to perform image based authentication. The experimental setup is shown in Fig. 5.5(a). Through the 3D NoC routers, tier 1 and tier 2 transmit two images to tier 3 for image authentication. Pearson correlation coefficient (PCC) is adopted as the metric to indicate whether the two images from tiers 1 and 2 depict the same person. Hardware Trojan insertion happens in the 3D router located in tier 2 or the TSVs connecting tiers 2 and 3. The activated Trojan tampers the header flits or payload flits of the image packets. The proposed method filters out the tampered flits. If a header flit is altered by a 3D hardware Trojan, the entire targeted packet is replaced by a malicious packet (baseline) or dropped with notifications (proposed). If a payload flit is sabotaged by a 3D hardware Trojan, only that flit is substituted by a dummy flit (baseline) or deleted (proposed) and the rest flits in that packet remain the same. The PCC between images from tiers 1 and 2 are computed in the victim unit (i.e., Corr in Fig. 5.5(a)). As shown in Fig. 5.5(b) and 5.5(c), our scheme removes malicious flits significantly and thus reduces the correlation coefficient. This means that the tampered images are less likely to pass the authentication. For instance, the proposed method can reduce the PCC from 0.6755 to 0.3122. As each



(a)



(b)

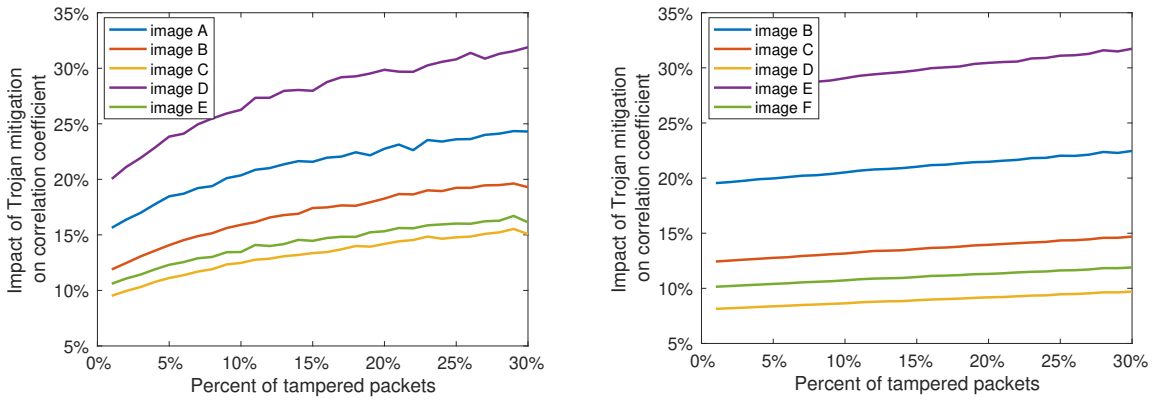


(c)

Figure 5.5: Impact of Trojans on the application of 3D image authentication. (a) attack scenario, (b) impact of attacking header flit on correlation coefficient, and (c) impact of attacking payload flits on correlation coefficient.

NoC packet is composed of one header flit and multiple payload flits, the baseline scheme is more sensitive to Trojan attacks aiming at header flits than at payload flits. In contrast, our Trojan mitigation overcomes that sensitivity.

We examined the Trojan mitigation effect with six images shown in Fig. 5.5(a). Images B, C, D, E, and F are correlated with image A (after Trojan detection and mitigation). As shown in Fig. 5.6(a), the proposed method can reduce the PCC by 31%. As the percent of tampered packets increases, our mitigation method will further reduce the correlation coefficient. The exact amount of reduction on correlation coefficient varies with the images used in authentication.



(a) (b)

Figure 5.6: Reduction on correlation coefficient achieved by Trojan mitigation.

5.5 Conclusion

The emerging 3D integration techniques potentially bring in attack surfaces for new type of hardware Trojans, cross-tier 3D Trojans. Given the 3D Trojan models published in recent literature, this chapter proposes to leverage 3D-NoC architecture to detect and mitigate the newly characterized hardware Trojans. Invariance on port access and routing history is exploited in this work to perform run-time Trojan detection. Simulation results show that the proposed method achieves a high Trojan detection rate at minor cost on area and power consumption.

CHAPTER 6

Improving Power Analysis Attack Resistance using Intrinsic Noise in 3D ICs

6.1 Introduction

In this chapter, we extend our groups's early work [2] by providing the second practical implementation method to alter the power supply of complete AES crypto module in FPGA and validate the correlation power analysis (CPA) attack resilience of our method. We propose to group the supply voltages from different 3D tiers temporally to drive the cypto module. In this way, the noise from 3D power distribution network (PDN) is induced to the crypto module to blur the correlation exploited by CPA attacks. We name this new method temporally varied supply voltage (TVSV).

6.2 Preliminary

6.2.1 Early Work of Using 3D PDN Noise to Mitigate CPA Attacks

In the work [2], the intrinsic noise within a 3D PDN has been proved to be additive noise. Furthermore, a countermeasure which utilizes the PDN noise to mitigate CPA attacks is introduced. As shown in Fig. 6.1, the original crypto unit is divided into multiple submodules. The supply voltages, V_{DD}s, from different 3D tiers are used to drive the submodules individually. In this way, the noise from different tiers are induced to the crypto unit and thus blur the power traces collected for CPA attacks.

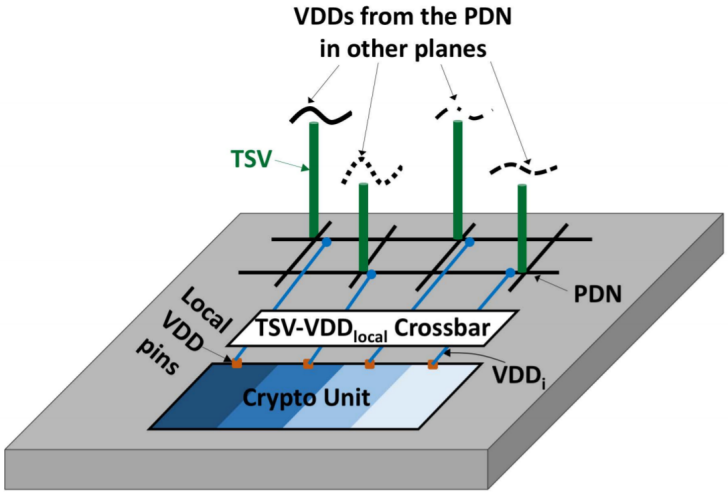


Figure 6.1: Countermeasure introduced in [2]

6.2.2 CPA Attack

CPA is an advanced power analysis attack for the crypto key retrieval from the hardware implementation of encryption systems. It leverages the correlation between the crypto key and the switching activities of the crypto hardware module to significantly shorten the time spent on the key guessing via brute force attempts. In a CPA attack, attackers will calculate the outputs of the encryption system and adopt Hamming distance/Hamming weight model to generate hypothetical power consumption [13]. Then the Pearson correlation coefficient (PCC) [62] is utilized to retrieve the secret key.

6.3 Proposed TVSV against CPA Attacks

6.3.1 Theoretical Foundation of TVSV

As introduced in [2], a circuit's power consumption can be modeled with Eq. (6.1).

$$P_{orig} = \alpha f C_L V_{DD}^2 \tag{6.1}$$

In which, α, f, C_L, and V_{DD} are switching activity factor, system clock frequency, load capacitance, and supply voltage, respectively. If we vary the supply voltage temporally, which means we change the supply voltage along with the time. The new power consumption can be formed as in Eq. (6.2).

$$\begin{aligned} P_{new} &= \frac{\sum_{i=1}^{N} \left(\alpha f C_L V_{DDi}^2\right)}{N} \\ &= \frac{\sum_{i=1}^{N} \alpha f C_L \left(V_{DD}+\Delta V_i\right)^2}{N} \\ &\approx P_{orig} + 2\alpha f C_L \cdot \frac{\sum_{i=1}^{N} \left(V_{DD}\cdot \Delta V_i\right)}{N} \end{aligned} \tag{6.2}$$

In which, N is the number of different supply voltages changed during the time period of interest. V_{DDi} represents each different supply voltage. We can alter V_{DDi} in a completely or periodically random fashion. The latter one requires less number of diverse V_{DDi}s, but it is less effective than the former method in regard to the resilience against CPA attacks. Figure 6.2 shows the application of periodically random noise that helps to reduce the correlation coefficient over the constant nominal supply voltage. The application of completely random noise leads to an approximately flat PCC even though the variance of multiple noises is in wide range.

6.3.2 Implementation of TVSV

We multiplex multiple voltage sources to power up the *entire* crypto unit, rather than multiple sub-units as what is proposed in [2]. As shown in Fig. 6.3, the four V_{DD}s from nearby planes are fed to a multiplexer *MUX*. At each period of time, only one of these V_{DD}s will be selected to drive the crypto unit. A dynamic rotator is used to control the multiplexer. The role of multiplexer is to assign varied supply voltages to the crypto unit at different time slots in a complete process of running the cryptographic algorithm. Figure 6.4 demonstrates the AES power consumption at three time periods. The values of sampling power are distin-

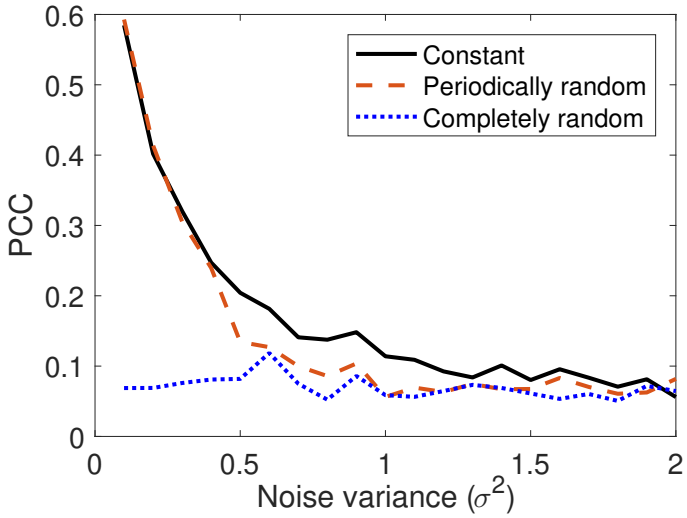


Figure 6.2: Impact of the combination of multiple additive noises on correlation coefficient.

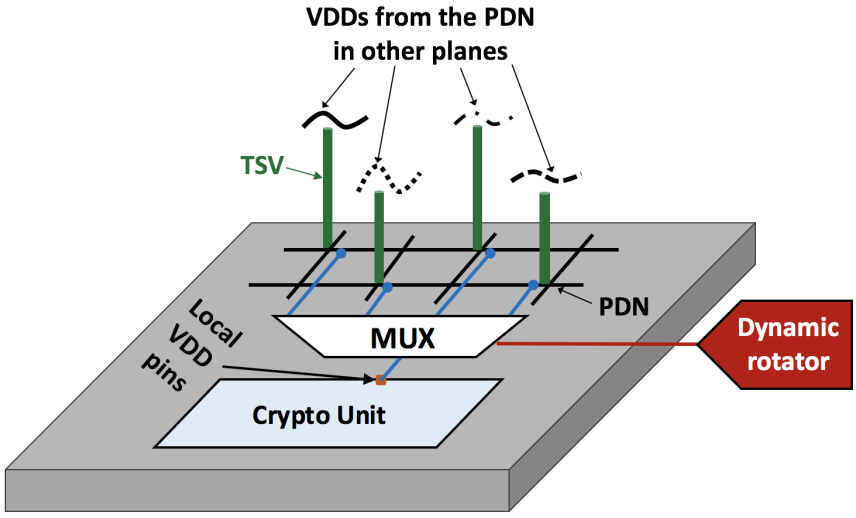


Figure 6.3: Proposed countermeasure multiplexing multiple voltage sources for the entire crypto unit.

guished from each other. This indicates the power traces captured through CPA attacks are altered by the voltage noise. Modification on the power consumption will impact the CPA efficiency.

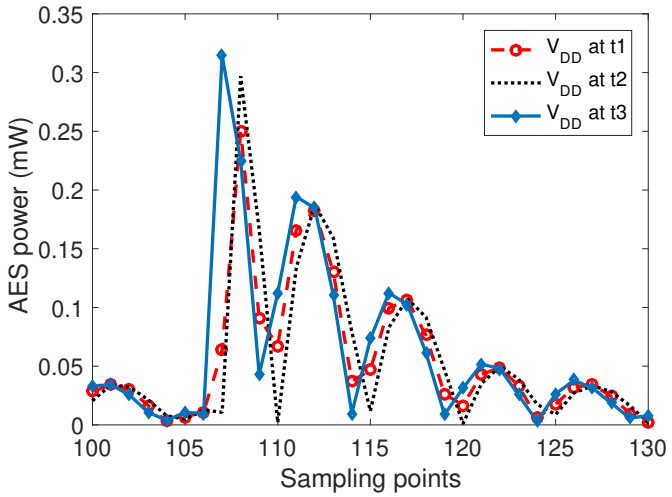


Figure 6.4: AES power profiles measured at three different operation periods.

6.4 Experimental Results

6.4.1 Experimental Setup

We evaluated the proposed method by using FPGA emulation. We used a SAKURA-G FPGA board. That board contains two Spartan-6 FPGAs: one (LX75 FPGA) for a cryptographic implementation and the other (LX9 FPGA) for power traces capturing. The bitsream asociated with the Verilog-HDL code for AES-128 was downloaded to the SAKURA-G board. A Python-based ChipWhisperer [13] software was used to perform power trace capturing and analysis. The other setup for the CPA attacks can be found in our prior work [14].

6.4.2 Improved Resilience against CPA Attacks

For AES-128, there are 16 key bytes in total. The main FPGA Spartan-6 XC6SLX75 is powered by a supply voltage from the $VCCINT$ pin. As the FPGA does not support multiple supply voltages, we adjusted the value of VCCINT through a trimmer $VR1$ to generate different supply voltages for the proposed TVSV implementation. The supply voltage for AES was monitored by a multimeter through the on-board pin $J1$. We first collected a set of AES power traces for each supply voltage at separate intervals and then

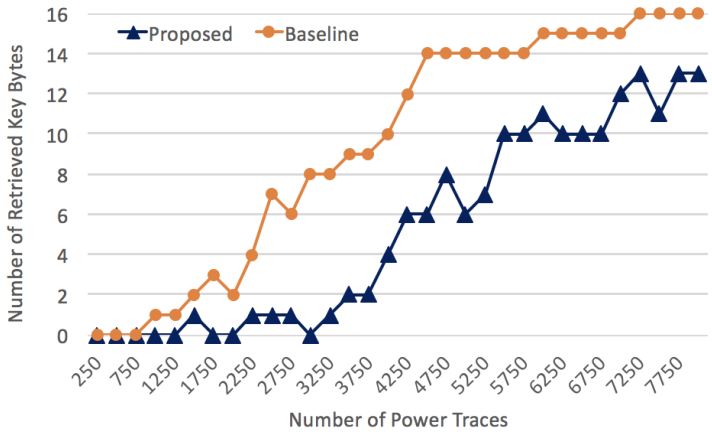
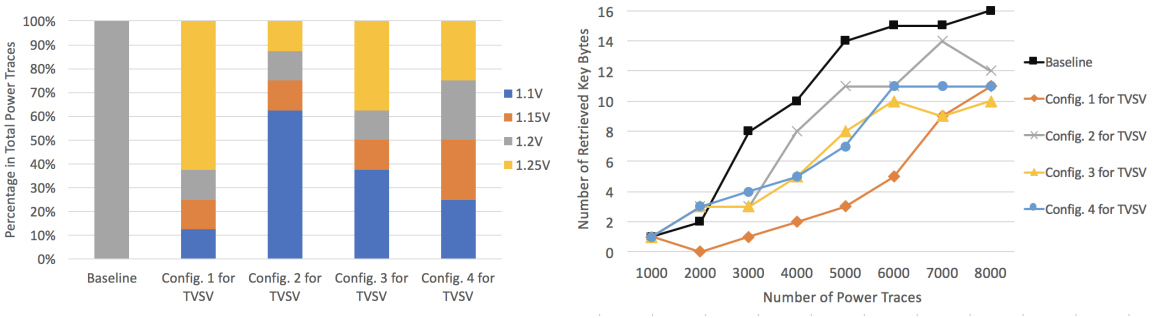


Figure 6.5: Reduction on the number of retrieved key bytes achieved by the proposed method.

combined the multiple sets of the collected power traces in the ChipWhisperer Capture and Analyzer tool during the process of CPA attack. In our emulation, we selected four voltage levels: 1.1V, 1.15V, 1.2V and 1.25V (the standard value of VCCINT is 1.2V).

Key retrieval speed: The ChipWhisperer Analyzer retrieves the correct key for the AES-128 by validating the crypto key byte by byte. Given a fixed number of power traces, the less number of retrieved key bytes means a better resilience achieved by the countermeasure against CPA attacks. At each voltage level, we generated eight ChipWhisperer projects, each including 250 power traces. Then, we combined all power traces for different supply voltages and different ChipWhisperer projects to form a complete power profile set for the CPA attack on AES-128. As shown in Fig. 6.5, the proposed countermeasure effectively thwarts the CPA attack. For the given 8000 power traces, the CPA attack is not able to retrieve all 16 key bytes for the AES-128 protected with proposed method. In contrast, the baseline leaks the crypto key with a rapider speed than our method. On average, our method leaks 4.25 less key bytes than the baseline. Note, the AES-128 only has 16 key bytes and hence 4.25 is a large portion of the total cryto key vector. The CPA attack on the baseline was based on the 8000 power traces collected from the AES-128 operating at the supply voltage of 1.2V.

The power traces for the experiment in Fig. 6.5 are *evenly* contributed by four different

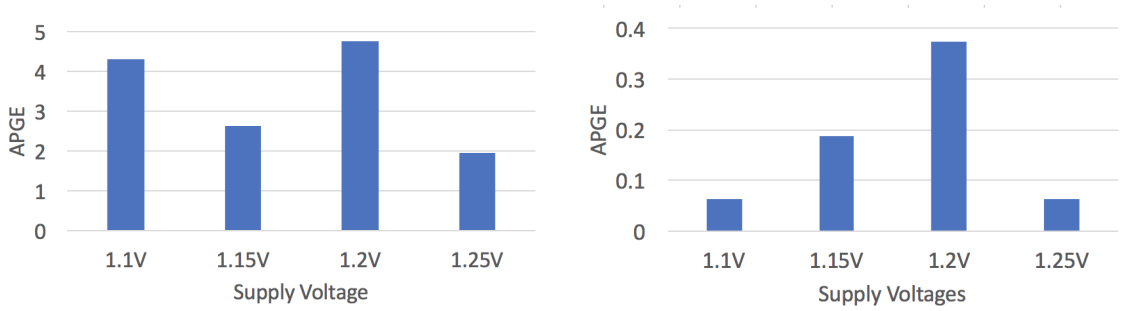


(a) (b)

Figure 6.6: Comparison of CPA key retrieval speed. (a) Power trace configuration and (b) Number of retrieved key bytes for different number of power traces.

supply voltages. We further examined whether other combinations of the power traces collected from different supply voltage scenarios will lead to a different key retrieval speed. In addition to the baseline (all traces with 1.2V), we assembled the power traces with the percentage shown in Fig. 6.6(a). For instance, in the configuration 1 (i.e. Config. 1 for TVSV), the dominate power traces are contributed by the case running the supply voltage of 1.25V. As shown in Fig. 6.6(b), no matter which configuration is used, our countermeasure reveals less number of key bytes than the baseline. In addition, none of our configuration allows the CPA attacks to retrieve all 16 key bytes within 8000 power traces. This further confirms the proposed countermeasure indeed impacts the key retrieval efficiency of CPA attacks.

Partial guessing entropy: To find out the reason behind the observation in Fig. 6.6(b), we studied the partial guessing entropy (PGE) for each key byte. A smaller PGE means less number of guessing is needed to identify the correct key byte. The accumulated PGE (APGE) represents the total number of guesses that may take to retrieve the entire crypto key. We examined the impact of different supply voltages on APGE. Figures 6.7(a) and (b) show the APGEs for 16 AES key bytes based on the analysis of 4000 and 5000 power traces, respectively. After comparing these two cases, we conclude that the general trend of APGE for each supply voltage decreases when more power traces are analyzed. However, there is



(a) (b)

Figure 6.7: APGE obtained in CPA attacks based on (a) 4000 and (b) 5000 power traces.

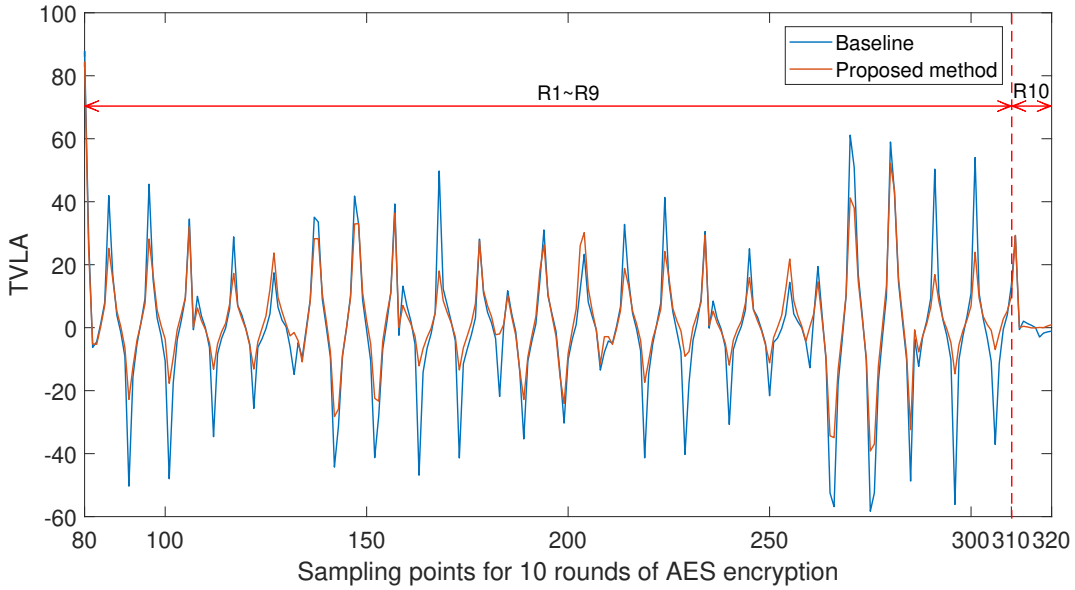


Figure 6.8: TVLA comparison between basic and our secured AES.

no obvious clue that indicates which voltage offers a better resilience against CPA attacks. That explains why it is not clear which configurations used in the experiment for Fig. 6.6(b) is the best in terms of resilience against CPA attacks.

Test vector leakage assessment: Leakage detection is important to validate the physical security of cryptographic devices. Test vector leakage assessment (TVLA) [63] approach is one of the popular technique to detect the leakage. In this method, a set of preselected test vectors is selected and then a statistical tests are performed on collected power measurement. The test results into a confidence score using which a fail/pass decision can be

made for the crypto under test.

We conducted the TVLA on the basic and proposed TVSV secured AES. Our goal is to evaluate to what extent the data-dependency of the AES power traces can be mitigated by our method. Each power trace collected by ChipWhisperer Capture consists of 396 time instants, representing 396 sampling points. Based on all the 8000 power traces, collected for the results shown in Fig. 6.5, we calculated the TVLA value for each time instant and plotted in Fig. 6.8. R1 to R10 represents the AES first to 10th round. As shown in Fig. 6.8, the TVLA absolute values for TVSV secured AES are generally lower than those for the baseline AES for 10 AES rounds (roughly from 80 to 320). A smaller TVLA absolute value means higher confidence to accept the null hypothesis [64]. We evaluate this confidence level with a probability Pr_{con} as expressed in Eq. 6.3.

$$Pr_{con} = 2 \int_{TVLA_{lr}}^{\infty} pdf(t, v)dt \tag{6.3}$$

In which, $pdf(t, v)$ is the probability density function of the Student's t distribution with the degrees of freedom of v. t is the t-test statistic and we simply use 16000 (8000 traces + 8000 traces) for v.

Because the previous results shown in Fig. 6.5 are obtained from the AES last round attack performed in ChipWhisperer Analyzer, we zoom in the TVLA values for the time instants observed in the AES last round. Those TVLA absolute values (roughly from 310 to 320) were averaged and saved in $TVLA_{lr}$. The corresponding Pr_{con} was also adopted to quantify the mitigation ability against CPA attack. The values of $TVLA_{lr}$ and Pr_{con} are listed in table 6.1. The $TVLA_{lr}$ for basic AES is 4.9477, which is greater than 4.5. Note, 4.5 is defined as a threshold to determine whether the traces carry sensitive information [65]. The $TVLA_{lr}$ of our proposed method is below the threshold of 4.5. This result indicates that our approach is less data dependent and leaks less sensitive information (i.e. key) than the baseline. Our method also improves Pr_{con} over 201$\times$ over the baseline.

Table 6.1: $TVLA_{lr}$ and Pr_{Con} for basic and our secured AES.

AES versions \ Result categories	$TVLA_{lr}$	Pr_{Con}
Baseline	4.9477	7.58×10^{-7} (100%)
Proposed method	3.7894	1.52×10^{-4} (201%)

6.4.3 Overhead on Power

The entire AES was implemented in the SAKURA-G FPGA board and the ChipWhisperer tool captured 8000 power traces for each supply voltage (i.e. 1.1V, 1.15V, 1.2V, and 1.25V), respectively. The average power consumption for each power trace was calculated in MATLAB. The baseline power is the one using 1.2V. Four configurations shown in Fig. 6.6(a) were adopted. We analyzed the power traces and balanced the power trace at the module level and round level. Module level power balancing is achievable by using differential CMOS logic (such as the method in [66]), which make each module consume the same power regardless of which input pattern is applied. We used the AES round consuming the highest power to replace the power profile for the remaining AES rounds, and calculated the average power for the module level power balancing. Round level power balancing can be realized by the current equalizer (such as the method in [67]). We assume the current equalizer technique compensates the fluctuation on the AES current throughout the entire AES round operation such that the AES power remains as high as the highest dynamic power observed in different AES rounds. We compared the proposed method with the baseline, module level and round level power balancing approaches and show the power reduction achieved by our method in Fig. 6.9. As our method does not introduce additional noise to flatten the power, our method can significantly reduce the power over the power balancing techniques. Depending on the TVSV configuration pattern, the power reduction achieved by our method is in the range of

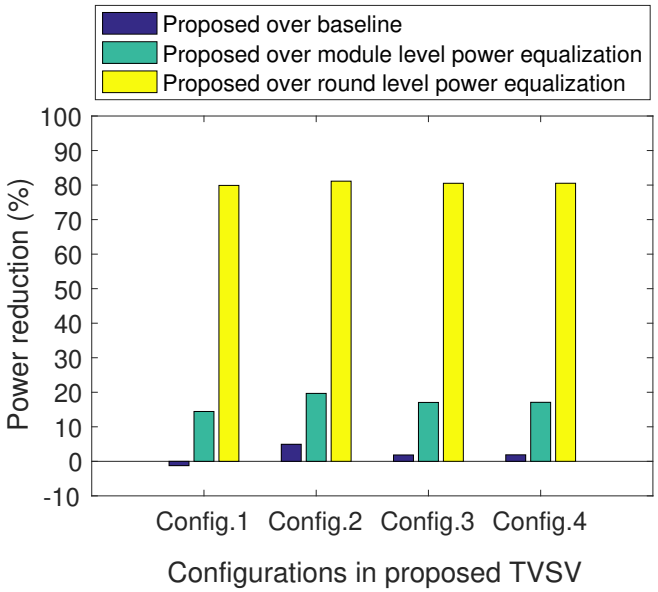


Figure 6.9: Power reduction comparison.

14.4% to 19.7% at the module level, and in the range of 79.9% and 81.1% at the round level. Since the TSV noise could lead the supply voltage exceed the nominal voltage, our method consume more power by 1.3% than the baseline in the scenario of configuration 1 (that is why power reduction is negative). For other three configurations, our method reduces the power by 1.8%∼4.9%.

As different detailed settings used in different approaches [66–68], we could not repeat the exact same experiment in our FPGA platform. We cited their reported power overhead and compared those numbers with ours in Table 6.2. As shown, the algorithmic approach [68] leads to 4.0× overhead on power, SABL consumes 1.9× power on AES, and the switched capacitor current equalizer brings in 33% more power consumption. Instead of relying on artificially induced noise, our method exploits the inherently existing noise to reduce the power correlation. Thus, we can effectively reduce the power consumption of the crypto module. Our case study shows that the proposed method leads to a power overhead no more than 1.25% over the baseline.

Table 6.2: Comparison of Power Overhead.

Methods	QuadSeal [68]	SABL [66]	Current Equalizer [67]	Proposed
Power overhead	+4.0×	+1.9×	+33%	-81.1%∼+1.25%

6.5 Conclusion

We extended our group's previous work of utilizing 3D PDN noise to mitigate CPA attack. We proposed a TVSV method in this chapter to induce the noise to the victim crypto module by combining the supply voltages from different 3D tiers temporally to drive the cypto module. Emulation on an FPGA platform prove that the proposed implementation method consumes significantly less power than the existing power balancing techniques. Our method reduces the power overhead by up to 81.1% over the round-level power balancing technique. The TVLA indicates proposed method reduces the risk of leaking sensitive information through power traces and that shows the improvement on CPA resilience.

CHAPTER 7

Towards Enhancing the IP Security of ICs and 3D ICs: Addressing the Resilience Against Power Analysis Attacks on Logic Locking

7.1 Introduction

To reduce the time-to-market and manufacturing cost, IPs are commonly used in IC designing. However, the fact that the modern IC manufacturing is often outsourced brings security threats (e.g. IP piracy) to the supply chain [69]. The untrusted IC designers or foundries in the supply chain have access to the IPs and they can overuse them for their own profit. Moreover, with the access to the GDSII file of the IPs, adversaries can even reverse engineer it to retrieve the original design.

To mitigate the impact caused by IP piracy, split-manufacturing and logic locking based countermeasures are presented in the existing literature [10, 69, 70]. Split-manufacturing proposes to split the layout of a design into the Front End Of Line (FEOL) and Back End Of Line (BEOL), each of which is sent to different foundries to fabricate [10]. Thus, each individual foundry only has partial knowledge of the design and they cannot make a counterfeit product even if they hold the IPs. However, split-manufacturing will not help on protecting M3D IPs because all the components of a M3D IC are fabricated by the same foundry. To protect the IP security of M3D ICs, logic locking could be a better defense mechanism which proposes to encrypt the netlist of the original IC and the encrypted chip will only be activated with the locking key after the fabrication. In this way, malicious foundries can not fully extract the logic function of the chip even with the access of its GDSII design details. In another word, even attackers might successfully reverse engineer

the chip to make counterfeits, the counterfeits cannot obtain the original function of the chip and that can effectively thwart IP piracy attacks. Logic locking techniques usually insert key-controlled logic gates or transistors to the netlist [3, 71] and will maintain or alter the logic of the locked chip depending on the key inputs. They can protect M3D IPs but it is critical to protect the key information from eavesdropping attacks, such as power analysis attacks.

Logic locking based encryption algorithm is considered as having natural defense against power analysis attacks [71]. This is because the locking keys are inserted in different places through out the entire chip under protection and contribute to the dynamic power consumption at different instants, which makes the sampling and alignment of the power traces difficult. However, there is limited work evaluating this resilience quantitatively. The work [71] proposed the first algorithm of performing differential power analysis (DPA) attacks to logic locking. According to the results reported, the DPA attack is able to retrieve part of the locking keys. With the increase of the number of keys, however, the computing complexity of the attack becomes higher while the key retrieving can be harder. This shows that logic locking is indeed resilient to the DPA attack. In this chapter, we evaluate the resilience of logic locking against a more powerful and more efficient power analysis attack, correlation power analysis (CPA) attack. Furthermore, we propose a logic-cone conjunction (LLC) based method and a key insertion guideline for the transistor-level logic locking to improve its CPA resilience.

7.2 Preliminary

7.2.1 M3D IC

Different from the stacked 3D IC, a M3D IC only has one silicon substrate and the components of all the M3D layers, including transistors, poly-silicon and metal, are fabricated on the substrate sequentially. The inter-lay connection uses monolithic inter-tier vias (MIVs) instead of TSVs. Compared to TSVs, MIVs' fabrication involves similar materials and pro-

cesses. However, MIVs are usually smaller in size which facilitate M3D ICs having even better performance than TSV-based stacked 3D ICs. On the other hand, M3D layers are fabricated on one single wafer in one foundry. In this case, the split manufacturing strategy cannot be applied to protect M3D ICs from IP piracy attacks. To solve this security problem, logic locking-based methods have been investigated by researchers.

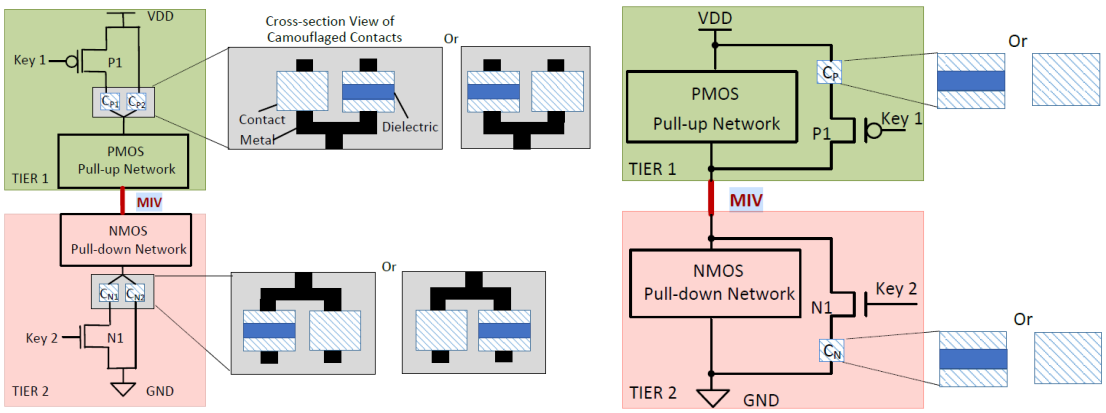
Logic locking is an encryption technology for securing the original logic function of digital circuits from IP piracy attacks.

7.2.2 Conventional Gate-Level Logic Locking

Conventionally, logic locking [17] inserts key-controlled gates to the original design that needs to be protected, as know as the gate-level logic locking. The specific implementation of gate-level logic locking varies with the locking goal, such as obtaining higher output corruptibility or better resilience against key-retrieval attacks. For example, the fault analysis-based logic locking (FLL) inserts the key-controlled gates to the locations, which have the highest fault impacts on achieving the maximum output corruptibility [72]. A strong inference-based logic locking (SLL) is proposed in [73] to thwart the key sensitization attack. The work [74] uses multiplexers, instead of XOR/XNOR gates, to expand the logic cone size and thus improve the defense capability against cone-based brute-force attacks.

7.2.3 Transistor-Level Camouflaged Logic Locking

In contrast, the transistor-level logic locking internally modifies the existing logic gates of the original circuit by injecting key-controlled transistors. Comparing to the gate-level logic locking, the transistor-level logic locking usually incurs much less overhead that is caused by the extra logic gates. For example, the transistor-level camouflaged logic locking method introduced in [3] protects the IP security of M3D ICs by inserting parallel or serial locking transistors and camouflaged contacts in M3D tiers. Fig. 7.1 shows two styles of this locking scheme: serial locking shown in Fig. 7.1(a) and parallel locking shown in Fig. 7.1(b). More



(a) (b)

Figure 7.1: The transistor-level camouflaged logic locking in (a) serial locking and (b) parallel locking styles [3].

specifically, the locking scheme has four specific configurations including PMOS serial locking (PSL), NMOS serial locking (NSL), PMOS parallel locking (PPL), and NMOS parallel locking (NPL). The locked network only functions normally when the correct locking key is provided otherwise, will generate a floating or constant output.

Because our research focuses on evaluating and improving the attack resilience of the logic locking techniques in M3D ICs, this transistor-level camouflaged logic locking method will be investigated further in the chapter. We simplify the two styles to create a follow-up version of the transistor-level logic locking and an example of a NAND gate locked in the simplified version is shown in Fig. 7.2. The simplified version also has two styles: PMOS serial locking plus NMOS parallel locking (*PSLNPL*) and PMOS parallel locking plus NMOS serial locking (*PPLNSL*). The wrong key in the PSLNPL style will lead the NAND gate output to be a constant 0. Likewise in the configuration of PPLNSL, the wrong key will yield a constant 1 at the output of NAND.

7.2.4 DPA and CPA Attacks

CPA attack has been introduced in Chapter 6. Besides CPA, differential power analysis (DPA) attack is anther strong power analysis attack. It was first proposed in Paul Kocher's

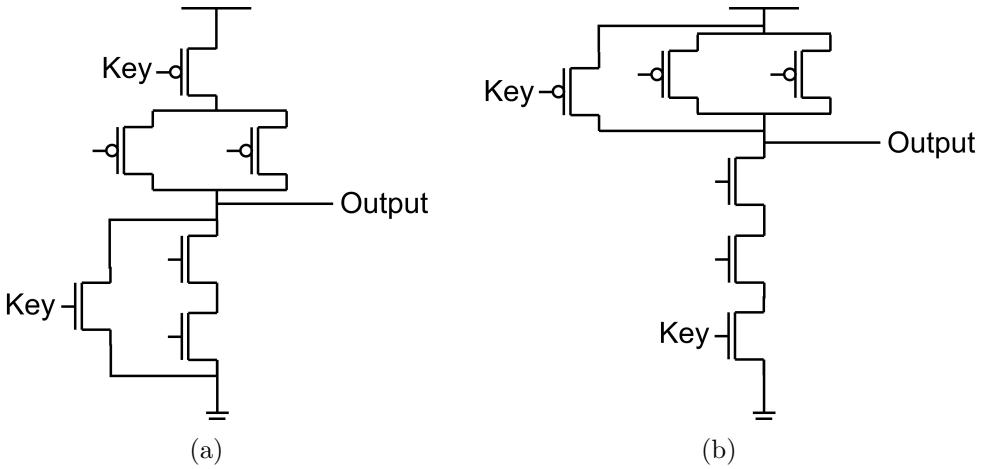


Figure 7.2: A NAND gate locked with the transistor-level logic locking in (a) PSLNPL and (b) PPLNSL configurations.

paper [75] in 1999 and has drawn great attention over the last two decades. Similar with CPA, DPA also exploits the fact that the power consumption of a chip is correlated to its internal data being processed to retrieve the secret key applied in the crypto hardware module. Attackers need to collect the power traces from their target chips that run the encryption algorithm with an unknown secret key and a set of known input patterns. The same input patterns will be used to calculate the outputs of encryption with a guessing key. Based on the collected power traces and the calculated outputs for different guessing keys, a statistical metric *differential trace* can be generated to guide attackers to determine which guessing key is the correct one applied in the hardware crypto module.

Theoretical Difference Between DPA and CPA

Assume that the target encryption process is $\mathcal{E}(p, k)$, in which $\mathcal{E}$ stands for the encryption algorithm and it is usually also defined as a selection function. The variables p and k represent the plaintext and the encryption key, respectively. For a given plaintext, the corresponding ciphertext c equals to $\mathcal{E}(p, k)$. Attackers will randomly guess a key, k_{guess}, and then calculate a set of ciphertexts, C, with regards to a group of plaintexts P. The same process is repeated for different k_{guess} while the set of P remains the same. The same plaintexts P will also be

applied to the real chip to produce the corresponding power traces T. Thus, each guessing key k_{guess} will have a pair of the ciphertext set C and the power trace set T.

In DPA attacks, depending on each element c in the set C equal to 0 or 1, each single power trace t in T is first classified to one of the two sets $T0$ and $T1$. Next, the DoM defined in Eq. 7.1 is calculated for each guessing key k_{guess}. The differential trace is also known as difference of means (DoM).

$$DoM = |\overline{T0} - \overline{T1}| \tag{7.1}$$

In which, $\overline{T0}$ and $\overline{T1}$ are the averages of $T0$ and $T1$, respectively. The k_{guess} that yields the highest DoM is considered as the correct key by the DPA attack.

In CPA attacks, a hypothetical power consumption T_{hyp} is estimated based on the set C for each k_{guess} and the power estimation model, Hamming distance or Hamming weight model [13]. Equation 7.2 describes how the PCC between T_{hyp} and T is calculated.

$$PCC = \frac{E[T_{hyp} \cdot T] - E[T_{hyp}] \cdot E[T]}{\sqrt{E[T_{hyp}^2] - (E[T_{hyp}])^2} \cdot \sqrt{E[T^2] - (E[T])^2}} \tag{7.2}$$

Each guessing key k_{guess} will have a corresponding PCC. The k_{guess} that has the highest PCC will be considered as the correct key by the CPA attack.

According to the experimental results from [76], the incorrect key guesses may generate spikes on the differential trace for DPA, known as "harmonics" [12], which will lead to the unsuccess of key retrieval. However, the impact of harmonics on CPA is minimum. Furthermore, the comparative analysis in [77] indicates that the noise from semiconductor integration process also has less impact to CPA than DPA. Other literature [13] also shows that CPA outperforms DPA in both efficiency and robustness because CPA can better tolerate noise than DPA. As a result, CPA is a better choice when the target of attack is in a complicated noise environment, such as 3D ICs.

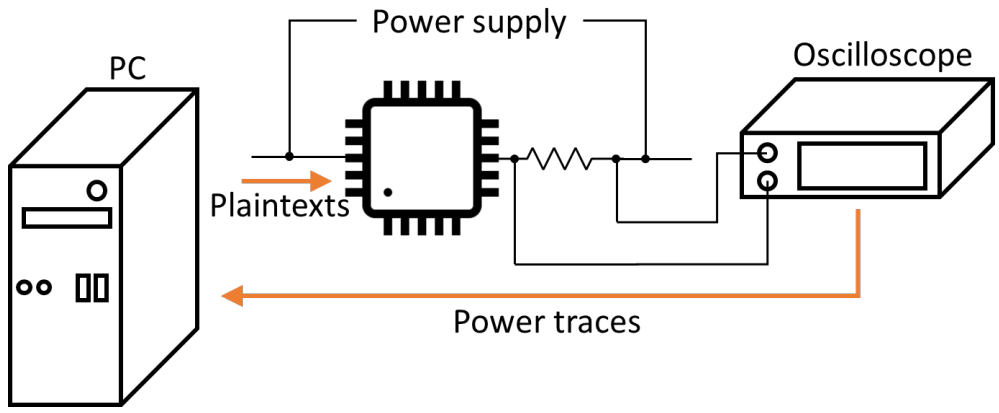


Figure 7.3: Hardware setup of power analysis attacks

Practical Implementation Comparison

The advantages of CPA attack over DPA attack are originated from its statistical metric but not practical implementation. In fact, DPA and CPA attacks have the identical experimental setups, as shown in Fig. 7.3. They both drive the target chip that implements crypto modules with a set of plaintexts. The plaintexts could be generated in and exported from a personal computer (PC). The power measurement is obtained through a resistor (R) that is connected in series with the chip. The resister could be inserted in between the chip and power supply (V_{DD}), or in between the chip and ground [78]. The oscilloscope measures the voltage (V) across the resistor and the transient current (I) can be calculated as V/R. The power consumption (P) of the chip can be obtained as $P = (V/R) \cdot V_{DD}$ (note that R is usually very small so that its power consumption can be ignored). The only difference between the two attacks is the statistical metrics used are different. The metric of PCC outperforms the metric of differential trace in tolerating noise and reducing computing complexity so that CPA attack is usually considered to be more powerful than DPA attack.

DPA and CPA Attacks in Logic Locking

There is limited work discussing the DPA and CPA attack resilience of logic locking techniques. In the existing literature, only DPA attacks are performed on gate-level logic lock-

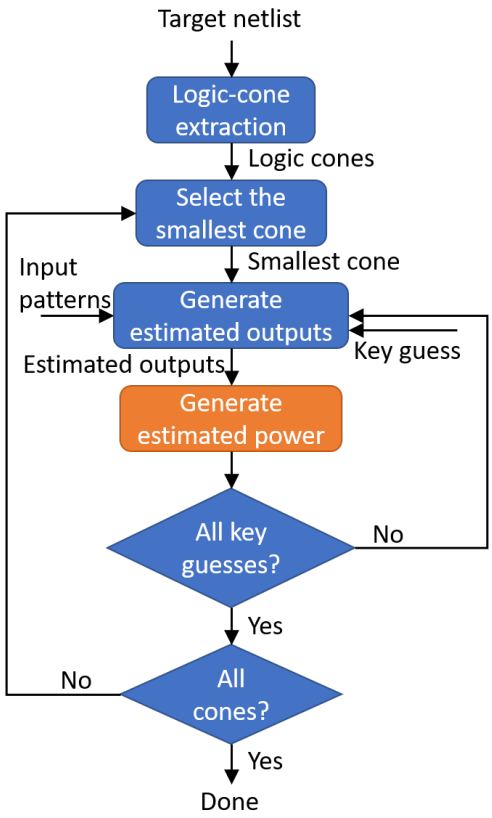


Figure 7.4: The proposed flow for the CPA attack on a general circuit protected by logic locking.

ing [71] but CPA attacks have not been examined in the context of logic locking techniques. To fill this gap, this chapter will propose a CPA attack to logic locking and further comprehensively evaluate the DPA and CPA resilience of the transistor-level logic locking comparing to the conventional gate-level logic locking.

7.3 Proposed Attack Flow for CPA Attacks on Locked Circuits

CPA is more powerful than DPA in key retrieval and it is necessary to evaluate the resilience of transistor-level logic locking against CPA attacks. In this chapter, inspired by the divide-and-conquer strategy of [71], we modify the conventional CPA attack flow for cryptosystems and propose a feasible general power estimation procedure. Our CPA attack on the transistor-level logic locking includes four steps.

Algorithm 2: Proposed logic cone extraction.

```
Data: Locked netlist
Result: Logic function of each logic cone
PrimaryOut[] ← Find primary outputs;
PrimaryIn[] ← Find primary inputs;
Key[] ← Find key inputs;
i = 1;
while i ≤ length(PrimaryOut[]) do
    Target ← PrimaryOut[i];
    while Target ∉ PrimaryIn[] && Target ∉ Key[] do
        SelFunc ← Target;
        Search logic gate G(Input, Target);
        Substitute G(Input, Target) into SelFunc;
        Target ← Input;
    end
    return SelFunc;
    i = i + 1;
end
```

Step 1: logic cone extraction. The flow of estimated power generation is depicted in Fig. 7.4. First, the logic cones of the locked netlist are extracted based on the primary outputs. To facilitate the logic cone extraction, we develop a Python script and Algorithm 2 shows the its pseudo-code. The script returns the logic function of each logic cone, which will be used as the selection function ($SelFunc$) in the CPA attack. Given a locked netlist, Algorithm 2 searches for the logic gate (G) that generates each primary output of the netlist. The inputs of G will be the target of the next search until all the new targets are either the primary inputs or the key inputs of the netlist. The located G during this process will form the final $SelFunc$ for the primary out. This process is repeated for each primary output until all logic cones are completed.

Step 2: divide-and-conquer-based power estimation. The CPA attack starts from the smallest logic cone, which includes the least number of locking key bits. This ascending order is adopted for two main reasons. First, the ratio of the number of keys to the number of primary inputs (#Keys/#Primary Inputs) of a smaller cone is smaller, too. As a result, retrieving the keys in a smaller cone is easier than in a larger one [71]. Second, some keys

may appear in multiple cones and the keys that have been previously retrieved in the smaller cones can be used in the attack of the current cone. In this case, the attack will be more likely to succeed since the number of unresolved keys is reduced. Next, a set of input patterns with a random key guess are fed to the extracted selection function ($SelFunc$ in Step 1) of the cone and the estimated outputs for the key guess are calculated. We utilize a Hamming distance model to generate the estimated power. The same process will be repeated for all key guesses and all logic cones.

Step 3: power trace collection. Similarly, the real power trace collection starts from the smallest logic cone and follows the ascending order. For each cone, the same set of the input patterns used in the power estimation are applied to the chip under attack and the physical power consumption is collected. Since only the inputs of the cone currently under attack will be fed with the input patterns, the switching activities of other cones will be minimized and thus there is limited interference from other cones. In parallel with the power consumption measuring, the output patterns of the same cone under attack are recorded from the chip to verify the retrieved key values.

Step 4: correlation analysis. We calculate the PCC between the estimated power and the real power consumption to retrieve the keys of each logic cone. The key guess which yields the highest PCC is considered as the correct key retrieved by the CPA attack.

7.4 Resilience Assessment of Logic Locking Against Power Analysis Attacks

With the proposed CPA attack, we are able to perform a comprehensive evaluation of the attack resilience of logic locking against power analysis attacks, including both DPA and CPA attacks. In this section, we discuss our evaluation from two perspectives which are the comparison between the DPA and CPA resilience and the comparison between the gate-level and the transistor-level logic locking techniques.

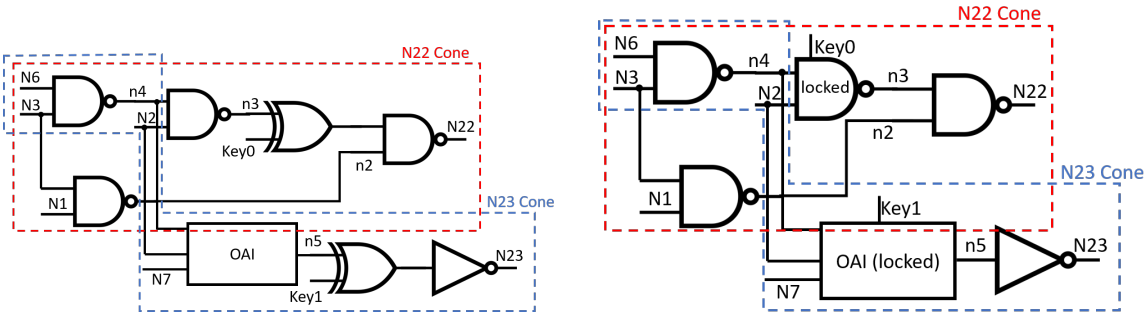


Figure 7.5: c17 protected with (a) XOR-based gate-level and (b) transistor-level logic locking.

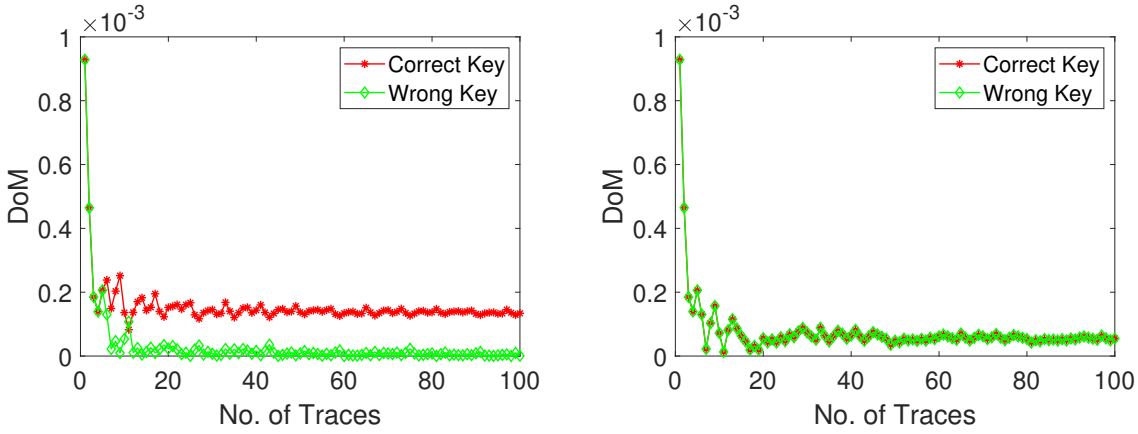
7.4.1 Perspective 1: Comparison of the DPA and CPA Resilience of Logic Locking

Resilience against DPA Attack

We performed the DPA attack on an ISCAS'85 benchmark circuit, c17. As the circuit c17 has two output ports *N22* and *N23*, there are two logic cones highlighted by the two dash-line boxes shown in Fig 7.5. The c17 locked by XOR-based gate locking is shown in Fig. 7.5(a). We also applied PSLNPL and PPLNSL to the NAND and OR-AND-INVERT (OAI) logic gates in c17, as shown in Fig. 7.5(b). The detailed experimental setup is described in Section 7.4.2.

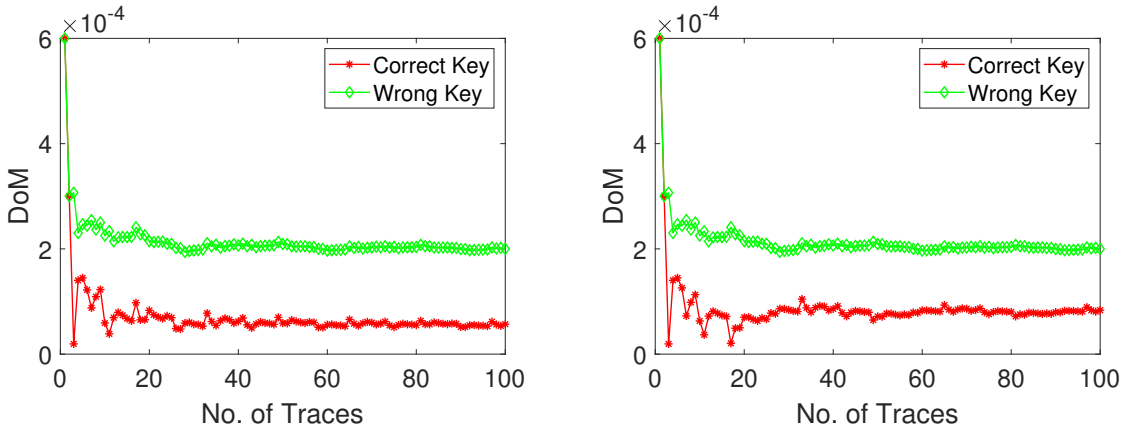
The DoM measured by the DPA attack on c17 protected with three logic locking methods are reported in Figs. 7.6, 7.7 and 7.8. For the N22 cone, Fig. 7.6(a) shows that the DoM line for the correct key is above that for the wrong key, which indicates that the locking key bit applied in the N22 cone can be retrieved by the DPA attack. For the N23 cone, as shown in Fig. 7.6(b), the DoM lines for the correct and wrong keys are overlapped, which means that the DPA attack does not find the correct key. Overall, Fig. 7.6 confirms that gate-level logic locking has 50% resilience against the DPA attack. Based on the measured DoM metrics for PSLNPL and PPLNSL transistor-level locking shown in Figs. 7.7 and 7.8, we conclude that the DPA attack fails to retrieve the locking key bits in 75% of the test cases.

According to the introduction in Section 7.2.3, the wrong locking key at the transistor-level locking will lead to a constant output. This characteristic could form a natural defense



(a) (b)

Figure 7.6: DoM for (a) N22 cone and (b) N23 cone in c17 locked with XOR-based gate-level locking.



(a) (b)

Figure 7.7: DoM for (a) N22 cone and (b) N23 cone in c17 locked with PSLNPL.

line to thwart the DPA attack. When the constant output induced by the wrong key is fed to another logic gate as a controlling bit (e.g., constant 1 to OR gate), the output of the subsequent gate will be constant, too. If more key bits are inserted in the circuit, the probability of propagating the constant output to the primary output is likely to increase. Since the wrong key guess leads the primary output to be a constant 1 (0), all the power traces will be grouped into the power set $T1$ ($T0$). Consequently, the wrong key guess will yield a higher DoM than the correct key, and thus the DPA attack will conclude a wrong key. In summary, once the wrong key causes the primary output of the locked netlist to be

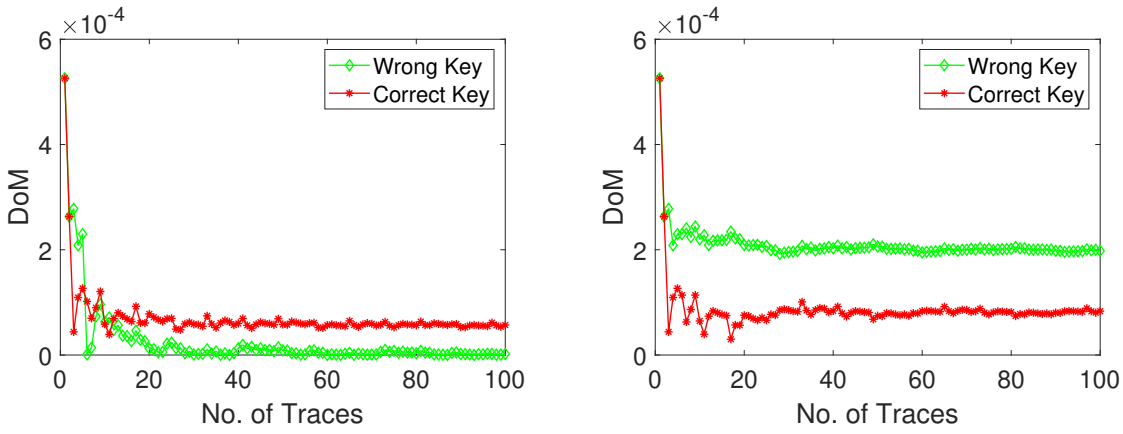


Figure 7.8: DoM for (a) N22 cone and (b) N23 cone in c17 locked with PPLNSL.

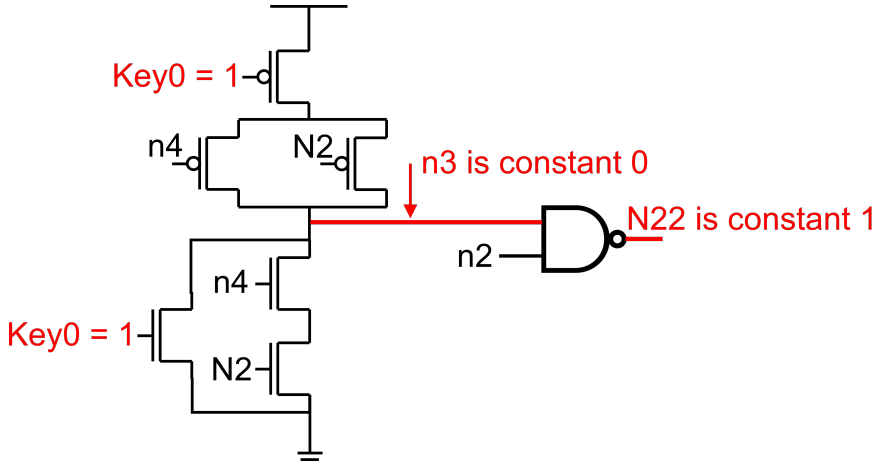


Figure 7.9: The impact of transistor-level locking on the output.

constant, the DoM metric will mislead the DPA key retrieval.

We studied the three DPA failed cases shown in Figs. 7.7(a) and (b) and Fig. 7.8(b) and observed that the primary outputs in those cases are indeed constant regardless of what primary inputs were provided. For example, if the guessed Key0 is wrong in the PSLNPL configuration, the output of the locked NAND in Fig. 7.5(b) will be constant 0, which further causes N22 to be constant 1, as shown in Fig. 7.9. In this case, the locking key obtained by the DPA attack is wrong because all the power traces are grouped to $T1$. Due to the same reason (but different constant outputs of the locked gates), the other two cases represented by Figs. 7.7(b) and 7.8(b) also fail to retrieve the correct locking keys.

Resilience against CPA Attack

The CPA attack was executed on the same c17 circuit for both the XOR-based gate-level locking and the transistor-level locking. The metric PCC was utilized to differentiate the correct key from the wrong ones. As shown in Fig. 7.10, for the N22 cone, the PCC of the correct key case is much higher than the PCC of the wrong key; while for the N23 cone, the correct and incorrect key guesses lead to comparable PCCs after the initial vibration stage. This observation means, in the case of XOR-based locking, the CPA attack can successfully retrieve the key bit in the N22 cone but cannot retrieve the key bit in the N23 cone. In the case of the c17 locked with the transistor-level locking, the PCC for the correct key guess is higher than that for the wrong key guess after 40 power traces. This observation holds true for both N22 and N23 logic cones, as shown in Figs. 7.11 and 7.12. This means that the locking key bits in both cones can be successfully identified by the CPA attack. In the case of the N23 cone locked with the PPLNSL configuration, the estimated power consumption has no correlation with the real power traces. This is because the wrong Key1 shown in Fig. 7.5(b) leads to a constant 0 on the output of the N23 logic cone. Based on the Hamming distance/Hamming weight model, the estimated power consumption is constant 0, too. According to Eq. (7.2), no valid PCC can be calculated. In summary, the CPA attack can retrieve all the locking key bits in the transistor-level locking cases but the DPA attack only partially recovers the key.

The experimental results above indicate that the PCC metric used in CPA attacks is not affected by the constant output caused by the wrong key guess at the transistor-level locking. Instead, the constant output facilities the CPA attack to succeed. This is because the estimated power consumption T_{hyp} will be constant once the estimated output C is constant due to the wrong key guess. Based on the definition of PCC, a constant sequence will have no correlation with the real power consumption T. As a result, the wrong key guess can be easily excluded by the CPA attack.

On the other hand, the CPA attack can be mitigated by the gate-level logic locking,

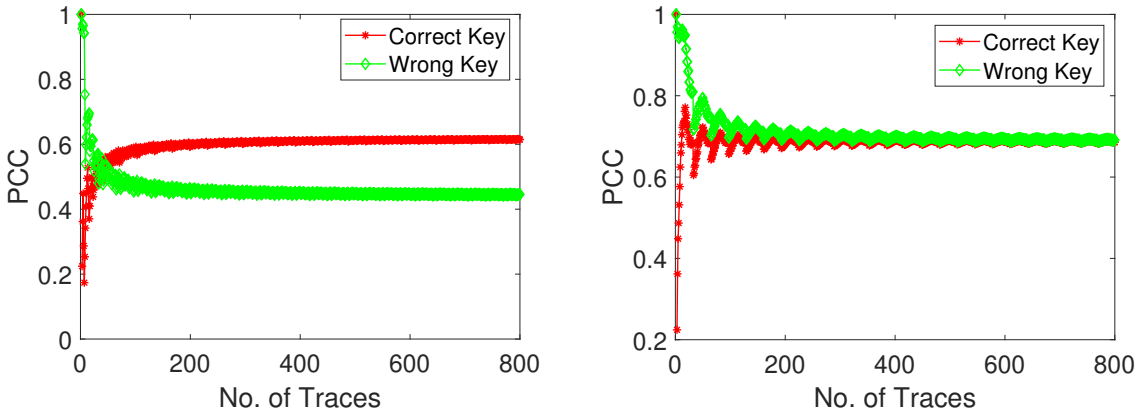


Figure 7.10: PCC for (a) N22 cone and (b) N23 cone in c17 locked with XOR-based gate-level locking.

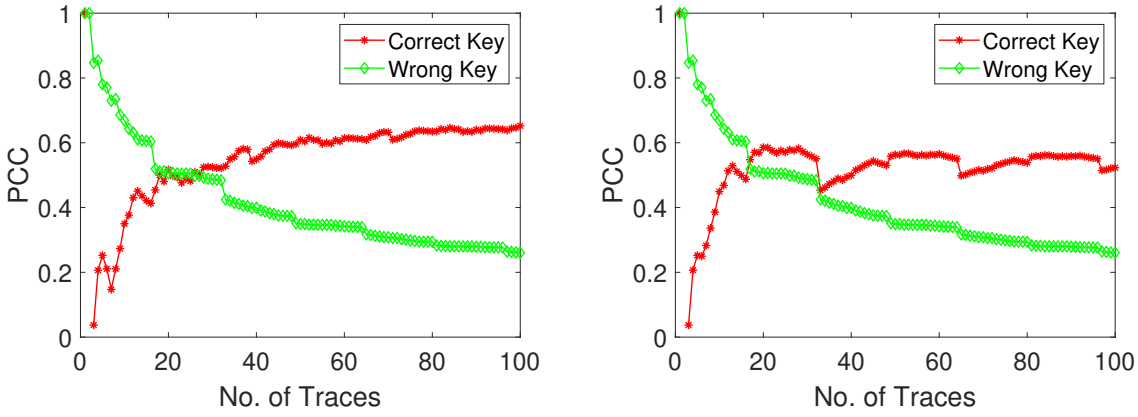


Figure 7.11: PCC for (a) N22 cone and (b) N23 cone in c17 locked with PSLNPL.

as shown in Fig. 7.10(b). The XOR-based gate-level locking will lead the locked gate to produce a flipped output if a wrong key is applied. Once the wrong output is propagated to the primary output of the logic cone, the PCCs for the wrong and correct key cases will be the same, no matter which power model is employed in power estimation. If the Hamming distance model is used, T_{hyp} for the wrong key guess will be identical with the one for the correct key and so is PCC. For example, the original output sequence is [10010] and the flipped sequence is [01101]. Then, the T_{hyp} based on the Hamming distance model is [1011] for both sequences. If the Hamming weight model is adopted, T_{hyp} for the wrong key guess

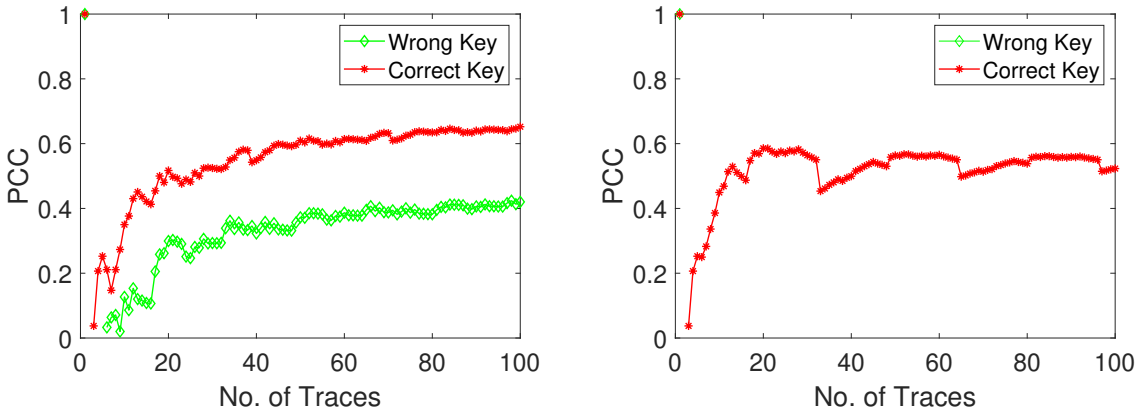


Figure 7.12: PCC for (a) N22 cone and (b) N23 cone in c17 locked with PPLNSL.

will toggle oppositely to the one for the correct key guess. Although this flipped T_{hyp} for the wrong key results in a reversed PCC, the |PCC| is still the same with the one for the correct key guess. Consequently, the CPA attack cannot differentiate the wrong key from the correct key for either case. We zoomed in the failed CPA case in the c17 locked with the XOR-based locking and found that its primary output was indeed flipped when a wrong key was given. However, it is not always possible to propagate the constant output in bigger circuits. The CPA resilience provided by the gate-level locking only happens in rare cases.

In summary, our case study indicates that the proposed CPA attack outperforms the DPA attack in the scenario of transistor-level logic locking. We will zoom in on the CPA resilience of transistor-level logic locking and further compare it with the gate-level logic locking from different perspectives using more benchmarks in the following section.

7.4.2 Perspective 2: Comparison of CPA Resilience between the Transistor-Level and Gate-level Logic Locking Techniques

We performed various experiments to evaluate the CPA resilience of the PSLNPL and PPLNSL based transistor-level locking and the XOR-based gate-level techniques using the following setup. The three logic locking methods were applied to the ISCAS'85 benchmark circuits, including c432, c880, c2670 and c3540. The HOPE simulator [79] was adopted to

execute the FLL strategy [72] for key bit insertion. The CPA attack was performed by FPGA emulations and transistor-level simulations in Cadence Virtuoso. In the FPGA emulation, the locked circuits were mapped to the SAKURA-G FPGA board and the power traces were collected through ChipWhisperer. In the transistor-level simulation, the locked circuits were implemented with the NCSU FreePDK45 technology.

The key recovery rate (KRR) [71] defined in Eq. (7.3) is used to assess the efficiency of CPA attacks on the benchmark circuits protected with XOR-based gate-level logic locking and the transistor-level PSLNPL and PPLNSL logic locking. In this subsection, we examine the impact of locking level, key insertion location, number of key bits, and other factors on the attack resilience.

$$KRR = \frac{No.\,Retrieved\,Key\,Bits}{No.\,Inserted\,Key\,Bits} \tag{7.3}$$

Impact of Locking Level on Attack Resilience

The key insertion locations for the XOR-based gate-level locking were determined by the FLL strategy recommended by the HOPE simulator. The key bits for the transistor-level locking were inserted to the same locations recognized by FLL. Due to the different numbers of logic gates in c432 and c2670, 8 and 16 key bits were used in the encryption, respectively. As shown in Table 7.1, the PPLNSL transistor-level locking achieves better CPA resilience than the XOR-based gate-level locking in the case of c432; however, as the circuit scale increases, the XOR-based locking on c2670 outperforms both PSLNPL and PPLNSL, reducing the KRR by 66%. We further analyzed the guessing entropy to compare the key retrieval speed. As shown in Fig. 7.13, the guessing entropy of the CPA attack on c2670 protected with XOR-based locking is always higher than that for the same circuit encrypted by the transistor-level locking. Both KRR and guessing entropy indicates that PSLNPL and PPLNSL transistor-level locking is more vulnerable to the CPA attack than XOR-based gate-level locking.

Due to the different circuit scale, 7 and 11 key bits were applied to c432 and c880,

Table 7.1: KRR results for CPA attacks on two locked benchmark circuits.

circuit \ locking configuration	XOR	PSLNPL	PPLNSL
c432	100%	100%	50%
c2670	18.75%	56.25%	56.25%

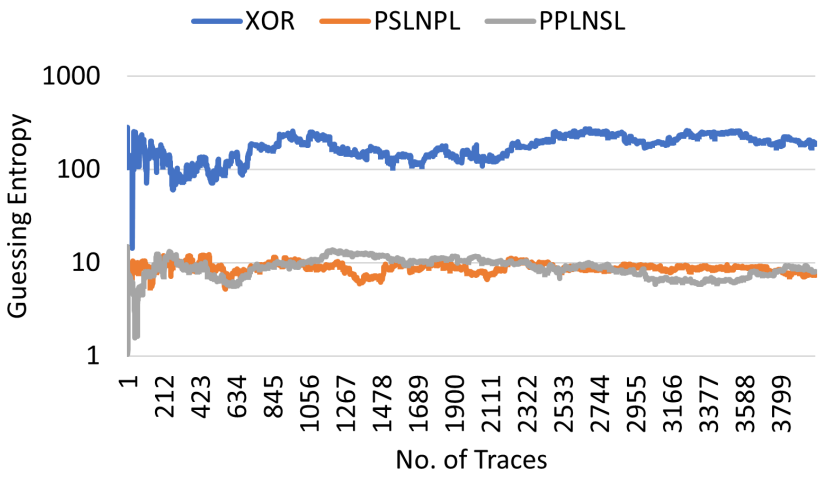
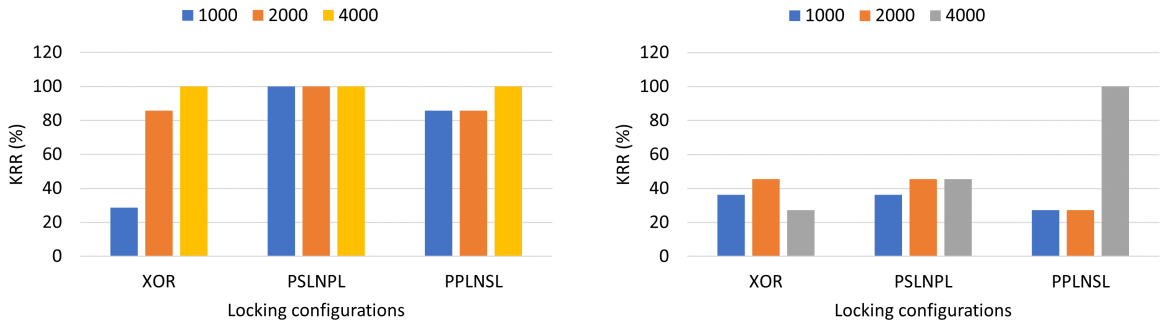


Figure 7.13: Guessing entropy comparison for the case of c2670.



(a) (b)

Figure 7.14: KRR results for (a) c432 (b) c880.

respectively. The KRR of the CPA attack on c432 and c880 is shown in Fig. 7.14. With 4000 power traces, our CPA attack retrieved all the key bits for c432 no matter which locking configuration was used; for the bigger circuit c880, the CPA attack also achieved a 100% KRR in the PPLNSL configuration.

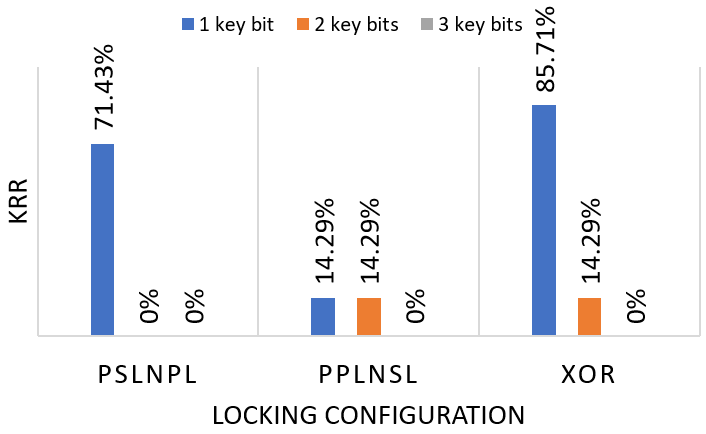


Figure 7.15: Impact of the number of key bits per cone on KRR.

Impact of Number of Key Bits on Attack Resilience

We swept the number of key bits inserted in c432 from 1 key bit per cone to 3 key bits per cone for both the gate-level and transistor-level logic locking techniques. As indicated in Fig. 7.15, given 800 power traces, all three locking methods achieve the KRR of 0% as the number of key bits increases. Our case study indicates that increasing the key space will improve the resilience against the CPA attack. This motivates us to develop a mitigation method to enlarge the key space and the logic cone size interested in the CPA attack.

Impact of Different Key Locations on Attack Resilience

We randomly selected the key locations for the c432 locked with both the XOR-based locking and the transistor-level locking. The KRR results shown in Fig. 7.16 imply that the KRR has strong dependency on the locking location, no matter at gate level or transistor level. This motivates us to propose a key insertion location guideline to find the best key locations for the transistor-level logic locking to achieve the highest CPA resilience.

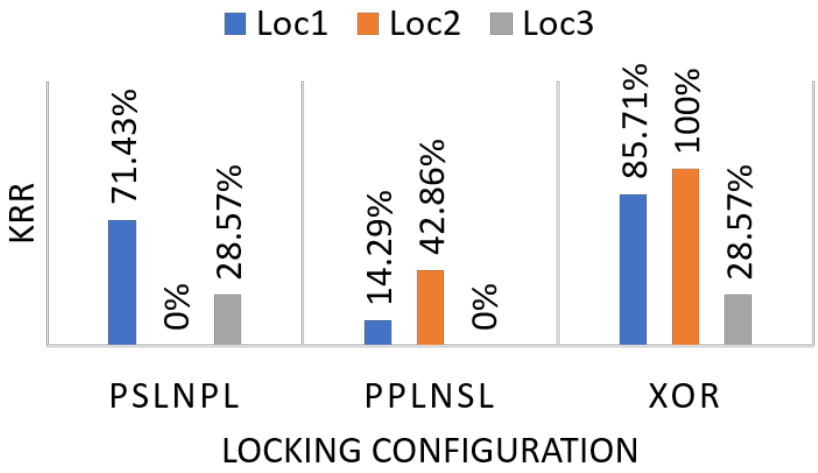


Figure 7.16: The impact of random key insertion locations on KRR.

7.5 Proposed Logic-Cone Conjunction (LCC) Method against CPA Attacks

The CPA attack in Section 7.3 follows the divide-and-conquer strategy to break the locked circuit cone by cone. To thwart the cone-based brute force attack, the work [74] uses MUX-based key gates to connect logic cones so that the key space and the cone size for one single cone is expanded. Inspired by that work, we propose a logic-cone conjunction (LCC) method to mitigate the CPA attack on transistor-level logic locking circuits. The MUX-based attack mitigation method uses multiplexers to create a small overlap area between two logic cones. However, there will always be some keys out of the overlap area. Those keys will not help in expanding the key space unless the two nets connected by the multiplexers are both primary outputs. In contrast, our LCC method embeds one entire logic cone into another one such that the key space can be enlarged significantly. Typically, increasing the number of key bits is a common practice to raise the difficulty of CPA attacks. The proposed LCC method does not induce additional key bit insertion; instead, our method makes full use of the existing locking keys in a locked circuit to maximize the key space of each logic cone. The key space means the number of all possible distinct key combinations.

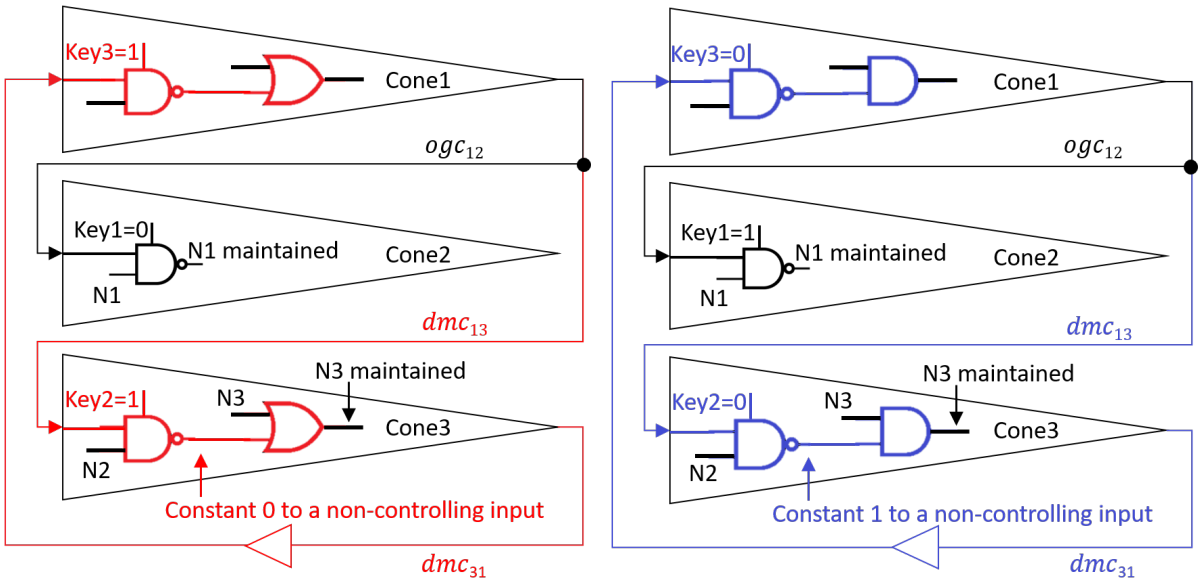
The proposed LCC inserts a key-controlled dummy connection dmc_{ij} between two independent logic cones C_i and C_j to extend the size of each logic cone. To maximize the key

space after our logic-cone conjunction, the selected independent logic cones C_i and C_j should use the exclusive key vectors, $\overrightarrow{K_i}$, and $\overrightarrow{K_j}$, respectively. The LCC method will increase the key space for the logic cone C_i from 2^{K_i} to $2^{K_i+K_j}$. In the best case, $K_i + K_j$ will be equal to the number of all key bits inserted in the locked circuit.

Figures 7.17 (a) and (b) illustrate how the proposed LCC method is applied to the PSLNPL and PPLNSL transistor-level locking circuits. Cone1 and Cone2 are dependent due to the original connection ogc_{12}. In contrast, Cone1 and Cone3 are originally independent since there is no logic overlap between them.

For the configuration of PSLNPL shown in Fig. 7.17(a), designers can insert one key bit, Key1, to the NAND gate in Cone2. The correct key value for Key1 should be logic 0 since Cone1 and Cone2 are originally connected and Cone2 needs signal ogc_{12} to switch normally. Note that inserting Key1 is an important step in LCC. In Cone 3, a dummy NAND locked by the PSLNPL configuration with the key bit Key2 is added. This NAND gate is driven by the output signal dmc_{13} from Cone1 and the net N2, which is any existing net in Cone3. The correct key value for Key2 should be logic 1, which forces the output of the NAND to be constant 0. As the constant 0 will be given to an input of an OR gate, the dummy connection dmc_{13} will not interrupt the original Cone3 operation. Because 0 is the non-controlling bit of an OR gate, its output will be determined by the original net N3. The output of the Cone3 dmc_{31} is brought back to Cone1 to form a cyclic structure between Cone1 and Cone3 using a similar dummy connection. The dummy conjunction between Cone1 and Cone3 will increase the key space for both cones. In this case, the key space for Cone1 (Cone3) is increased from 2^{K_1} (2^{K_3}) to $2^{K_1+K_3}$. Furthermore, no matter which cone is attacked, the cyclic logic structure makes the other cones also switch, thus inducing noise to the power traces collected for CPA attacks. The power noise blurs the correlation between the locking key and the power traces.

The LCC for PPLNSL configuration can be implemented in a similar way. As shown in Fig. 7.17(b), we replace the OR gate with an AND gate and the correct Key2 is logic 0. This



(a) (b)

Figure 7.17: LCC diagram for (a) PSLNPL and (b) PPLNSL configurations.

is because applying a logic 0 to the NAND gate locked with PPLNSL will lead to a constant 1, which is the non-controlling bit for the AND gate. The logic gates used in Fig. 7.17 can be substituted with other gates as long as the normal operations of the revised cones can be maintained when the correct key is provided. To achieve the maximum key space, there could be more than two cones in the conjunction.

The proposed LCC significantly improves the CPA resilience of the transistor-level logic locking for two reasons. First, it significantly enlarges the key space for every single cone to mitigate the cone-based CPA attack. Second, as the LCC method forms the connected cones as a cyclic structure, no matter which cone in the structure is attacked, all other cones will switch. The increased switching activities lead to some power noise, which interferes with the power trace measurement for CPA attacks.

7.6 Proposed Key Insertion Guideline for Transistor-Level Logic Locking to Improve CPA Resilience

In this section, we propose a new strategy that facilitates to search for better key insertion locations for defending CPA attacks.

The work [71] evaluates the DPA resilience of gate-level logic locking techniques. That work also provides two suggestions to harden the locking circuit against DPA attacks: (1) increase the ratio of key bits to the number of primary inputs of the logic cone, and (2) insert key bits in a way that the locked circuit functions closely to the original circuit even when a wrong key is given. However, to the best of our knowledge, there is no prior work available discussing how to enhance the transistor-level logic locking with respect to the CPA attack resilience. To fill this gap, we propose a new guideline (composed of three rules) for the optimal key insertion locations in PSLNPL and PPLNSL based transistor-level locking configuration.

- **Rule 1:** *Avoid inserting a key bit to a gate, whose wrong constant output can be propagated to the primary outputs of the locked circuits.*
- **Rule 2:** *Use the PSLNPL configuration to lock the gates that have logic 0 as their majority output (e.g., AND and NOR gates).*
- **Rule 3:** *Use the PPLNSL configuration to lock the gates that have logic 1 as their majority output (e.g., OR and NAND gates).*

As we observed in Section 7.4.1, the wrong key induced constant primary output will result in an invalid PCC in the CPA algorithm and thus those wrong key guesses can be easily eliminated from the attack process. The proposed rule 1 defers the quick key elimination. In some cases, the primary output may be reversely constant (e.g., logic 1 at the primary output but logic 0 at the gate output), which should be avoided, too.

Output corruptibility is a classic metric evaluating the ability of logic locking techniques in altering the original logic function when wrong keys are given. Usually, a higher output corruptibility will provide a better defense to IP piracy attacks or counterfeiting. However, a lower output corruptibility is more favorable in the sense of thwarting the CPA attack. We denote the difference between the PCC values for a wrong key and a correct key as $DIFF_{PCC}$. As the CPA attack retrieves the correct key by searching for the key yielding the highest PCC, we suggest exploring countermeasures against the CPA attack that can minimize $DIFF_{PCC}$. A method that lowers the output corruptibility helps to achieve a smaller $DIFF_{PCC}$ and obtain a better CPA attack resilience.

Inspired on the relation between the output corruptibility and the CPA resilience, the proposed rules 2 and 3 will enable the transistor-level logic locking to reduce the output corruptibility. We use an NAND gate as an example to explain the utilization of the proposed rules 2 and 3. The majority of the NAND output is logic 1. When a wrong key is given, PSLNPL (PPLNSL) will force the output of the locked gate to be a constant 0(1). In this case, the wrong key of a PPLNSL locked NAND gate may not change the original logic output of the locked circuit as much as that of the PSLNPL configuration. This is because the NAND gate outputs logic 1 for most time (if its input 0/1 is uniformly distributed). It is reasonable to infer that the PCC for the wrong key in PPLNSL is closer to that for the correct key compared to the scenario of PSLNPL. As a result, we conclude that $DIFF_{PCC}(PPLNSL) \lessapprox DIFF_{PCC}(PSLNPL)$. Thus, it is more difficult for CPA to differentiate the correct key from the wrong keys in the PPLNSL configuration than in the case of PSLNPL configuration.

7.7 Experimental Results for The Proposed CPA Resilience Enhancement Methods

7.7.1 Experimental Setup

We performed the experimental verification and evaluation for the proposed LCC and the key insertion guideline through FPGA emulations. Both PSLNPL and PPLNSL configurations, with and without LCC, using and not using the proposed key insertion guideline were applied to the ISCAS benchmark circuits and implemented in a SAKURA-G FPGA board. The power traces were collected using ChipWhisperer software. The CPA algorithm was realized in MATLAB and the Xilinx Vivado design suite. The hardware overhead of LCC was assessed in the Xilinx PlanAhead and XPower Analyzer software.

7.7.2 Experimental Results for LCC

Seven key bits were inserted to c432 for both PSLNPL and PPLNSL configurations following the fault analysis-based logic locking (FLL) [72] to achieve the maximum output corruptibility. c432 has seven logic cones for the primary outputs N223, N329, N370, N421, N430, N431, and N432. N223 is also an input for the logic cone of N329. Furthermore, N329 is fed to the logic cone of N370. Finally, N370 drives the logic cones for N421, N430, N431, and N432. Based on the locked netlist, we found that connecting the output of N421 cone to N223 cone can help to maximize the key space and induce the largest amount of noise to the power traces. In the following experiments, we assume that there is an extra key beside the seven keys to lock the dummy connection logic between N421 and N223 cones and this extra key is known to the CPA attacker for a fair comparison with the baseline.

Improved Resilience against CPA Attack

We collected 4000 power traces for the assessment of KRR. As shown in Fig. 7.18, the CPA attack successfully retrieves all the 7 keys (100% KRR) of the baseline c432 for both locking

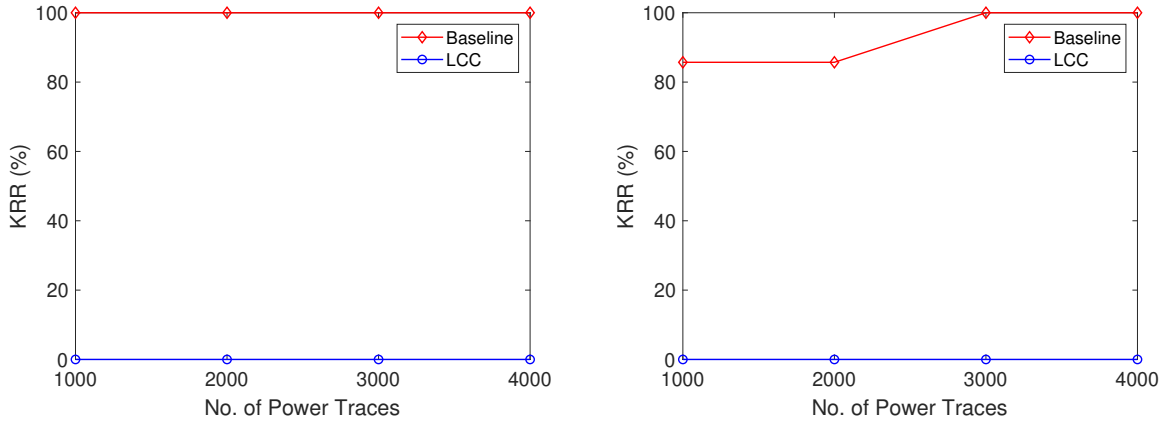


Figure 7.18: KRR comparison for (a) PSLNPL and (b) PPLNSL configurations.

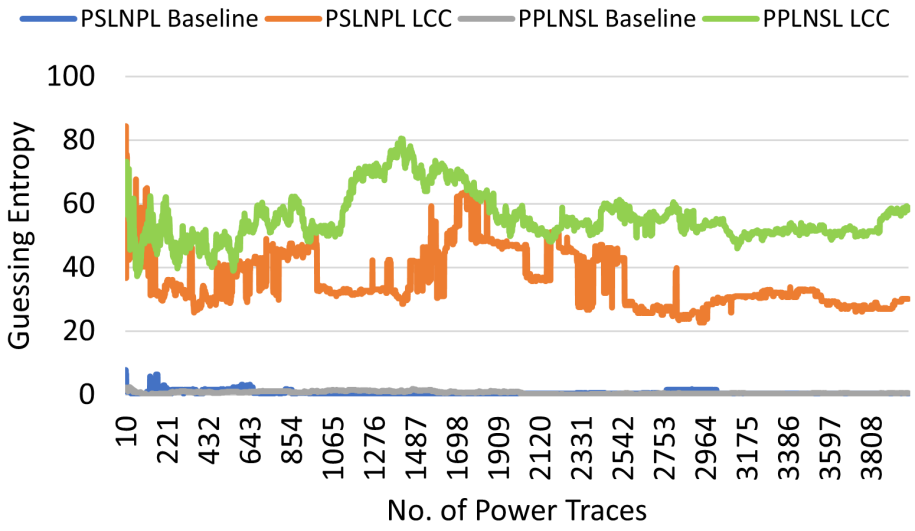


Figure 7.19: Guessing entropy comparison.

configurations. In contrast, the KRR of the c432 protected with LCC decreases from 100% to 0% for both PSLNPL and PPLNSL locking configurations. We also zoom in the guessing entropy for the baseline and the LCC-protected c432. As shown in Fig. 7.19, the guessing entropy of the baseline is close to 0 while the proposed LCC improves the entropy to a much higher level. Both KRR and guessing entropy indicate that the LCC method successfully enhances the locking circuit's resilience against the CPA attack.

The improved attack resilience is originated from the cyclic structure generated by LCC.

Table 7.2: Comparison of Cone Interference (CI).

cones	PSLNPL		PPLNSL	
	Baseline	LCC	Baseline	LCC
N432	81.44%	82.37%	81.86%	81.45%
N431	81.30%	81.75%	81.45%	82.11%
N430	82.13%	82.05%	82.19%	81.13%
N421	90.32%	90.02%	90.62%	90.59%
N370	83.42%	82.53%	82.81%	83.25%
N329	**27.96%**	**86.72%**	**26.84%**	**86.60%**
N223	**0%**	**95%**	**0%**	**94.87%**

Because of the cyclic logic loop, the uninterested cones will have logic switching when the target cone is under attack. Consequently, LCC yields some power noise and thus undermines the CPA attack. We use a metric *Cone Interference*(CI) expressed in Eq. (7.4) to evaluate the noise induced by LCC.

$$CI = \frac{Switching\ Events\ of\ Uninterested\ Cones}{Switching\ Events\ of\ Entire\ Circuit} \tag{7.4}$$

In which *Switching Events of Uninterested Cones* is the total number of bit flips on the primary outputs of the logic cones that the attacker is not interested in. *Switching Events of Entire Circuit* is the total number of bit flips on all the primary outputs of the circuit. Based on the results shown in Table 7.2, LCC significantly improves the cone interference in the attacks to N223 and N329 cones. This is because LCC forces all 7 cones of c432 to switch no matter which one is under attack. Because cones N223 and N329 are the smallest cones that are included in any other cone of c432, interfering with the power traces of these two cones is extremely important to securing the entire circuit. The cone interference for the LCC protected c432 and the baseline are comparably high after N329. Since all the remaining cones are driven by all the primary inputs of c432, all 7 cones of c432 will switch when any of the logic cones N370, N421, N430, N431, and N432 is under attack.

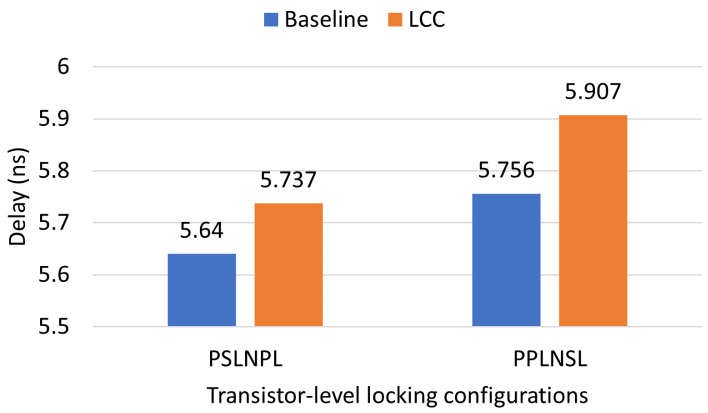


Figure 7.20: Delay overhead.

Overhead on Delay and Power

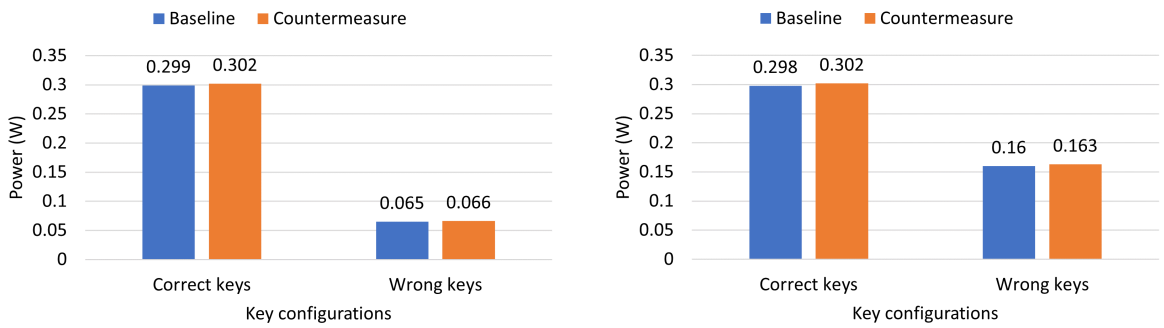
As the proposed LCC makes full use of the existing locking keys to expand the key space without inducing new key insertions, the hardware cost for our method is minor. The critical-path delay was measured via the Xilinx PlanAhead software. As shown in Fig. 7.20, the proposed LCC only increases the delay by 1.72% and 2.62% for the PSLNPL and PPLNSL configurations, respectively.

The power overhead was measured via the Xilinx XPower Analyzer. Based on the results shown in Fig. 7.21, for the PSLNPL based logic locking, LCC consumes 1% and 1.54% more power when the correct keys and the wrong keys are applied, respectively. For the PPLNSL configuration, LCC leads to 1.34% and 1.88% more power consumption for the scenarios that the correct and wrong keys were applied, respectively.

7.7.3 Experimental Results for Key Insertion Guideline

Improved Resilience against CPA Attack

Next, we followed the proposed three rules for optimal key locations to lock c2670. Based on the comparison in Table 7.3, our method reduces the KRR of the transistor-level locking to be the same with the KRR that the XOR-based gate-level locking obtains. As shown in Table 7.4, the selected logic gates for the PSLNPL key insertion have a relatively high



(a) (b)

Figure 7.21: Power overhead for (a) PSLNPL and (b) PPLNSL configurations.

Table 7.3: Impact of FLL and proposed key location selection strategy on KRR.

Key insertion strategy	Locking configuration	KRR
FLL	XOR	18.75%
	PSLNPL	56.25%
	PPLNSL	56.25%
Proposed Strategy	PSLNPL	18.75%
	PPLNSL	18.75%

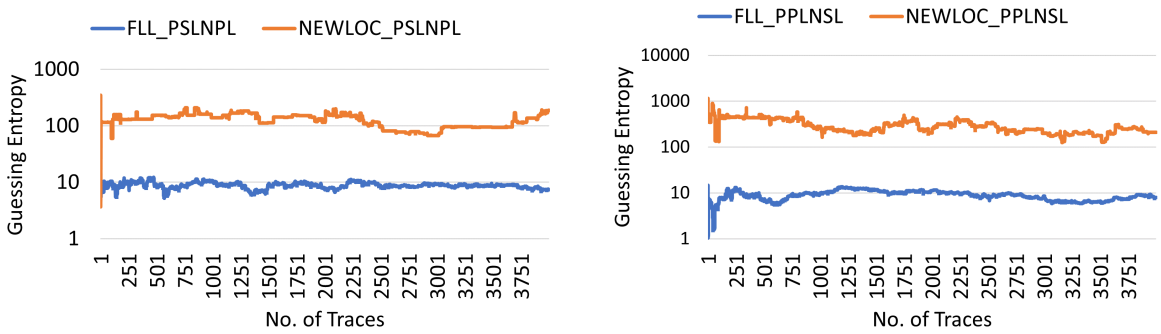
Table 7.4: Key locations following the proposed strategy.

Locking configuration	Key insertion
PSLNPL	7 AND5, 8 AND4, 1 AND3
PPLNSL	2 OR5, 12 OR4, 2 OR3

probability to have logic 0 as outputs. Likewise, the selected key gates for the PPLNSL configuration have a high probability to output logic 1. The FLL strategy for the XOR-based locking is not an option for the transistor-level locking. The constant gate outputs caused by wrong keys could be propagated to the primary outputs, which facilitate the CPA attack.

Furthermore, the guessing entropy for the transistor-level locking configured following the FLL and the proposed key location is compared in Fig. 7.22. NEWLOC_PSLNPL and NEWLOC_PPLNSL represent the PSLNPL and PPLNSL locking configured with our proposed locking locations. As can be seen, our method improves the guessing entropy by 25.36× and 26.04× for PSLNPL and PPLNSL, respectively, based on the 4000 power traces.

The output corruptibility of the XOR-based gate-level locking and transistor-level locking



(a) (b)

Figure 7.22: Guessing entropy comparison for (a) PSLNPL and (b) PPLNSL configurations.

Table 7.5: Comparison of output corruptibility.

Key inserting strategy	Locking configuration	1 bit wrong	5 bits wrong	All bits wrong
FLL	XOR	1.90%	6.99%	10.24%
	PSLNPL	0.98%	4.23%	8.00%
	PPLNSL	0.93%	3.95%	7.89%
Proposed strategy	PSLNPL	0.11%	0.32%	0.62%
	PPLNSL	0.48%	1.58%	3.35%

is compared in Table 7.5. We measured the output corruptibility for the cases of 1 bit, 5 bits and all bits wrong. For the cases of 1 bit wrong, we swept the wrong key bit for all 16 locations and reported the averaged output corruptibility. For the cases of 5 bits wrong, we randomly selected the wrong key bit locations four times and presented the averaged result of the four corresponding output corruptibility. As shown in Table 7.5, if the FLL strategy is applied, the transistor-level locking has the comparable output corruptibility with the XOR-based locking in each test case. However, the proposed new locking location strategy can significantly reduce the output corruptibility, which is consistent with the guessing entropy trend shown in Fig. 7.22.

Key Retrieval Rate and Speed in Large Circuit

We also evaluated the CPA resilience for the locking configurations in a larger benchmark circuit, c3540. We used FLL to configure the XOR-based locking and the proposed strategy

Table 7.6: Cone-based CPA attack effort and its KRR.

	FLL_XOR	NEWLOC_PSLNPL	NEWLOC_PPLNSL
KRR	6.25%	0%	0%
Iteration	1300	1032	6667
Execution time	0.36 hours	0.29 hours	1.85 hours

to configure the PSLNPL and PPLNSL locking. 32 key bits were inserted for each locking. We define 1 day as the time limit so that the logic cone in which the CPA attack takes more than 24 hours is considered as no key bit will be retrieved. The KRR results based on 4000 power traces are shown in Table 7.6. The CPA attack obtains a KRR of 6.25% for the case of XOR-based locking. However, the CPA attack cannot retrieve any key bit in the cases of PSLNPL and PPLNSL locking. Based on the fact that one iteration takes around 1 second in our CPA algorithm ran on a computer at 1.8GHz and with 8GB memory, we estimate the averaged execution time that the CPA attack will take on one logic cone. The comparison of the averaged execution time is listed in Table 7.6. Interestingly, a lower KRR does not necessarily represent a higher attack effort, as the NEWLOC_PSLNPL case actually consumes less iterations and execution time than the FLL_XOR case.

7.8 Conclusion

The nature of M3D tiers being fabricated by one single foundry makes the split manufacturing strategy not applicable for securing M3D ICs from IP piracy attacks. The transistor-level logic locking technique has been proposed to encrypt the original function of the M3D ICs under protection. However, limited works evaluate the strength of transistor-level logic locking on resisting power analysis attacks. This chapter proposes a CPA attack flow that is applicable to the transistor-level logic locking. Our analysis and experimental results indicate that the proposed CPA attack outperforms the DPA attack in transistor-level logic locking and achieves a 100% KRR in the locked c432 with 4000 power traces. Furthermore, we propose two methods to improve the CPA resilience of transistor-level logic locking.

First, a logic-cone conjunction (LCC) method is introduced to enlarge the key space. Our case study on c432 shows that our method can successfully reduce the KRR to zero with negligible overhead on delay and power. Furthermore, we propose three rules as a guideline for the key insertion of transistor-level locking. Our experimental results show that our method improves the guessing entropy by 25.36× and 26.04× for PSLNPL and PPLNSL, respectively, over the FLL based key insertion location, and successfully mitigates the CPA attack.

CHAPTER 8

Conclusion

This dissertation investigates the security of an emerging technology, 3D IC, and provides solutions for the potential security attacks to it. We first reveal a security threat of hardware Trojans to 3D ICs and further provide novel countermeasures to fill the gap in the existing literature of missing effective protection schemes. Our research also protects the confidential information in 3D ICs from side-channel leakage using more effective and less costly methods comparing to existing protection mechanisms. This dissertation makes an important contribution not only to the hardware security community but also to building a reliable and trusted 3D world in the future. To continue this study, more works can be done in combining the existing Trojan detection methods with the proposed 3D IC testing framework and extending the proposed key insertion guideline to be compatible with various specific logic locking techniques.

LIST OF REFERENCES

[1] Zhiming Zhang and Qiaoyan Yu. Towards Energy-Efficient and Secure Computing Systems. *Journal of Low Power Electronics and Applications*, 8(4):48, Dec 2018.

[2] Jaya Dofe and Qiaoyan Yu. Exploiting pdn noise to thwart correlation power analysis attacks in 3d ics. In *Proceedings of the 20th System Level Interconnect Prediction Workshop*, SLIP '18, pages 6:1–6:6, New York, NY, USA, 2018. ACM.

[3] J. Dofe, Chen Yan, S. Kontak, E. Salman, and Q. Yu. Transistor-level camouflaged logic locking method for monolithic 3d ic security. In *2016 IEEE Asian Hardware-Oriented Security and Trust (AsianHOST)*, pages 1–6, 2016.

[4] Johann Knechtel, Ozgur Sinanoglu, Ibrahim Abe M Elfadel, Jens Lienig, and Cliff CN Sze. Large-Scale 3D Chips: Challenges and Solutions for Design Automation, Testing, and Trustworthy Integration. *IPSJ Transactions on System LSI Design Methodology*, 10:45–62, 2017.

[5] Wolfgang Arden, Michel Brillouët, Patrick Cogez, Mart Graef, Bert Huizing, and Reinhard Mahnkopf. More-than-moore white paper. `http://www.itrs2.net/uploads/4/9/7/7/49775221/irc-itrs-mtm-v2_3.pdf`, 2010.

[6] Y. Xie, C. Bao, C. Serafy, T. Lu, A. Srivastava, and M. Tehranipoor. Security and Vulnerability Implications of 3D ICs. *TMSCS*, 2(2):108–122, Apr 2016.

[7] Frank Imeson, Ariq Emtenan, Siddharth Garg, and Mahesh Tripunitara. Securing Computer Hardware Using 3D Integrated Circuit (IC) Technology and Split Manufacturing for Obfuscation. In *Proc. the 22nd USENIX Security Symposium*, pages 495–510, 2013.

[8] Jaya Dofe, Qiaoyan Yu, Hailang Wang, and Emre Salman. Hardware security threats and potential countermeasures in emerging 3D ICs. In *Proc. GLSVLSI'16*, pages 69–74. ACM, 2016.

[9] Z. Zhang and Q. Yu. Modeling Hardware Trojans in 3D ICs. In *Proc. 2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 483–488, July 2019.

[10] Jeyavijayan JV Rajendran, Ozgur Sinanoglu, and Ramesh Karri. Is Split Manufacturing Secure? In *Proc. the Conference on Design, Automation and Test in Europe (DATE)*, pages 1259–1264, Mar 2013.

[11] Y. Wang, P. Chen, J. Hu, G. Li, and J. Rajendran. The Cat and Mouse in Split Manufacturing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 26(5):805–817, May 2018.

[12] Paul Kocher, Joshua Jaffe, Benjamin Jun, and Pankaj Rohatgi. Introduction to differential power analysis. *J. Cryptographic Eng.*, 1(1):5–27, Apr 2011.

[13] Eric Brier, Christophe Clavier, and Francis Olivier. Correlation power analysis with a leakage model. In *Proc. CHES'04*, pages 16–29, 2004.

[14] Jaya Dofe, Hoda Pahlevanzadeh, and Qiaoyan Yu. A Comprehensive FPGA-Based Assessment on Fault-Resistant AES Against Correlation Power Analysis Attack. *J. Electron. Test.*, 32(5):611–624, October 2016.

[15] Peng Gu et al. Leveraging 3D Technologies for Hardware Security: Opportunities and Challenges. In *Proc. GLSVLSI '16*, pages 347–352, 2016.

[16] Y. Xie, C. Bao, C. Serafy, T. Lu, A. Srivastava, and M. Tehranipoor. Security and Vulnerability Implications of 3D ICs. *IEEE Trans.on Multi-Scale Computing Systems*, 2(2):108–122, April 2016.

[17] J. A. Roy, F. Koushanfar, and I. L. Markov. Epic: Ending piracy of integrated circuits. In *Proc. DATE'08*, pages 1069–1074, 2008.

[18] M. Tehranipoor and F. Koushanfar. A survey of hardware trojan taxonomy and detection. *IEEE Design Test of Computers*, 27(1):10–25, Jan 2010.

[19] S. Narasimhan, X. Wang, D. Du, R. S. Chakraborty, and S. Bhunia. TeSR: A robust Temporal Self-Referencing approach for Hardware Trojan detection. In *2011 IEEE International Symposium on Hardware-Oriented Security and Trust*, pages 71–74, June 2011.

[20] S. Bhasin and F. Regazzoni. A survey on hardware trojan detection techniques. In *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2021–2024, 2015.

[21] He Li, Qiang Liu, and Jiliang Zhang. A survey of hardware trojan threat and defense. *Integration*, 55:426–437, 2016.

[22] J. Francq and F. Frick. Introduction to hardware trojan detection methods. In *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 770–775, 2015.

[23] T. Inoue, K. Hasegawa, M. Yanagisawa, and N. Togawa. Designing hardware trojans and their detection based on a svm-based approach. In *Proc. ASICON'17*, pages 811–814, Oct 2017.

[24] G. Zarrinchian and M. S. Zamani. Latch-based structure: A high resolution and self-reference technique for hardware trojan detection. *IEEE Transactions on Computers*, 66(1):100–113, Jan 2017.

[25] H. Salmani, M. Tehranipoor, and J. Plusquellic. A novel technique for improving hardware trojan detection and reducing trojan activation time. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 20(1):112–125, 2012.

[26] M. Banga and M. S. Hsiao. Odette: A non-scan design-for-test methodology for trojan detection in ics. In *2011 IEEE International Symposium on Hardware-Oriented Security and Trust*, pages 18–23, 2011.

[27] Siraj Fulum Mossa, Syed Rafay Hasan, and Omar Elkeelany. Hardware trojans in 3-D ICs due to NBTI effects and countermeasure. *Integration*, 59:64–74, 2017.

[28] S. R. Hasan, S. F. Mossa, O. S. A. Elkeelany, and F. Awwad. Tenacious hardware trojans due to high temperature in middle tiers of 3-d ics. In *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 1–4, Aug 2015.

[29] Jaya Dofe, Peng Gu, Dylan Stow, Qiaoyan Yu, Eren Kursun, and Yuan Xie. Security threats and countermeasures in three-dimensional integrated circuits. In *Proceedings of the on Great Lakes Symposium on VLSI 2017*, pages 321–326. ACM, 2017.

[30] S. Madani and M. Bayoumi. A Security-Aware Pre-partitioning Technique for 3D Integrated Circuits. In *Proc. MTV'17*, pages 57–61, Dec 2017.

[31] S. Alhelaly, J. Dworak, T. Manikas, P. Gui, K. Nepal, and A. L. Crouch. Detecting a trojan die in 3D stacked integrated circuits. In *2017 IEEE North Atlantic Test Workshop (NATW)*, pages 1–6, May 2017.

[32] S. Madani, M. R. Madani, I. K. Dutta, Y. Joshi, and M. Bayoumi. A hardware obfuscation technique for manufacturing a secure 3D IC. In *Proc. MWSCAS'18*, pages 318–323, Aug 2018.

[33] P. Yang and M. Marek-Sadowska. Making split-fabrication more secure. In *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–8, Nov 2016.

[34] K. Salah. Survey on 3d-ics thermal modeling, analysis, and management techniques. In *2017 IEEE 19th Electronics Packaging Technology Conference (EPTC)*, pages 1–4, 2017.

[35] Calvin R King Jr. *Thermal management of three-dimensional integrated circuits using inter-layer liquid cooling*. PhD thesis, Georgia Institute of Technology, 2012.

[36] E. J. Marinissen. Challenges and emerging solutions in testing TSV-based 2 1 over 2D- and 3D-stacked ICs. In *Proc. DATE '12*, pages 1277–1282, March 2012.

[37] Xuan Thuy Ngo, Zakaria Najm, Shivam Bhasin, Debapriya Basu Roy, Jean-Luc Danger, and Sylvain Guilley. Integrated Sensor: A Backdoor for Hardware Trojan Insertions? *in Proc. 2015 Euromicro Conference on Digital System Design*, pages 415–422, 2015.

[38] J. Dofe and Q. Yu. Exploiting PDN Noise to Thwart Correlation Power Analysis Attacks in 3D ICs. In *2018 ACM/IEEE International Workshop on System Level Interconnect Prediction (SLIP)*, pages 1–6, June 2018.

[39] Z. Zhang and Q. Yu. 3D Thermal Triggered Hardware Trojan. *Hardware Demo in IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, May, 2019.

[40] S. M. Satheesh and E. Salman. Power Distribution in TSV-Based 3-D Processor-Memory Stacks. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2(4):692–703, Dec 2012.

[41] V. H. Nguyen, P. Christie, A. Heringa, A. Kumar, and R. Ng. An analysis of the effect of wire resistance on circuit level performance at the 45-nm technology node. In *Proc. IITC'05*, pages 191–193, June, 2005.

[42] S. M. Satheesh and E. Salman. Power Distribution in TSV-Based 3-D Processor-Memory Stacks. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2(4):692–703, Dec 2012.

[43] M. H. Jabbar, D. Houzet, and O. Hammami. 3D multiprocessor with 3D NoC architecture based on Tezzaron technology. In *Proc. of 3DIC '11*, pages 1–5, Jan 2012.

[44] Li Jiang and Qiang Xu. Fault-Tolerant 3D-NoC Architecture and Design: Recent Advances and Challenges. In *Proc. of NOCS '15*, pages 7:1–7:8, 2015.

[45] Jonathan Frey and Qiaoyan Yu. A hardened network-on-chip design using runtime hardware Trojan mitigation methods. *Integration, the VLSI Journal*, 56:15 – 31, 2017.

[46] L. Lin, W. Burleson, and C. Paar. MOLES: Malicious off-chip leakage enabled by side-channels. In *2009 IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*, pages 117–122, Nov 2009.

[47] A. Eghbal, P. M. Yaghini, N. Bagherzadeh, and M. Khayambashi. Analytical fault tolerance assessment and metrics for tsv-based 3d network-on-chip. *IEEE Transactions on Computers*, 64(12):3591–3604, 2015.

[48] Taigon Song, Chang Liu, Yarui Peng, and Sung Kyu Lim. Full-chip signal integrity analysis and optimization of 3-d ics. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(5):1636–1648, 2015.

[49] Ahmad Patooghy, Maral Filvan Torkaman, and Mehdi Elahi. Your hardware is all wired up! attacking network-on-chips via crosstalk channel. In *Proceedings of the 12th International Workshop on Network on Chip Architectures*, pages 1–6, 2019.

[50] Zhiming Zhang, Jaya Dofe, Pruthvy Yellu, and Qiaoyan Yu. Comprehensive analysis on hardware trojans in 3d ics: Characterization and experimental impact assessment. *SN Computer Science*, 1(4):1–13, 2020.

[51] Ieee standard for test access architecture for three-dimensional stacked integrated circuits. *IEEE Std 1838-2019*, pages 1–73, 2020.

[52] Yu Li, Ming Shao, Hailong Jiao, Adam Cron, Sandeep Bhatia, and Erik Jan Marinissen. Ieee std p1838's flexible parallel port and its specification with google's protocol buffers. In *2018 IEEE 23rd European Test Symposium (ETS)*, pages 1–6, 2018.

[53] F. Karabacak, U. Y. Ogras, and S. Ozev. Detection of malicious hardware components in mobile platforms. In *Proc. ISQED'16*, pages 179–184, 2016.

[54] Michael M Goodwin. *Adaptive signal models: Theory, algorithms, and audio applications*, volume 467. Springer Science & Business Media, 2012.

[55] J. He, Y. Zhao, X. Guo, and Y. Jin. Hardware trojan detection through chip-free electromagnetic side-channel statistical analysis. *TVLSI*, 25(10):2939–2948, 2017.

[56] S. M. Satheesh and E. Salman. Power distribution in tsv-based 3-d processor-memory stacks. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2(4):692–703, 2012.

[57] Aamir Zia, Sachhidh Kannan, H. Jonathan Chao, and Garrett S. Rose. 3D NoC for Many-Core Processors. *Microelectronics Journal*, 42:1380 – 1390, Dec 2011.

[58] K. J. Chen, C. Chao, and A. A. Wu. Thermal-Aware 3D Network-On-Chip (3D NoC) Designs: Routing Algorithms and Thermal Managements. *IEEE Circuits and Systems Magazine*, 15(4):45–69, Nov 2015.

[59] Biresh Kumar Joardar, Wonje Choi, Ryan Gary Kim, Janardhan Rao Doppa, Partha Pratim Pande, Diana Marculescu, and Radu Marculescu. 3D NoC-Enabled Heterogeneous Manycore Architectures for Accelerating CNN Training: Performance and Thermal Trade-offs. In *Proc. the International Symposium on Networks-on-Chip (NOCS)*, pages 18:1–18:8, Oct 2017.

[60] A. Prodromou, A. Panteli, C. Nicopoulos, and Y. Sazeides. NoCAlert: An On-Line and Real-Time Fault Detection Mechanism for Network-on-Chip Architectures. In *Proc. the IEEE/ACM International Symposium on Microarchitecture*, pages 60–71, Dec 2012.

[61] J. Dofe and Q. Yu. Novel Dynamic State-Deflection Method for Gate-Level Design Obfuscation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(2):273–285, Feb 2018.

[62] O. X. Standaert, E. Peeters, G. Rouvroy, and J. J. Quisquater. An overview of power analysis attacks against field programmable gate arrays. *Proceedings of the IEEE*, 94(2):383–394, Feb 2006.

[63] Jeremy Cooper and E. Demulder. Test vector leakage assessment (tvla) methodology in practice (extended abstract). 2013.

[64] Debapriya Basu Roy, Shivam Bhasin, Sylvain Guilley, Annelie Heuser, Sikhar Patranabis, and Debdeep Mukhopadhyay. Leak me if you can: Does tvla reveal success rate. Technical report, Cryptology ePrint Archive, Report 2016/1152, 2016.

[65] Tobias Schneider and Amir Moradi. Leakage assessment methodology. In Tim Güneysu and Helena Handschuh, editors, *Cryptographic Hardware and Embedded Systems – CHES 2015*, pages 495–513, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[66] K. Tiri, M. Akmal, and I. Verbauwhede. A dynamic and differential cmos logic with signal independent power consumption to withstand differential power analysis on smart cards. In *Proceedings of the 28th European Solid-State Circuits Conference*, pages 403–406, Sept 2002.

[67] C. Tokunaga and D. Blaauw. Securing Encryption Systems With a Switched Capacitor Current Equalizer. *IEEE Journal of Solid-State Circuits*, 45(1):23–31, Jan 2010.

[68] D. Jayasinghe, A. Ignjatovic, J. A. Ambrose, R. Ragel, and S. Parameswaran. Quadseal: Quadruple algorithmic symmetrizing countermeasure against power based side-channel attacks. In *2015 International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*, pages 21–30, Oct 2015.

[69] D. Zhang, X. Wang, M. T. Rahman, and M. Tehranipoor. An on-chip dynamically obfuscated wrapper for protecting supply chain against ip and ic piracies. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 26(11):2456–2469, 2018.

[70] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri. Security analysis of logic obfuscation. In *DAC Design Automation Conference 2012*, pages 83–89, 2012.

[71] A. Sengupta, B. Mazumdar, M. Yasin, and O. Sinanoglu. Logic locking with provable security against power analysis attacks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 39(4):766–778, 2020.

[72] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri. Fault analysis-based logic encryption. *IEEE Trans Comput*, 64(2):410–424, 2015.

[73] M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri. On improving the security of logic locking. *IEEE TCAD*, 35(9):1411–1424, 2016.

[74] Y. Lee and N. A. Touba. Improving logic obfuscation via logic cone analysis. In *Proc. LATS'15*, pages 1–6, 2015.

[75] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In Michael Wiener, editor, *Advances in Cryptology — CRYPTO' 99*, pages 388–397, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[76] Owen Lo, William Buchanan, and Douglas Carson. Power analysis attacks on the aes-128 s-box using differential power analysis (dpa) and correlation power analysis (cpa). *Journal of Cyber Security Technology*, pages 1–20, 09 2016.

[77] M. Alioto, M. Poli, and S. Rocchi. Power analysis attacks to cryptographic circuits: a comparative analysis of dpa and cpa. In *2008 International Conference on Microelectronics*, pages 333–336, 2008.

[78] Hasindu Gamaarachchi and Harsha Ganegoda. Power analysis based side channel attack. *arXiv preprint arXiv:1801.00932*, 2018.

[79] Hyung Ki Lee and Dong Sam Ha. Hope: an efficient parallel fault simulator for synchronous sequential circuits. *IEEE TCAD*, 15(9):1048–1058, 1996.