

UCLA

UCLA Electronic Theses and Dissertations

Title

Consistency-based System Security Techniques

Permalink

<https://escholarship.org/uc/item/4fk379ms>

Author

Wei, Sheng

Publication Date

2013

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Consistency-based System Security Techniques

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Sheng Wei

2013

© Copyright by
Sheng Wei
2013

ABSTRACT OF THE DISSERTATION

Consistency-based System Security Techniques

by

Sheng Wei

Doctor of Philosophy in Computer Science

University of California, Los Angeles, 2013

Professor Miodrag Potkonjak, Chair

Gate-level characterization (GLC) is the process of characterizing each gate of an integrated circuit (IC) in terms of its properties, such as power and delay. It is a key step in the IC applications regarding cryptography, security, and digital rights management. However, GLC is challenging due to unpredictable process variations, gate correlations, and difficulties to scale to large designs.

We have developed a new approach for hardware and system security using consistency-based GLC and statistical analysis. In particular, we first conduct input vector control, test point insertion, and thermal conditioning to impose extra variations to the IC properties and break the correlations among gates. Then, we partition the circuit into small segments and characterize the gate-level IC properties in each segment. Finally, we employ statistical methods to analyze the consistency of the gate-level properties, both intra- and inter-segments, to identify and diagnose malicious modifications (e.g., hardware Trojans) or other misconduct (e.g., IC counterfeiting) made by an adversary.

Based on our research findings in the consistency-based GLC, we develop a group of hardware security applications, including (1) hardware Trojan detection and diagnosis; (2) hardware metering and digital rights management; and (3)

remote and in-field wireless security. The effectiveness of the consistency-based GLC in varieties of applications indicates that it is the foundation and enabler for reliable hardware and system security techniques.

The dissertation of Sheng Wei is approved.

Hongquan Xu

Majid Sarrafzadeh

Milos Ercegovac

Miodrag Potkonjak, Committee Chair

University of California, Los Angeles

2013

TABLE OF CONTENTS

1	Introduction	1
1.1	Motivation	1
1.2	Challenges	2
1.3	Design Principles	3
1.4	Contributions and Organizations	4
2	Background and Preliminaries	7
2.1	Power Models	7
2.2	Delay Model	8
2.3	Process Variation Model	9
2.4	IC Aging Model	10
3	Gate-level Characterization	11
3.1	GLC Overview	11
3.2	Gate-level Power Characterization	12
3.2.1	Objectives	12
3.2.2	Power Measurements and Equations	12
3.2.3	Technical Issues in Power GLC	14
3.2.4	Power Characterization Overall Flow	15
3.2.5	Correlation Detection	16
3.2.6	Correlation Elimination	18
3.2.7	Improving the Objective Function	21

3.2.8	MLE Post-processing	24
3.3	Gate-level Delay Characterization	25
3.3.1	Delay Characterization Overview	25
3.3.2	Delay Paths Identification and Selection	27
3.3.3	Delay GLC Using Linear Programming	30
3.3.4	SAT-based Approach for Characterizing All Gates	32
3.4	Scalability Techniques for GLC	34
3.4.1	IC Segmentation by Conducting Input Vector Control	35
3.4.2	Region-based Circuit Partition for Delay Characterization	36
3.5	GLC Evaluation Results	40
3.5.1	Power GLC	40
3.5.2	Delay GLC	41
3.6	GLC Related Work	47
4	Hardware Trojan Detection and Diagnosis	49
4.1	Hardware Trojan Attack Model	50
4.1.1	One-gate HT Model and Benchmark	50
4.1.2	Customizable HT	58
4.2	HT Variable-based Hardware Trojan Detection and Diagnosis	62
4.2.1	HT Detection Using HT Variable	62
4.2.2	HT Diagnosis Using HT Variable	64
4.3	Consistency-based Hardware Trojan Detection and Diagnosis	67
4.3.1	Consistency-based Hardware Trojan Detection	67

4.3.2	Consistency-based Hardware Trojan Diagnosis	72
4.3.3	Self-consistency Analysis via Optimal Subsegments Creation	74
4.4	Hardware Trojan Evaluation Results	79
4.4.1	Effectiveness of Hardware Trojan Attack Models	79
4.4.2	HT variable-based HT Detection and Diagnosis Results . .	85
4.4.3	Consistency-based HT Detection and Diagnosis Results . .	87
4.5	Hardware Trojan Related Work	90
4.6	Summary and Discussions of Hardware Trojans	92
4.6.1	Discussions on HT attacks	92
4.6.2	Boundaries of the HT Detection Approach	93
4.6.3	Target of HT Detection	94
5	Hardware Metering and Digital Rights Management	96
5.1	Hardware Metering Using Physical GLC	96
5.1.1	IC Metering Overview	96
5.1.2	IC Metering based on Physical Level Characterization . . .	97
5.1.3	Coincidence Estimation	101
5.2	IC Auditing Using Statistical Analysis	104
5.2.1	IC Auditing Overview	104
5.2.2	IC Auditing Model	105
5.2.3	IC Auditing Post-processing	107
5.2.4	IC Auditing Validation	108
5.3	Hardware Metering Evaluation Results	108

5.3.1	IC Metering	109
5.3.2	Coincidence Estimation	113
5.3.3	IC Auditing	113
5.4	Hardware Metering Related Work	116
5.4.1	Extrinsic IC Metering	117
5.4.2	Intrinsic IC Metering	118
6	Remote In-field Wireless Security Techniques	119
6.1	Wireless Security Challenges	119
6.2	Energy Hardware Trojans in Wireless Systems	120
6.2.1	Energy Hardware Trojan Overview	120
6.2.2	Energy Hardware Trojan Attacks	122
6.2.3	Energy Hardware Trojan Detection	126
6.3	Online Security Attack and Defense	131
6.3.1	Attack and Defense Models	132
6.3.2	Online Detection by In-field Power Measurements	134
6.3.3	Online Replay Attack	134
6.3.4	Trusted Detection Using Physically Unclonable Functions	135
6.4	Wireless Security Experimental Results	137
6.4.1	Effectiveness of Energy Hardware Trojan Attack	137
6.4.2	Effectiveness of Temperature-aware Hardware Trojan De- tection	138
6.4.3	Effectiveness of PUF-based Online In-field Detection	140

7 Concluding Remarks	142
References	144

LIST OF FIGURES

1.1	Major contributions and organizations of the dissertation.	6
3.1	Overall flow of our GLC scheme. We use a three-phase procedure (pre-processing, GLC, and post-processing).	17
3.2	Flow of thermal conditioning for GLC. We increase the temperatures of a subset of gates in the circuit to break the correlations in the system of linear equations.	19
3.3	An example of the likelihood function and the piecewise linear approximation, where $l(e_r) = \log(5 - e_r)$, and two breakpoints u_1 and u_2 are being considered.	23
3.4	Example of uncharacterizable IC components using delay measurements.	26
3.5	Overall flow of gate-level delay characterization.	27
3.6	Example of test points insertion for delay characterization.	29
3.7	Example of segmentation using input vector control.	35
3.8	Example of MFFC-based circuit partition for timing characterization.	40
3.9	Area overhead of the inserted test points.	46
3.10	Delay overhead under process variation.	47
4.1	Overall architecture of the one-gate HT attack model.	51
4.2	Overall flow for HT creation and placement.	52

4.3	Switching activities of all gates on ISCAS benchmark C499, under the application of 5000 pairs of input vectors.	53
4.4	Motivational example of the undetectable hardware Trojan horses: (a) finite state machine of a mod-3 up/down counter, which includes 3 normal states (i.e., S_0 , S_1 , and S_2) and 1 redundant state (i.e., S_3); and (b) demonstration of the HT state transition using device aging.	60
4.5	Example of GLC-based HT detection scheme on benchmark C17. The coefficients (nominal leakage power values) [104] are shown in the lookup table. We add one extra HT variable z to the system of measurement equations as the indicator of HTs. e_i ($i = 1, 2, \dots, 8$) represents the leakage power measurement errors. The solution of z is zero when no HTs are present, and it is a large value (256.4) in the case where HTs exist.	64
4.6	Probability density function of HT variable for 500 runs of HT detection on benchmark C17. For all the 500 runs, the value of the HT variable is 0 in the case where there are no HTs, and it is a large value between 220 and 440 when HTs are present.	65
4.7	Example of the segmentation-based HT detection approach: (a) shows that a circuit with five gates is segmented into two segments, and gate $X5$ is the overlapping gate of the two segments; (b) shows the nominal leakage power values for all the gates in the circuit; and (c) demonstrates the formulation of systems of linear equations and their solutions in three cases regarding whether a HT is present in each segment. The discrepancy in the results of the overlapping gate (i.e., $X5$) is an indicator of whether any HT exists or not. . .	69

4.8	Example of consistency-based HT diagnosis. We demonstrate the gate characterization in three segments with overlapping gates. The consistency in Segment 1 and Segment 3 exposes the possible HTs in Segment 2.	71
4.9	Example of the variable elimination technique using linear transformation.	77
4.10	Simulation results of switching activities of all gates on ISCAS'85 benchmarks.	83
4.11	PDF of the HT variable in HT detection, integrated with all the ISCAS and ITC benchmarks in Table 4.5. In the case when no HTs exist, the HT variable has a small value from 0 to 11.2. When a single HT gate is present, the HT variable ranges from 151 to 1214. There is a large enough gap between the two cases to enable us to draw a decision line at around 70 to distinguish the two cases. . .	85
4.12	Distribution of the inconsistency values in the HT-present and HT-free cases.	89
4.13	Comparison of the numbers of measurements in the consistency-based hardware Trojan detection.	89
4.14	Simulation results for the consistency-based HT diagnosis.	90
5.1	Flow of IC metering.	98
5.2	Probability that a gate has coincidence with other gates in terms of L_{eff} (benchmark C432 with 160 gates; mean value of L_{eff} is 1.2).103	
5.3	Validation of our IC auditing scheme: (a) on known sets of chips; N varies from 1 to 1500; (b) on 500 runs of the analytical simulation; N is fixed to 800.	109

5.4	Accuracy of L_{eff} characterization on a set of ISCAS benchmarks.	110
5.5	Accuracy of V_{th} characterization on a set of ISCAS benchmarks .	111
5.6	IC auditing results: prediction error vs. total number of chips (the number of samples is fixed to 20; the sample sizes are fixed to 20; the total number of chips varies from 1 to 2000; and no post-processing of the prediction results is performed).	114
5.7	IC auditing results: prediction error vs. number of samples (the number of chips is 1600; the number of samples varies from 10 to 50; and the sample sizes are fixed to 20 chips.)	116
5.8	IC auditing results: prediction error vs. sample sizes (the number of chips is 1600; the number of samples is fixed to 20; and the sample sizes vary from 10 to 50.)	117
6.1	Leakage current of an inverter and a NAND gate under different input vectors [104].	123
6.2	Example of energy attack via input vector manipulation. The energy consumption caused by the attack input vector is 2.96 times compared to the optimal input vector.	123
6.3	Example of energy attack using forward body biasing.	124
6.4	Example of hiding the HT triggers: (a) a HT trigger embedded in the reconvergent paths where delay is non-observable; (b) a rarely switching HT trigger driven by multiple inputs; and (c) a HT trigger activated by a 5-state finite state machine, which reduces the activation probability exponentially.	125
6.5	Principal component analysis (PCA) model to define the hardware Trojan indicator.	130

6.6	Online attack and defense model.	132
6.7	PUF architecture [61].	136
6.8	PUF-based trusted HT detection.	137
6.9	Leakage power increase due to forward ABB attack.	138
6.10	Accuracy of temperature characterization.	139
6.11	Energy HT detection results: HT indicator values in HT-free and HT-present cases.	140
6.12	The probability of output bit O_i being 1 in the PUF ($w=32, h=3$) following the architecture in Figure 6.7.	141

LIST OF TABLES

3.1	Accuracy of gate-level power characterization.	42
3.2	Results of delay characterization with and without test points. The number of characterizable gates without test points is no more than 60% of all the gates. However, with inserted test points, we can characterize 100% of the gates accurately in all the tested benchmark circuits.	44
3.3	Overhead of test point insertion in terms of the number of test points (i.e., area overhead) and the number of measured paths (i.e., cost of test)	46
4.1	Leakage energy reduction via aging for HT benchmark creation (Benchmark C6288).	80
4.2	Leakage energy reduction via aging for HT benchmark creation (Benchmark C7552).	81
4.3	Switching probability of the HT gate using the maximum independent set approach.	83
4.4	Simulation results regarding uncharacterizable gates due to reconvergences. The high percentage of uncharacterizable gates in each design indicates that there is a large number of candidate locations for embedding the non-detectable one-gate HT trigger.	84
4.5	HT Detection and Diagnosis on ISCAS and ITC Benchmarks . . .	86

4.6	HT detection results using consistency and self consistency-based GLC: the values in the “HT-Free” and “HT-Present” columns represent the average discrepancy of the overlapping gates in terms of their scaling factors.	88
5.1	Accuracy of coincidence estimation. “FP” and “FN” stand for “False Positives” and “False Negatives”, respectively.	112
5.2	IC auditing on large numbers of ICs.	115

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor and doctoral committee chair Prof. Miodrag Potkonjak for his valuable guidance and continuous help ever since I joined the UCLA computer science PhD program in Fall 2008. He not only guided me through the world of academic research and taught me how to become a good researcher, but also offered me great insights and valuable advices that will benefit my future career and life.

I am very grateful to my doctoral committee members, Prof. Milos Ercegovac, Prof. Majid Sarrafzadeh, and Prof. Hongquan Xu, for their valuable comments on my prospectus, dissertation and the final defense. I benefited a lot from their insights while improving this work.

Many thanks to the professors in the UCLA Computer Science Department: Prof. Jason Cong, Prof. Songwu Lu, and Prof. Y.C. Tay (visiting professor), who taught me not only solid knowledge in computer science, but also how to solve challenging research problems and accomplish successful research projects.

Also, many thanks to the researchers at Adobe Research: Vishy Swaminathan, Saayan Mitra, and Tom Jacobs, who offered me great opportunities to gain valuable industrial research experiences through both the internship and the university collaboration program at Adobe.

Furthermore, I would like to thank my colleagues and collaborators: Saro Meguerdichian, Jason Zheng, James Wendt, Jong Hoon Ahnn, Nathaniel Conos, Vishwa Goudar, Teng Xu, Eun-Sook Sung, Prof. Farinaz Koushanfar, Prof. Ani Nahapetian, and Kai Li, who worked together with me in several research projects and offered me great help in research, experiments, and paper writing. Some chapters of this dissertation are based on my publications co-authored with

them and my advisor Prof. Miodrag Potkonjak. For example, a part of Chapter 3 is based on Publications P1-P2, P5, P9, P11, P13, and P16; a part of Chapter 4 is based on Publications P1-P2, P5-P6, P11-P13, and P16-P17; a part of Chapter 5 is based on Publications P3 and P7-P8; and a part of Chapter 6 is based on Publications P4, P10, and P15. In particular, Prof. Farinaz Koushanfar and Kai Li provided experimental platforms, data, and paper writing assistance to support a part of the work in Chapters 3.3, 3.4, 4.1, and 5; Prof. Ani Nahapetian provided experiments and paper writing assistance to support a part of the work in Chapter 5; Saro Meguerdichian provided experiments and paper writing assistance to support a part of the work in Chapters 3.2, 4.2, and 6.4; and my advisor Prof. Miodrag Potkonjak oversaw all my research projects and provided me with great help in all aspects of this dissertation.

Finally, special thanks to my wife, my son, my parents, and all my other family members who gave me great support, patience, and love during my PhD study. Without their support, I could not have finished this work.

VITA

- 2000–2004 B.E. (Computer Science)
Yanshan University, China
- 2004–2008 Graduate Study / Research Assistant (Computer Science)
Beihang University, China
- 2008–2013 M.S. (Computer Science)
University of California, Los Angeles, USA

PUBLICATIONS

- P1. S. Wei, S. Meguerdichian, M. Potkonjak, Gate-level Characterization: Foundations and Hardware Security Applications, Design Automation Conference (DAC), pp. 222-227, 2010.
- P2. S. Wei, M. Potkonjak, Scalable Segmentation-based Malicious Circuitry Detection and Diagnosis, International Conference on Computer-Aided Design (ICCAD), pp. 483-486, 2010.
- P3. S. Wei, M. Potkonjak, Integrated Circuit Security Techniques Using Variable Supply Voltage, Design Automation Conference (DAC), 248-253, 2011.
- P4. M. Potkonjak, S. Meguerdichian, A. Nahapetian, S. Wei, Differential Pub-

lic Physically Unclonable Functions: Architecture and Applications, Design Automation Conference (DAC), pp. 242-247, 2011.

P5. S. Wei, S. Meguerdichian, M. Potkonjak, Malicious Circuitry Detection Using Thermal Conditioning, IEEE Transactions on Information Forensics and Security, Vol. 6, No. 3, pp. 1136-1145, 2011.

P6. S. Wei, M. Potkonjak, Scalable Consistency-based Hardware Trojan Detection and Diagnosis, International Conference on Network and System Security (NSS), pp. 176-183, 2011.

P7. S. Wei, F. Koushanfar, M. Potkonjak, Integrated Circuit Digital Rights Management Techniques Using Physical Level Characterization, ACM workshop on Digital rights management (DRM), pp. 3-14, 2011.

P8. S. Wei, A. Nahapetian, M. Potkonjak, Robust Passive Hardware Metering, International Conference on Computer-Aided Design (ICCAD), pp. 802-809, 2011.

P9. S. Wei, A. Nahapetian, M. Nelson, F. Koushanfar, M. Potkonjak, Gate Characterization Using Singular Value Decomposition: Foundations and Applications, IEEE Transactions on Information Forensics and Security, Vol. 7, No. 2, pp. 765-773, 2012.

P10. S. Wei, M. Potkonjak, Wireless Security Techniques for Coordinated Manufacturing and On-line Hardware Trojan Detection, ACM conference on Security

and Privacy in Wireless and Mobile Networks (WiSec), pp. 161-172, 2012.

P11. S. Wei, M. Potkonjak, Scalable Hardware Trojan Diagnosis, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 20, No. 6, pp. 1049-1057, 2012.

P12. S. Wei, K. Li, F. Koushanfar, M. Potkonjak, Hardware Trojan Horse Benchmark via Optimal Creation and Placement of Malicious Circuitry, Design Automation Conference (DAC), pp. 90-95, 2012.

P13. S. Wei, K. Li, F. Koushanfar, M. Potkonjak, Provably Complete Hardware Trojan Detection Using Test Point Insertion, International Conference on Computer-Aided Design (ICCAD), pp. 569-576, 2012.

P14. S. Wei, J. Zheng, M. Potkonjak, Low Power FPGA Design Using Post-Silicon Device Aging (Abstract Only), International Symposium on Field Programmable Gate Arrays (FPGA), pp. 277, 2013.

P15. S. Wei, J. Ahnn, M. Potkonjak, Energy Attacks and Defense Techniques for Wireless Systems, ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec), pp. 185-194, 2013.

P16. S. Wei, M. Potkonjak, Malicious Circuitry Detection Using Fast Timing Characterization via Test Points, IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 113-118, 2013.

P17. S. Wei, M. Potkonjak, The Undetectable and Unprovable Hardware Trojan Horse, Design Automation Conference (DAC), Article No. 144, 2013.

CHAPTER 1

Introduction

1.1 Motivation

Integrated Circuits (ICs) are fundamental building blocks of virtually all electronic equipments that are widely used today. The cost, performance, and reliability of ICs have become essential design objectives in the modern IC industry. In particular, there are two trends that dominate and play important roles in the IC design and manufacturing.

Firstly, from the perspective of semiconductor technologies, as the scaling of high-performance ICs moves to deep-submicron feature sizes, a higher degree of semiconductor integration provides ever increasing performance. However, the performance gain comes with new challenges such as increased leakage energy, increased substrate noise, profound and intrinsic process variation (PV), and increased susceptibility to environmental (e.g., thermal) and operational (e.g., supply voltage) variations. Among them, PV has emerged as the most limiting factor that essentially redefines the IC synthesis and analysis flow. For example, it has been reported that PV results in up to 20X variations in power consumption and around 30% in timing [14]. Consequently, PV could not only completely compromise the existing IC optimization efforts, it also makes the ICs more vulnerable to even ultra-small variations, due to unintentional defects, unpredictable external factors (e.g., environments), or malicious attacks (e.g., hardware Trojans). As

PV has transitioned the domain of IC design and analysis from deterministic to probabilistic, it becomes rather important to bring it back to the deterministic domain, in order to accurately analyze the ICs for design optimization, quality control, and security.

Secondly, from the perspective of business models, IC outsourcing has been widely adopted by most of the IC design companies in order to reduce the manufacturing cost and thus increase their revenue. However, IC outsourcing induces potential security concerns to the manufactured IC, due to the fact that an untrusted foundry has complete access to the hardware during the manufacturing process. In particular, the security concerns include but not limited to the following: (1) The untrusted foundry may embed hardware Trojans (HTs) in the circuit, which are unwanted and malicious components that would make the IC malfunction or extract confidential information from the application system; and (2) IC counterfeiting, in which an untrusted foundry is capable of manufacturing additional unauthorized copies of ICs to obtain extra profits.

In summary, in the modern and pending IC industry, it is essential to be able to accurately characterize and analyze the manufactured IC in a deterministic manner, from which both the design optimization mechanism and security primitives can benefit. In this dissertation, we focus on the security aspect of the problem and develop a set of consistency-based techniques to characterize and analyze the target ICs.

1.2 Challenges

The task of characterizing an IC in an accurate and deterministic manner is challenging due to the following reasons. First, process variation causes the key IC

properties to vary from their nominal design specifications in a random and unpredictable manner. It is difficult for the characterization methods to determine the IC properties after manufacturing based on the specification. Second, even if one can measure the IC properties precisely, the characterized properties, such as delay and power, are subject to change due to environmental factors (e.g., temperatures) or the nature of the silicon (e.g., device aging). The potential variations do not only increase the complexity of the characterization approach by requiring it to execute repeatedly, they also complicate the corresponding procedure of analysis by mixing various sources of variations. Third, with the rapid growth of transistor scaling, modern ICs often contain huge numbers of transistors, in the magnitude of millions or even more. The scale of the circuit size, in terms of the transistor count, challenges the scalability of the characterization and analysis techniques. Finally, from the perspective of attackers, they tend to minimize the exposure of the attacks while maximizing the damage, which adds another layer of difficulty for the characterization approach in addition to the naturally existed variations.

1.3 Design Principles

In order to address the challenges, we develop a set of consistency-based IC characterization and analysis techniques. Our intuition is that the unexpected component would pose additional variations in the observable IC properties. Even though the variations can be hidden under process variation and become indistinguishable, we observe an important phenomenon that the behavior of the unexpected components would exhibit inconsistent pattern compared to the normal IC components, due to the fact that the controllability over the unexpected components is different from that over the normal gates. In particular, if we par-

tition the IC into several small segments, an unexpected component in any of the segments would cause the shared component of the segments to have inconsistent properties, making it possible to be detected via consistency analysis.

Following this important observation, our design principle is to characterize the gate-level IC properties and conduct consistency-based statistical analysis to identify abnormal IC behavior and patterns. In order to conduct consistency analysis, our initial step is to partition the large IC into a number of small and overlapping segments, where we characterize the gate-level IC properties using global power or delay measurements. Then, we find representative components that are shared by various groups of the segments and analyze their properties in terms of consistency. An inconsistent pattern across segments indicates that there exist either unexpected or malicious components in the target IC.

1.4 Contributions and Organizations

Figure 1.1 summarizes the major contributions of the dissertation. We develop a consistency analysis approach that is based on the accurate characterization of gate-level properties. Based on the characterization results, we demonstrate three system security applications, including hardware Trojan detection, hardware metering and digital rights management, and remote in-field wireless security techniques. In particular, we have the following detailed contributions in each category:

- A systematic consistency analysis approach based on efficient and accurate gate-level characterization using non-intrusive delay and power measurements (Chapter 3):
 - Characterization of gate-level power properties using thermal condi-

- tioning and linear programming (Chapter 3.2); and
- Characterization of gate-level delay properties using test point insertion (Chapter 3.3);
- A set of circuit partition and cost reduction techniques to ensure that the proposed GLC and consistency analysis approaches are scalable to large industrial designs (Chapters 3.4):
 - A circuit segmentation approach by selectively freezing a subset of the inputs and varying the others (Chapter 3.4.1);
 - A region-based circuit partition method to further reduce the measurement cost in delay characterization (Chapter 3.4.2).
- The research and demonstration of a group of system security applications that are based on the GLC and the consistency analysis (Chapters 4–6):
 - A complete set of hardware Trojan attack, detection, and defense models and methods using consistency-based analysis (Chapter 4);
 - An IC metering and auditing approach using physical GLC and statistical analysis (Chapter 5); and
 - A set of wireless security techniques that detect and diagnose hardware Trojans in remote in-field wireless systems (Chapter 6).

Consistency-based System Security Techniques

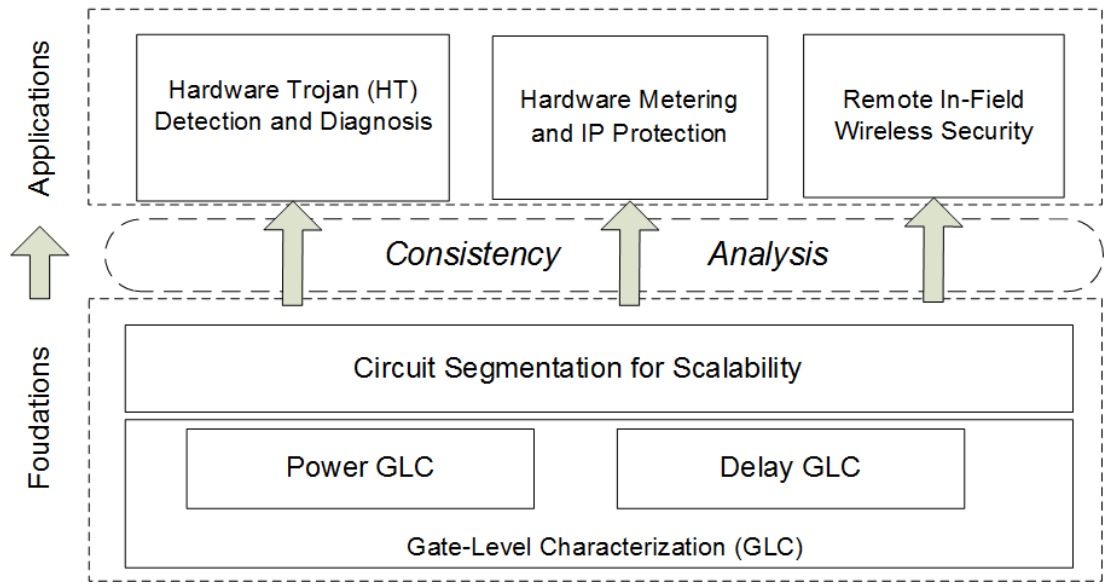


Figure 1.1: Major contributions and organizations of the dissertation.

CHAPTER 2

Background and Preliminaries

In this chapter, we introduce the system models that we employ in the discussion of the dissertation, including the power/energy models, delay model, process variation model, and IC aging model. We regard these models as both the principles that lead to our design methodologies and the evaluation criteria that justify our research findings.

2.1 Power Models

There are typically two possible sources of power dissipation on an IC. One is from gate switching (also termed switching power or dynamic power), where the ICs dissipate power by charging the load capacitances of wires and gates. The other source is static power (also termed leakage power), where the gates dissipate power due to the leakage current even if they do not switch.

Equation (2.1) is the gate-level leakage power model [59], where W is width of the transistor in the driving stage, L is the channel length, V_{th} is the threshold voltage, V_{dd} is the supply voltage, n is the subthreshold slope, μ is the mobility, C_{ox} is the oxide capacitance, k is the Boltzmann constant, T is the absolute temperature, q is the magnitude of charge on an electron, and σ is the drain

induced barrier lowering (DIBL) factor.

$$P_{leakage} = 2 \cdot n \cdot \mu \cdot C_{ox} \cdot \frac{W}{L} \cdot \left(\frac{kT}{q}\right)^2 \cdot V_{dd} \cdot e^{\frac{\sigma \cdot V_{dd} - V_{th}}{n \cdot (kT/q)}} \quad (2.1)$$

Also, we note that the leakage power has a non-linear (exponential) relation with the temperature T , which provides us a way to vary the leakage power by changing the temperature of the circuit. In particular, if we apply a set of primary input vectors to the circuit that switch a set of gates, the gates can be heated up, and the heat will be transferred to other gates on the circuit, which causes the temperatures on the circuit to vary over time. In this way, we can condition the temperatures on the circuit and utilize the exponential relation between temperature and leakage power.

The gate-level switching power model [59] is described by Equation (2.2), where the switching power is dependent on the switching probability per unit of time α , the load capacitance C_L , the transistor width W , the channel length L , and the supply voltage V_{dd} :

$$P_{switching} = \alpha \cdot C_L \cdot W \cdot L \cdot V_{dd}^2 \quad (2.2)$$

2.2 Delay Model

The delay of a single logic gate can be expressed as

$$d = gh + p \quad (2.3)$$

where g and h are logical effort and electrical effort, respectively; and p is parasitic delay. In particular, We use the delay model in [59] that relates the gate delay

to its sizing and operating voltages:

$$Delay = \frac{k_{tp} \cdot k_{fit} \cdot L^2}{2 \cdot n \cdot \mu \cdot \phi_t^2} \cdot \frac{V_{dd}}{(\ln(e^{\frac{(1+\sigma)V_{dd}-V_{th}}{2 \cdot n \cdot \phi_t}} + 1))^2} \cdot \frac{\gamma_i \cdot W_i + W_{i+1}}{W_i}, \quad (2.4)$$

where subscripts i and $i + 1$ represent the the driver and load gates, respectively; γ is the ratio of gate parasitic to input capacitance; and k_{tp} and k_{fit} are fitting parameters.

2.3 Process Variation Model

Process variation (PV) during IC manufacturing causes IC key parameters to vary from their nominal design specifications. For example, PV may vary leakage power by up to 20X and frequency by 30% on a single wafer [14]. In particular, there are two physical level properties that are major sources of PV: threshold voltage (V_{th}) and effective channel length (L_{eff}). For example, the effective channel length of a manufactured gate can be expressed by Equation (2.5), where L_{nom} is the nominal design value of the effective channel length, and ΔL is the variation caused by PV.

$$L_{eff} = L_{nom} + \Delta L \quad (2.5)$$

Several models have been proposed to capture the impact of PV [7][23][24], which formulate ΔL as a random distribution or a combination of multiple distributions to represent the spatial correlations on a chip, as well as the inter-chip variations. In the discussion of this dissertation, we follow the quad-tree model presented by Cline et al. [24] for the variation of L_{eff} (i.e., ΔL). In particular, ΔL is distributed into multiple levels where there are different numbers of grids allocated at each level. The grids at each level are assigned variation values that

follow a Gaussian distribution. Then, we can calculate the total ΔL as the sum of variation values in each level of the grids to which the corresponding gate belongs. Equation (2.6) shows the total variation in the effective channel length of gate j , where ΔL_{ij} is the variation in the i th level grid to which gate j belongs, and μ_i and σ_i are the mean and variance of the Gaussian distribution at level i , respectively.

$$\Delta L_j = \sum_i \Delta L_{ij}, \quad \text{where } \Delta L_{ij} \sim N(\mu_i, \sigma_i) \quad (2.6)$$

For V_{th} , we use the model presented by Asenov et al. [7], where the distribution of V_{th} is obtained by the simulation study of random dopants. V_{th} in this model is fit into a Gaussian distribution, where the parameters are determined by the dopant number and the dopant position.

2.4 IC Aging Model

IC aging causes the threshold voltage of the transistors to increase and, consequently, the speed of the circuit to decrease. In particular, the threshold voltage shift caused by the negative bias temperature instability (NBTI) effect is a function of stress time, temperature, and applied gate voltage, as shown in the following equation [20]:

$$\Delta V_{th} = A \cdot \exp(\beta V_G) \cdot \exp(-E_\alpha/kT) \cdot t^{0.25} \quad (2.7)$$

where t is the stress time; T is the temperature; V_G is the applied gate voltage; A , β and k are constants; and E_α is the measured activation energy of the NBTI process. We employ this aging model to quantify the threshold voltage increase of the gates that are in the stress mode.

CHAPTER 3

Gate-level Characterization

In this chapter, we discuss in details our approach of characterizing each gate of an IC in terms of its manifestational properties, such as delay and power. We show that the proposed characterization approach addresses the concerns caused by process variation and serves as the foundation of conducting consistency-based analysis for system security applications.

3.1 GLC Overview

With the ever increasing trend of transistor scaling, process variation (PV) has emerged as the most limiting factor that essentially redefines IC synthesis and analysis flow. PV is the deviation of IC key parameters from nominal specifications. For example, it has been reported that the frequency of a chip can vary by up to 30% from its nominal design values [14]. For leakage current, the variations are much higher and may reach up to 20 times [14].

Gate-level characterization (GLC) is the process of characterizing each gate of an IC in terms of its properties, such as power and delay. Several research efforts [48][6][70] have proposed conceptually different non-destructive GLC techniques. However, none of them are capable of characterizing all the gates due to insufficient diversity of linear equations that correspond to power or delay measurements. We have developed a group of new GLC approaches for characterizing

all the gates in a target circuit [88][89][90][92][96] [93]. To the best of our knowledge, this is the first report of a technique that guarantees a complete gate-level characterization.

3.2 Gate-level Power Characterization

Our goal in gate-level power characterization is to characterize the power scaling factor of each gate, which is the deviation of the gate leakage power or switching power from the nominal specifications due to process variation.

3.2.1 Objectives

We have the following two primary objectives in power GLC:

- *Accuracy.* The results of characterization must be accurate, i.e., the difference between the characterized scaling factors and their actual values is minimal. Since there exist measurement errors, we must filter out the noises of errors.
- *Number of characterized gates.* Our goal is to characterize all the gates in the target circuit. In most cases, this objective is challenging due to the fact that there are a large portion of gates in the circuit that are correlated and thus have extremely low individual observabilities.

3.2.2 Power Measurements and Equations

We begin our power GLC approach by applying m different input vectors that are stored in flip-flops to the combinatorial logic and measure the total leakage power of the circuit for each of them. Next, we generate a system of m equations and

formulate a linear program (LP). The objective function of the linear program is to minimize the sum of the absolute value (l_1 -norm) of measurement errors, as shown in Equation (3.1), where m is the number of measurements, and e_i is the error of the i -th measurement.

$$\min \sum_{i=1}^m |e_i| \quad (3.1)$$

The system of linear equations (i.e., the constraints in the LP) can be formulated as the following:

$$K \cdot s = \tilde{p} + e \quad (3.2)$$

where $K \in R^{m \times n}$ is the nominal design values represented by a matrix of coefficients, which depend on gate types and their input vectors and can be found in a lookup table [104]; m is the number of measurements; n is the number of gates on the chip; s , \tilde{p} , and e are one-dimensional vectors representing the scaling factor of each gate, the measured power, and the unknown measurement error in each measurement, respectively. The format of (3.2) meets that of a linear constraint in a LP. Note that we abstract the impact of PV on leakage power (or switching power) using a scaling factor s_i for each gate i in the circuit. Our goal in GLC is to characterize the scaling factors of all gates by solving the LP. After obtaining the scaling factors, we can recover the manifestational properties due to PV from their nominal design values.

Note that the IC design specification may specify a range of nominal values (i.e., the K values) accounting for the process variation or other possible variations during the manufacturing process. In our GLC formulation, we consider this variation (i.e., the range of nominal values) by employing the scaling factor

s for each gate, which is exactly the set of variables we solve in the LP. However, our GLC formulation does require a constant nominal value for each type of gates under each combination of inputs, which appears as the coefficient matrix K of the linear equations (i.e., Equation (3.2)). In this case, the range of nominal values in the design specification cannot be used directly. Our solution to this problem is to use a representative value (e.g., the average) of the nominal range. Once we solve the GLC equations, the variation of the nominal range will be reflected in the scaling factor values.

3.2.3 Technical Issues in Power GLC

The structure of matrix K and thus the formulation of the LP are highly dependent on the selection of input vectors. In order to characterize the scaling factors accurately, one must minimize the dependencies amongst the variables in the system of equations. A simple way of achieving the goal is to create as many equations as possible. However, this technique has two strong negative ramifications:

- *Large-Size LP.* The number of gates is large in modern IC designs. Hence, the formulated LP may easily exceed the processing power of LP solvers.
- *Correlations.* Even if we are able to handle a large number of equations, ideally all the possible input vectors, we still cannot characterize the gates that are correlated in the system of linear equations. We define two types of correlations: (1) between gates that have the same ratio of coefficients in all the equations (collinearity correlation); and (2) between gates for which we are not able to obtain a sufficient number of independent equations due to the fact that the number of variables is larger than the number

of equations (insufficient controllability). We observe that the collinearity correlation often occurs when multiple gates are of the same type and have the same input signals. Also, the insufficient controllability correlation is a consequence of the IC structure where a subpart of the circuit has many gates but few intermediate inputs that control them.

The running times and coverage issues in the pertinent set of equations requires us to reduce the size of the LP and to break the correlations. We address the first issue by pre-processing the input vectors in such a way that we maximize the number of unique coefficients in front of each variable. Also, we address the second issue using correlation detection and thermal conditioning.

3.2.4 Power Characterization Overall Flow

In order to achieve the objectives in GLC and address the technical issues, we develop a power GLC approach that includes three phases, namely pre-processing, GLC, and post-processing, as shown in Figure 3.1. In the pre-processing phase, we first generate the IC instances that take into account the impact of PV. We combine the PV model for individual devices presented by Asenov et al. [7] with the spatial correlation model proposed by Cline et al. [24]. Next, we generate a set of input vectors that can be applied to obtain various leakage voltage values. The goal during input vector generation is to ensure that the maximum number of gates have all their possible input signal combinations, so that the possibility of linear dependencies in the system of linear equations is minimized. Also, we selectively heat up the circuit by switching certain gates on the circuit in such a way that all the remaining linear dependencies among the linear equations can be resolved. After pre-processing, we begin the process of GLC, in which we apply the set of input vectors and measure the total leakage power for each of

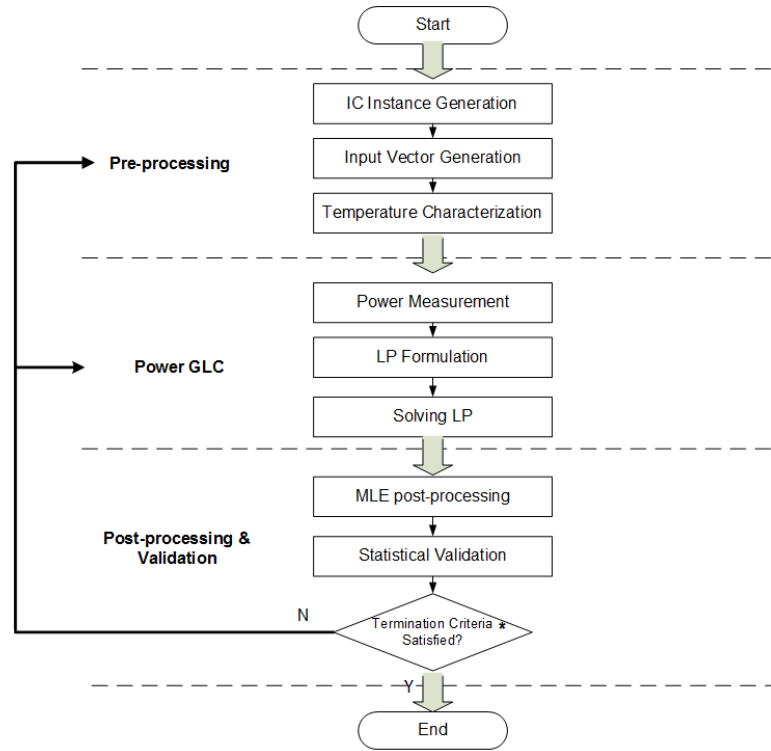
them. For each measurement value, we formulate a linear equation by adding the leakage power values of all the gates with the measurement error. Next, we solve the system of linear equations using a LP solver and obtain results for each PV scaling factor.

We repeat the GLC procedure multiple times and conduct post-processing using the obtained results. We apply maximum likelihood estimation (MLE) that selects the most likely scaling factor for each gate as our eventual results of GLC. Finally, we employ statistical methods to validate our prediction results. For this purpose we use resampling, where 60% of the GLC results are used for the training set and 40% for the testing set. The entire GLC procedure terminates when the validated GLC accuracy is within a user predefined threshold value.

3.2.5 Correlation Detection

Since the correlated variables in the system of linear equations cannot be solved by the LP solver, we detect them so that we can either break the correlations or, in the worst case, combine those variables to reduce the size of the LP. We have developed two techniques to detect the collinearity correlations. The first one is straightforward: we check the coefficients for all pairs of variables using exhaustive enumeration. If there exists a pair of gates for which the ratio of coefficients are identical over all the equations, the pair of gates is correlated. The second technique employs our LP formulation itself. We add one more constraint in the LP formulation that sets one of the potentially correlated gates to a very large value; if correlations exist, the LP solution would show that several other variables become very small. Therefore, these gates are correlated with the gate whose value has been modified by the extra constraint.

For the insufficient controllability correlations, the detection is not trivial, as



* The iterative process terminates when the validated GLC accuracy is within a predefined threshold value

Figure 3.1: Overall flow of our GLC scheme. We use a three-phase procedure (pre-processing, GLC, and post-processing).

the number of subparts of the circuit that can possibly have correlations is large. We solve this problem by manipulating the objective function. In particular, we change the objective function to maximizing only one of the variables. If a subset of the gates in the circuit have the insufficient controllability correlation, the other variables would become very small.

3.2.6 Correlation Elimination

In this section, we discuss our thermal conditioning approach to GLC that resolves the correlation issues in the system of linear equations.

3.2.6.1 Thermal Conditioning Overview

As discussed in Chapter 3.2.3, there are two technical issues that we must resolve in GLC. Firstly, if the target circuit is large, it requires a large number of measurements as well as a very large LP that is difficult to solve. Secondly, we must break the correlations in the system of linear equations in order to characterize all the gates on the circuit. The only way to break these correlations is to find alternatives to supplement input vector variation for changing the coefficients of the scaling factor variables.

We solve both technical issues using thermal conditioning, where we heat up a subset of gates to change their coefficients in the system of linear equations. Our intuition is based on the fact that gate-level leakage power depends on the temperature of the gate (as shown in Equation (2.1)) and that IC heat-up is much faster than the cooling process (as discussed in [62]). Therefore, thermal conditioning provides us with an additional means of controlling a subset of gates in the circuit and thus enables us to change the coefficients of the scaling factors in the system of linear equations. By using thermal conditioning, we can obtain

sufficient numbers of equations regardless of the number of input vectors that can be applied. Furthermore, we are able to break the collinearity correlation, since we can obtain different coefficients even without changing the input vectors.

We show our flow of thermal conditioning in Figure 3.2. We first conduct correlation detection using the techniques discussed in Chapter 3.2.5 to determine the subset of gates that are either subject to collinearity correlation or insufficient controllability. Next, we perform thermal conditioning on the set of correlated gates by applying a set of input vectors that cause the gates to switch. The heat generated while switching increases the temperatures of the gates. In order to calculate the new coefficients of the scaling factors in the system of linear equations, we select a subset of gates in the circuit as the representative gates, which can provide us with the temperature profile of the entire circuit (as shown in Algorithm 1). We characterize the new temperatures of the subset of gates by measuring leakage power, switching power, and delay and by solving a system of non-linear equations following the power and delay models discussed in Chapter 2. We utilize the characterized temperatures of the gates as representative temperatures and determine the temperature of each gate in the circuit under the consideration of heat transfer. Finally, we apply the new coefficients to GLC using leakage power measurements and characterize the scaling factor of each gate.



Figure 3.2: Flow of thermal conditioning for GLC. We increase the temperatures of a subset of gates in the circuit to break the correlations in the system of linear equations.

3.2.6.2 Thermal Conditioning Using Gate Switching

We conduct input vector control to increase by different amounts the temperatures of the subset of gates that are subject to correlations in the system of linear equations. In particular, we select a set of input vectors in such a way that they can switch the set of correlated gates identified by correlation detection in different ways. The heat generated during switching increases the temperatures of the gates and thus change their coefficients in the system of linear equations. The key observation is that the time needed for gate switching is very fast (on the order of nanoseconds), while the cooling process is much slower (on the order of seconds) [62]. Therefore, we can increase the temperatures of the subset of gates rapidly and assume that the new temperatures we obtain stay constant for seconds until we completely characterize all gates in GLC.

3.2.6.3 Temperature Characterization Using Leakage Power, Switching Power and Delay Measurements

In order to calculate the new coefficients in the system of linear equations after thermal conditioning, we must determine the temperature profile of all the gates in the circuit. There are three variables in the gate-level properties, as shown in Equations (2.1), (2.2), and (2.4), temperature (T), effective channel length (L) and threshold voltage (V_{th}). From these three equations, we are able to solve for temperature T . The formulation of these three equations requires that we obtain gate-level leakage power, switching power, and delay. Our approach for temperature characterization is nondestructive and does not require complicated thermal models or thermal management tools. We first select a subset of gates for which we can characterize all the three properties using GLC, i.e., the gates that are on critical path, switch often, and do not have correlations in the sys-

tem of linear equations with other gates in terms of leakage power. Next, we conduct gate-level leakage power, switching power, and delay characterization of the selected gates using the GLC method. After obtaining their gate level leakage power, switching power, and delay, we formulate three non-linear equations according to Equations (2.1), (2.2), and (2.4) for each gate and solve for the variables T , L , and V_{th} . Finally, we use the temperatures of the selected gates as the representative temperatures and determine the temperature profile of the entire circuit, under the consideration that the gates that are physically close to each other on the circuit have similar temperatures due to heat transfer. Algorithm 1 shows the pseudocode for selecting the representative gates and calculating the temperatures.

3.2.6.4 Gate Level Leakage Power Characterization

After obtaining the new temperatures of the gates, we follow Equation (2.1) to set the new coefficients in the system of linear equations. In this way, we are able to create various independent linear equations that break both types of correlations.

3.2.7 Improving the Objective Function

So far we have been using the $l1$ -norm of measurement errors as the objective function in the LP. Although the $l1$ -norm helps reducing the large errors, it is not capable of reflecting the actual measurement errors. In order for the optimization process to follow exactly the measurement error distribution, we consider the likelihood function of the measurement error distribution in our objective function. Our goal is to find the solution of maximum likelihood. For example, if we assume the error follows a triangular distribution, the objective function can be formulated as the following:

Algorithm 1 Temperature Characterization: We conduct leakage power, switching power, and delay measurements on the pertinent IC and formulate three non-linear equations with variables T , L , and V_{th} . We calculate T by solving the three non-linear equations.

Input: Target circuit for characterization.

Output: Temperature T_i of each gate i .

- 1: **repeat**
 - 2: Select a set I of input vectors;
 - 3: Determine (via correlation detection) $S_{leakage}$, the set of gates that do not have correlations with other gates in the system of linear equations in leakage power GLC;
 - 4: Determine (via simulation) $S_{switching}$, the set of gates that switch with more than $k\%$ probability when I is applied;
 - 5: Determine (via simulation) S_{delay} , the set of gates that are on critical path with more than $k\%$ probability when I is applied;
 - 6: $S_{rep} = S_{leakage} \cap S_{switching} \cap S_{delay}$;
 - 7: **until** $S_{rep} \neq \emptyset$;
 - 8: Conduct leakage power, switching power, and delay characterization using GLC;
 - 9: **for** each gate i in S_{rep} **do**
 - 10: Formulate leakage power equation using Equation (2.1);
 - 11: Formulate switching power equation using Equation (2.2);
 - 12: Formulate delay equation using Equation (2.4);
 - 13: Solve the three equations for T , L , and V_{th} using non-linear programming;
 - 14: **end for**
 - 15: Calculate the new temperatures T_i for all gates in the circuit using S_{rep} ;
 - 16: **return** T_i ;
-

$$\max l(e_r) = \sum_{i=1}^m \log(e_s + p(|e_r|)) \quad (3.3)$$

where $p(\cdot)$ is the probability density function of the triangular distribution. The new objective function is non-linear and cannot be handled by the LP solver directly. Our solution is to create a piecewise linear function that approximates the non-linear function. In particular, we find a subset of breakpoints on the non-linear curve. By connecting these breakpoints using piecewise linear segments, we obtain a linear form of the objective function. Figure 3.3 shows an example of the likelihood function and the piecewise linear approximation.

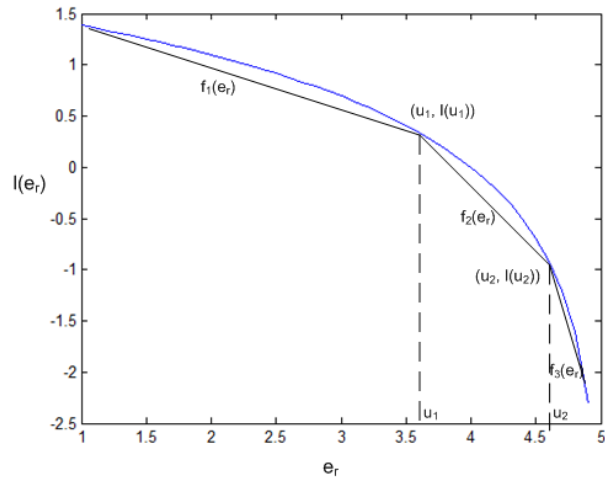


Figure 3.3: An example of the likelihood function and the piecewise linear approximation, where $l(e_r) = \log(5 - e_r)$, and two breakpoints u_1 and u_2 are being considered.

We define the problem of finding the optimal breakpoints as an optimization problem, where we minimize the approximation error in the following form:

$$\min d_N(a, b) = \int_a^b (l(e_r) - f(e_r))^2 de_r \quad (3.4)$$

where a and b are parameters of the triangular distribution; $l(e_r)$ is the non-linear likelihood function; and $f(e_r)$ is the piecewise linear function determined by the breakpoints. In particular, we formulate $f(e_r)$ as the following expression.

$$f_i(e_r) = a_i e_r + b_i, \quad u_{i-1} \leq e_r \leq u_i \quad (3.5)$$

where $f_i(e_r)$ is the i -th segment of the piecewise linear function. Suppose we have N breakpoints and thus $N + 1$ linear pieces. $u_i (1 \leq i \leq N + 1)$ are the breakpoints, with $u_0 = 0$ and $u_{N+1} = b$. a_i and b_i are parameters determined by u_{i-1} and u_i .

The problem of finding breakpoints in the piecewise linear approximation can be solved provably optimally in polynomial time using dynamic programming [13]. The linearization is sufficient to obtain a piecewise linear representation of any arbitrary function. However, it requires the error function to be convex in order to guarantee the optimality of the results.

3.2.8 MLE Post-processing

The results obtained from the LP are impacted by several factors including the precision of the LP solver and the accuracy of the power measurements. In order to obtain more accurate results, we post-process the characterization results using maximum likelihood estimation (MLE). In particular, we repeat the GLC procedure multiple times and collect the results from the LP solver. Next we apply goodness-of-fit tests on the data from each run, and estimate the statistical distribution of the scaling factors over different runs. According to the distribution that each scaling factor follows, we create its approximate density function, namely $p(s_i)$, and set our estimated value of s_i to be the one that maximizes the

following likelihood function:

$$\tilde{s}_i = \operatorname{argmax}_{s_i} \log p(s_i) \quad (3.6)$$

We repeat the MLE post-processing over all the scaling factors and obtain the final GLC values.

3.3 Gate-level Delay Characterization

In this section, we discuss in details our approach of conducting gate-level delay characterization for all the gates in the circuit.

3.3.1 Delay Characterization Overview

Delay (or timing) property of an IC has a higher resolution and lower noise when compared to the power property, because of the linear dependence of delays on PV and the limited number of components on each path under test. However, an unresolved difficulty of timing measurements is the inability of individually sensitizing and characterizing each component using the test vectors. This is because of the existence of parallel routes that reconverge to a single point which make it difficult to map the measured path delay to a specific path for the consideration of GLC. As shown in a small example in Figure 3.4, even though we can measure the delay between inputs $I1/I2$ and output O , we are unable to determine whether the measured delay is for path A or path B. In fact, since path B from gates 1-3-4-5-6-7 is much longer than path A (1-2-H-7), the addition of gate H can not be detected using delay measurements from inputs $I1$ and $I2$ to output O . The presence of PV would further complicate the case, since the delay of path A may be smaller than that of path B on one chip but could be

larger on another.

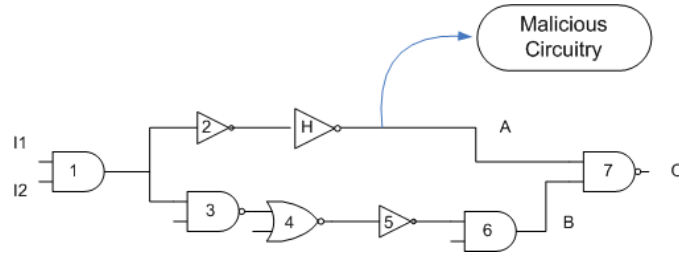


Figure 3.4: Example of uncharacterizable IC components using delay measurements.

We create a gate-level delay characterization approach that covers all the locations in the target circuit. In particular, we introduce a test point (TP) insertion methodology that separates the parallel paths and enables their observability via delay measurements. Next, we develop an input vector selection scheme for each single path by transforming the problem to a Boolean satisfiability (SAT) problem. Leveraging SAT problem formulation, we obtain gate-level delay properties for all the gates in the circuit. Furthermore, we employ a circuit partition method to scale down the problem space to a limited number of non-overlapping regions. In this way, we ensure that our timing characterization at all circuit locations is scalable to large industry IC designs. To summarize, our innovations and contributions include:

- a provably complete test points insertion to break the reconvergences and identify the measured paths;
- a SAT-based input vector selection for characterizing each single path; and
- a region-based circuit partition approach to reduce the overhead and ensure the scalability of the timing characterization.

3.3.2 Delay Paths Identification and Selection

In order to characterize the timing of all the existing gates under the impact of PV, we design a systematic way of identifying the delay paths, break the reconvergence points, and characterize the gate-level delay properties.

The overall flow of our approach is shown in Figure 3.5. We first partition the circuit into smaller regions, and for each region, we analyze the structure of the netlist and identify two types of delay paths between a specific pair of input and output: (1) those that only include one single path and thus are characterizable by direct delay measurements; and (2) those that include multiple paths in parallel and thus are not differentiable by direct delay measurements. After obtaining the two groups of paths, we first conduct GLC for the path group (1) by leveraging SAT for determining the input vectors that switch the gates and create multiple independent linear equations. Next, we select paths from group (2) to cover all the remaining gates in the circuit. In order to make the paths in group (2) characterizable, we insert additional test points at the reconvergence point of each set of parallel paths. As a result, any path that is originally in parallel with other paths would get an additional observation point for delay measurements. The inserted test points enable us to treat the parallel paths the same as the single paths in group (1). Thus, we would be able to conduct GLC on these paths and characterize all the gates.

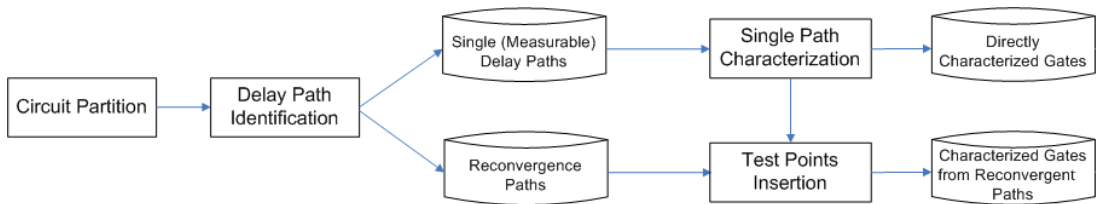


Figure 3.5: Overall flow of gate-level delay characterization.

3.3.2.1 Test Point Structure Design

To measure the delay for one of the parallel paths, we insert a test point flip-flop (D-type) at each reconvergence point. In particular, we feed the data input of the D flip-flop with the output of the last gate on the path which enables us to measure the path delay by using clocking and standard scan chain delay-fault testing methods.

We demonstrate a small example in Figure 3.6 regarding the test point insertion method that breaks initially unmeasurable delay paths (due to reconvergence). In this example, path a ($3 \rightarrow 10 \rightarrow 22$) and path b ($3 \rightarrow 11 \rightarrow 16 \rightarrow 22$) originate from the same input and reconverge to the same output. Consequently, if we measure the delay from input 3 to output 22, it is difficult to determine whether the measured delay is for path a or for path b ; therefore, we would not be able to create correct linear equations for the GLC of these two paths. We address this issue by inserting a D flip-flop at the end of each path, i.e., at pin 10 and pin 16. Now, we are able to bypass the reconvergence problem by measuring the delay between 3 and 10 (for path a) and between 3 and 16 (for path b) separately. This enables us to treat both paths a and b as single paths and create a system of linear equations to find their gate-level delay characteristics.

3.3.2.2 Reconvergence Identification

Reconvergence identification is an important step in our test point-based delay characterization. To do so, we analyze the structure of the netlist and identify the reconvergence points between each input/output pair. Assuming that the circuit netlist is a directed graph, with each interconnect as an edge $e_i \in E$, and each gate (or input, output) as a node $n_i \in N$, we define a reconvergence point as a node in the circuit graph that has an in-degree larger than 1.

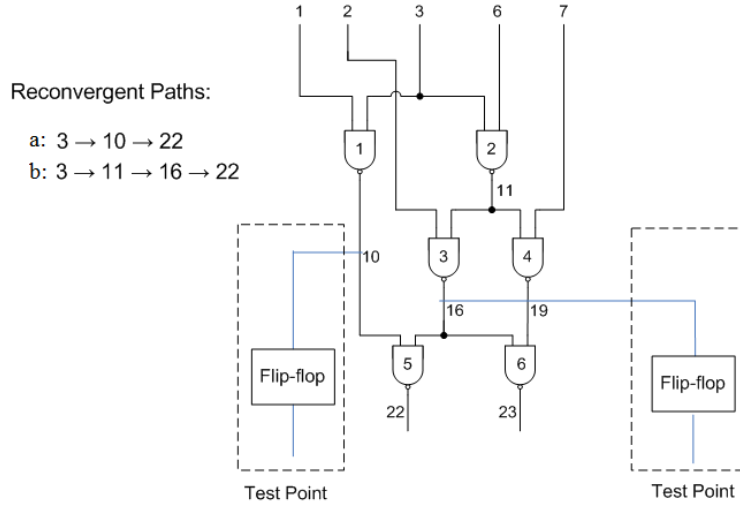


Figure 3.6: Example of test points insertion for delay characterization.

The identification of reconvergence points in the target circuit is straightforward, because it is already implied in the structure of the circuit graph whether a node is a reconvergence point or not. However, for the purpose of delay characterization of all gates, it is infeasible to insert test point at every reconvergence point, since it would create a very high overhead. Our goal in test point insertion is to minimize the number of added test points based on the consideration that a subset of gates are already characterizable by single path characterizations, and the fact that each gate may appear in multiple paths.

Our test point insertion algorithm is shown in Algorithm 2. We first characterize all gates that are measurable on single paths. Then, for the remainder gates, we search the circuit graph and find their transitive fan-in (inputs that control these gates) and transitive fan-out (outputs that these gates drive either directly or indirectly). Next, for each pair of transitive fan-in and transitive fan-out, we develop a backtracing-based search algorithm to determine all the paths between them. We conduct a depth-first search from a specific output towards

the inputs. During this process, we keep pruning the edges using backtracking to trace all the possible paths towards a specific input. After determining the list of paths for each pair of transitive fan-in and fan-out of all the unsolved gates, we obtain a list of candidate paths that we may consider to solve by adding test points. Next, we sort the list of paths in an ascending order by their length (i.e., the number of unsolved gates on the path) and (in the same order) use GLC to characterize their delay. Note that we conduct GLC for these paths in an iterative and incremental manner. In other words, the solved gates in the original paths are considered known in the next iteration, so that the linear system’s size can be reduced because many paths would have overlapping gates. The ascending order requires less run time and overhead since it is easier to solve the overlapping gates on a shorter path.

3.3.3 Delay GLC Using Linear Programming

In GLC [88, 89, 92, 96], the measured side-channel value is decomposed to its constituent components. For example, let us assume that a test vector sensitizes a path consisting of an interconnection of K gates where each gate changes its state after applying the incident input. For simplicity of example, let us ignore the wire delays for the moment. The overall measured path timing can be written as the sum of the delays of the K individual gates with an added measurement error term. Similarly, it is possible to test multiple paths and then write a linear system of timing equations. Noninvasive GLC aims at finding the individual gate delay values from the noisy equations by forming a linear optimization that attempts to minimize the measurement errors.

Assume the j -th test vector ($j = 1, \dots, J$) measures the delay of the path P_j denoted by $T_{meas}(P_j)$ and the measurement incurs the error $err_T(P_j)$. The path

Algorithm 2 Reconvergence identification and test point insertion for characterizing all gates.

Input: Circuit Netlist (Net), Gate Set (G);

Output: A set of test points (TP) for delay characterization;

- 1: $G_1 \leftarrow$ all measurable gates on single paths;
 - 2: Characterize G_1 using delay measurements;
 - 3: **for** each gate $g_i \in G - G_1$ **do**
 - 4: $TI_i \leftarrow$ transitive fan-in of g_i ;
 - 5: $TO_i \leftarrow$ transitive fan-out of g_i ;
 - 6: Find all the paths P_i between TI_i and TO_i using backtracking-based depth-first search;
 - 7: $len_i \leftarrow$ number of gates in $G - G_1$ that are covered by P_i ;
 - 8: Insert P_i in P ;
 - 9: **end for**
 - 10: Sort P in ascending order based on len_i ;
 - 11: $i \leftarrow 0$;
 - 12: **while** Not all gates in $G - G_1$ are characterized **do**
 - 13: Insert test point TP_i to separate paths P_i ;
 - 14: Insert TP_i in TP ;
 - 15: Characterize all gates covered by P_i ;
 - 16: $i \leftarrow i + 1$;
 - 17: **end while**
 - 18: Return TP ;
-

consists of the gates G_{k_j} with delay $T(G_{k_j})$, where $k_j = 1, \dots, K_j$ is the index of the elements on path j . The optimization problem objective function (OF) and constraints are then:

$$\begin{aligned}
 \text{Objective Function : } & \min_{1 \leq j \leq J} \mathcal{F}(\text{err}_T(P_j)) & (3.7) \\
 \text{Constraints : } & \sum_{k_j=1}^{K_j} T(G_{k_j}) = T_{\text{meas}}(P_j) + \text{err}(P_j), \\
 & P_j = \{G_{k_j}\}_1^{K_j}, \quad 1 \leq j \leq J.
 \end{aligned}$$

\mathcal{F} is a metric for quantifying the measurement errors; commonly used form of \mathcal{F} is the l_1 or l_2 norm of errors. $T(G_{k_j})$ is sometimes expressed as a product of its nominal value $T_{\text{nom}}(G_{k_j})$ and a scaling factor (because of PV) $\delta(G_{k_j})$, i.e., $T(G_{k_j}) = \delta(G_{k_j})T_{\text{nom}}(G_{k_j})$.

3.3.4 SAT-based Approach for Characterizing All Gates

In this section, we discuss in details our SAT-based input vector selection approach for characterizing all the gates in the circuit. The approach operates on the circuit with inserted test points. Our goal is to select input vectors that switch the gates in such a way that independent linear equations can be created to characterize all gates.

3.3.4.1 Input Vector Selection Problem Formulation

For gate-level delay characterization of a single path, or separated reconvergent paths by using test points, we must select a set of input vectors that switch all the gates on the measured path, so that we can create the linear constraints as described in Equation (3.7). To ensure that the gate-level delays are solvable from the linear program, the coefficients in the linear constraints must be independent from each other, or in a more formal statement, the rank of the matrix formed

by $\delta(G_{k_j})$ must be larger than the number of gates on the measured path.

Since the nominal delay values (i.e., the coefficients in the LP constraints) are dependent on the switching patterns of the corresponding gates, in order to increase the rank of the coefficient matrix, we must create a variety of independent combinations in terms of the switching patterns for all the gates. The input vector selection problem can be formulated as follows:

Input Vector Selection Problem. Given an IC with N gates, where each gate i ($1 \leq i < N$) has a required signal for each of its inputs in order to create solvable linear programs in the form of Equation (3.7), the input vector selection problem aims to find the input vectors that satisfy the signal requirements of all the gates.

3.3.4.2 Input Vector Selection Using SAT

In order to solve the input vector selection problem, we convert it to a SAT problem, where the signal of each gate can be represented by a Boolean expression in a clause. The SAT problem aims to find all the variable (gate/pin signal) values that evaluate all the clauses to be true. In particular, the SAT problem is formulated as follows:

$$C_1 \wedge C_2 \wedge \dots \wedge C_m = true \quad (3.8)$$

and

$$C_i = \begin{cases} O_i(I_1, I_2, \dots, I_k), & s_i = 1; \\ !O_i(I_1, I_2, \dots, I_k), & s_i = 0; \\ 1, & s_i = \text{don't-care}; \end{cases}$$

where C_i is the clause for setting pin i to its objective signal; $O(I_1, I_2, \dots, I_k)$ is

the Boolean expression for the signal of pin i based on inputs I_1 to I_k ; s_i is the objective signal of pin i ; m is the number of pins in the circuit; and k is the number of primary inputs of the circuit.

The solution of the SAT problem provides us with specific input signals of $O(I_1, I_2, \dots, I_k)$ that satisfy Equation (3.8), or in certain cases, reports that the solution is infeasible for the specified objective signals. SAT problem is an NP-complete problem, for which there has been a large number of SAT solvers proposed in the SAT community. For the discussion in this dissertation, we employ sat4j [77] to solve the SAT problem in finding the input vectors.

By solving the SAT problem using a SAT solver, we can determine the input vectors that create the switching patterns. In the case where the resulting SAT problem is not solvable, we iteratively add additional test points to break the single path into multiple paths, until all the paths are characterizable using delay measurements. In this way, we obtain a provably complete coverage of all the gates in the target circuit in terms of delay characterization.

3.4 Scalability Techniques for GLC

Scalability is one of the major concerns in GLC, as it typically requires a significant number of power/delay measurements as well as solving a large set of equations in order to obtain the gate-level properties. It is non-trivial to make this process scale to large industrial IC designs with millions of transistors. We solve the scalability issue in GLC as well as in the hardware security applications by using segmentation techniques. The main idea is to employ the divide-and-conquer paradigm, in which we partition the circuit into multiple small components and characterize each of them. In particular, we have developed three sets

of techniques for segmenting the circuit into small regions, including (1) Segmentation for power characterization by conducting input vector control; and (2) Region-based circuit partition for delay characterization.

3.4.1 IC Segmentation by Conducting Input Vector Control

The segmentation of an IC is based on the divide-and-conquer paradigm, in which we divide a large IC into multiple small segments and characterize each of them using GLC. Segmentation can be implemented using input vector control, where we freeze the signals of a subset of inputs and vary the other. Consequently, only the gates controlled by the varying inputs would change their coefficients in the system of linear equations, while the other gates would have identical coefficients in all the equations. Therefore, we can represent all the frozen gates using a single variable in the system of linear equations. In this way, the size of the LP is greatly reduced, to the extent that can be handled by LP solvers. Figure 3.7 shows an example of our segmentation method. We obtain Segment 1 (gates X1, X2, and X5) by freezing inputs 3 and 4 and by applying different input vectors to inputs 1 and 2. Similarly, we obtain Segment 2 (gates X3, X4, and X5) by freezing inputs 1 and 2.

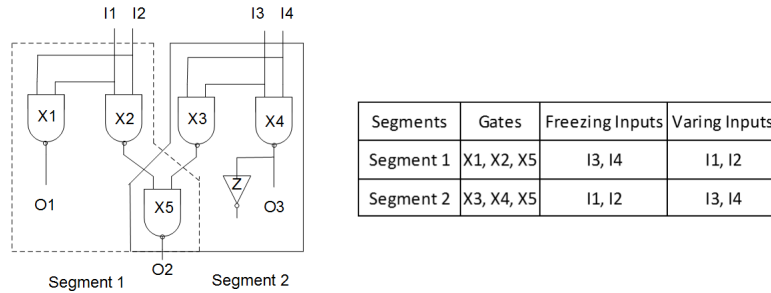


Figure 3.7: Example of segmentation using input vector control.

3.4.2 Region-based Circuit Partition for Delay Characterization

3.4.2.1 Motivation of Circuit Partition for Delay Characterization

We note that we must address the following three issues regarding the overhead and scalability in our delay GLC approach, in order to provide a reliable and scalable characterization solution.

- *Area overhead.* The area overhead caused by the inserted test points (i.e., flip-flops) cannot be ignored, as the area of a flip-flop is often 4 to 6 times larger than a regular gate. It is essential to take into account of this critical overhead toward the design and manufacture of the ICs.
- *Test Time.* The test time required by the delay measurements is an important metric for the cost of the proposed approach. Since it requires input vector control to conduct gate-level timing characterization, the GLC of multiple paths cannot be fully parallelized. Therefore, we must minimize the total number of measured paths to reduce the test time while still covering all circuit locations.
- *Scalability.* the search of reconvergence points in all circuit locations is a NP-complete problem that raises scalability concerns. The complexity grows exponentially with the increase of the number of inputs, outputs, or gates. The efficiency and scalability of the identification procedure must be improved for the consideration of large designs, e.g., with millions of transistors, in the modern IC industry.

To address these issues, we develop a circuit partition scheme that partitions the large IC into smaller regions, so that the scope of the timing characterization is reduced to the partitioned regions to address the scalability issue. Also, we

aim to minimize the number of test points that we must add during the course of circuit partition, while the goal is still to cover all the circuit locations in terms of gate-level timing characterization.

3.4.2.2 Circuit Partition Problem Formulation for Timing Characterization

We define the circuit partition problem as a specialized graph (netlist) partition problem with the goal of minimizing the required test points while controlling the number of gates in each region:

Circuit partition problem for timing characterization. Given a graph $G = (V, E, PI, PO)$ that represents the netlist of an IC, where V is the set of vertices (gates), E is the set of edges (wires), PI is the set of primary inputs, and PO is the set of primary outputs, find a graph partition that consists of k subgraphs (regions) so that (1) each region consists of $P_i (i = 1 \dots k)$ paths, which ensures that the total number of source and destination nodes of the paths that do not belong to $PI \cup PO$ is minimized; and (2) the number of gates in each component $N_i (i = 1 \dots k) < Th$, where Th is a configurable threshold determined by the gate-level timing characterization approach.

After partitioning the circuit, we can conduct the reconvergence identification and delay characterization in the scope of each single region. Since the size of the problem domain is reduced by factor of k , we argue that the complexity of the identification and characterization processes is reduced exponentially.

3.4.2.3 Circuit Partition Method

Our intuitions in addressing the circuit partition problem are three-fold. First, we should find regions that contain paths traversing from primary outputs toward the direction of primary inputs. In other words, the addition of nodes in the region during the search process should go vertically (i.e., depth first). This is based on the consideration of leveraging as many primary inputs and outputs in order to reduce the additional test points. Second, the number of gates in each region should be maximized as long as it can be handled by the characterization approach (i.e., below threshold Th), which would reduce the number of regions and thus the number of delay measurements. Third, it is beneficial that there is no or little overlap between different regions, in order to reduce unnecessary measurements and characterizations.

Based on the above intuitions and thoughts, we develop a circuit partition method using maximum fanout free cones (MFFCs) [101]. a MFFC for a node g_i is a sub-netlist where each node g_j is a transitive fan-in of g_i , and all the transitive fan-out's of g_j are included in the MFFC. In other words, a MFFC is a self-contained region that grows maximally from PO toward PI of the circuit G , which satisfies our intuitions 1 and 2. Furthermore, the MFFCs have an important property that two different MFFCs are either non-overlapping or one contains the other, which satisfies our intuition 3 as long as we remove the smaller MFFC from consideration in terms of circuit partition. Our circuit partition algorithm using MFFCs is presented in Algorithm 3. We first find the MFFC for each gate in the circuit using known algorithms [101]. Then, we sort the found MFFC set in the ascending order in terms of the size (i.e., number of gates) in the MFFC. Finally, we remove the MFFCs that have been fully covered by at least one other MFFC to remove the redundancy.

Algorithm 3 MFFC-based circuit partition.

Input: Circuit Netlist (Net), Gate Set (G);

Output: A circuit partition with k regions;

- 1: **for** each gate g_i in G **do**
 - 2: Find the MFFC C_i for g_i [25];
 - 3: **end for**
 - 4: Sort the MFFC set $C = \{C_i | i = 1 \dots N\}$ for each node in ascending order in terms of the number of gates involved;
 - 5: **for** each $C_i \in C$ **do**
 - 6: **if** $\exists C_j$ where $C_j \supset C_i$ **then**
 - 7: Remove C_i from C ;
 - 8: **end if**
 - 9: **end for**
 - 10: Output C as the circuit partition.
-

Figure 3.8 shows a small example of the proposed circuit partition method using MFFC. We partition the 6-gate circuit into 3 regions. Each region is a maximum fanout free cone from the bottom node, except that gate 2 and gate 3 are MFFCs individually, and we merge them to form a single region in order to reduce the number of test points. With this partition, the delay of each path is measurable, and it only requires one test point at location 16, since all other observation points are primary inputs or outputs of the target circuit.

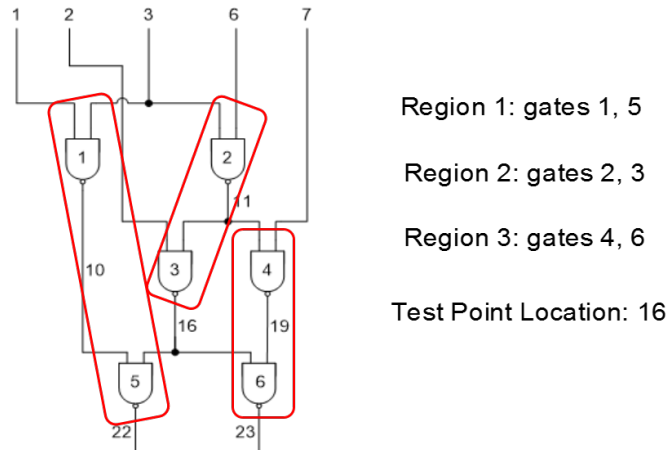


Figure 3.8: Example of MFFC-based circuit partition for timing characterization.

3.5 GLC Evaluation Results

In this section, we discuss the simulation results of our power and delay GLC approaches.

3.5.1 Power GLC

We evaluate our gate-level power characterization approach scheme on a set of ISCAS [16][15] and ITC [38] benchmarks. We employ the triangular distribution

with mean value 1% as our measurement error model.

The accuracy of characterization is evaluated using the relative characterization error that is calculated using the following formula:

$$Error_i = |s_{calc_i} - s_{real_i}| / s_{real_i} \quad (3.9)$$

where $Error_i$ is the relative characterization error; and s_{calc_i} and s_{real_i} are the calculated scaling factor of gate i and its real value, respectively. The resulting error over all gates in the circuit is calculated as the average of all the $Error_i$:

$$Error_{avg} = \frac{1}{n} \sum_{i=1}^n Error_i \quad (3.10)$$

where n is the number of gates in the circuit, and $Error_{avg}$ is the average result error for GLC. In the rest of this section, we use $Error_{avg}$ to evaluate the accuracy of GLC. The simulation results are shown in Table 3.1. We observe that the GLC errors are within 2%. It ensures accurate characterizations of the gate-level leakage power property, which we later use in the consistency-based analysis for hardware security applications.

3.5.2 Delay GLC

We evaluate our delay GLC approach on a set of ISCAS and ITC benchmarks. For each benchmark, we simulate the 45nm technology and generate the variation of threshold voltage following the Gaussian PV model [7]. Also, we consider the spatial correlation of effective channel length following the quad-tree model [24]. Furthermore, we add test points to the original design netlist to make all the gates visible for delay measurements and characterization.

We consider the following evaluation metrics: the GLC coverage (i.e., how

Table 3.1: Accuracy of gate-level power characterization.

Benchmark	Gates	Inputs	Outputs	GLC Error (%)
C17	6	5	2	0.0057
C432	160	36	7	0.11
C499	202	41	32	0.26
C880	383	60	26	0.34
C1355	546	41	32	0.40
C1908	880	33	25	0.98
C2670	1193	233	140	0.75
C3540	1669	50	22	1.72
C5315	2307	178	123	0.52
C6288	2416	32	32	0.13
C7552	3512	207	108	0.39
S526	214	3	6	0.33
S832	292	18	19	0.73
S38584	19253	12	278	0.20
b17	32192	37	97	0.60

many gates we are able to cover in terms of the delay GLC), characterization accuracy (i.e., how much is the characterization error between the characterized delay values and the actual ones), and the overhead of test point insertion (i.e., how much is the increase of delay due to the inserted test points).

3.5.2.1 GLC coverage

Table 3.2 compares the number of gates that are covered by the characterization approach with and without test points inserted. In case of no test points, the only possibility to conduct delay characterization is that there are no reconvergences from a specific input to a specific output in the original design. It can be seen that there is no full coverage of gate delays in any of the tested benchmarks. However, after we insert test points to break the reconvergences, we could characterize all the gates in each benchmark.

3.5.2.2 Accuracy of Delay Characterization

The rightmost column in Table 3.2 shows the average characterization errors in the test point-based approach. The simulations were conducted under the assumption of 1% delay measurement errors. Note that the delay measurement errors reported in [85] and [26] are far less than 1% and, therefore, we are overestimating the measurement errors that can be further improved in real scenarios. For all benchmarks, we observe less than 1% error for gate-level delay characterizations. The accuracy in delay characterization that covers all gates in the target circuits serves as a foundation for hardware security applications.

Table 3.2: Results of delay characterization with and without test points. The number of characterizable gates without test points is no more than 60% of all the gates. However, with inserted test points, we can characterize 100% of the gates accurately in all the tested benchmark circuits.

Benchmark	# Gates	# Characterized Gates (No TPs)	# Characterized Gates (With TPs)	GLC Error (%)
C499	202	122 (60.4%)	202 (100%)	3.1E-03
C880	383	178 (46.5%)	383 (100%)	0.10
C1355	546	0 (0%)	546 (100%)	0.10
C1908	880	141 (16.0%)	880 (100%)	0.10
C2670	1193	262 (22.0%)	1193 (100%)	0.98
C3540	1669	127 (7.61%)	1669 (100%)	0.95
C5315	2307	383 (16.6%)	2307 (100%)	0.11
C7552	3512	960 (27.3%)	3512 (100%)	0.51

3.5.2.3 Test Points Overhead

The test point-based delay characterization process incurs three sources of overhead: (1) the area cost for the inserted test points; (2) the test time required by the delay measurements; and (3) the delay increase on critical paths due to the insertion of test points.

We evaluate (1) and (2) by identifying the number of test points (i.e., flip-flops) and the number of measured paths, respectively. Figure 3.9 shows the area overhead caused by the inserted test points using the region-based method. Here we calculate the area overhead as the ratio between the transistor count of the inserted test points and that of the original design ¹. The results indicate that the area overheads in all tested benchmark circuits are below 25%.

In Table 3.3, we show the number of test points and characterized paths required in the region-based delay GLC approach. The results show that the region-based approach requires small numbers of test points and measured paths in order to characterize the gate-level delay properties. In particular, the number of measured paths is an important metric, since it is often the case that the measurements of multiple paths cannot be parallelized due to input vector control and, therefore, the overall test time is approximately the sum of test time on each path.

Furthermore, for the evaluation of (3), we further conducted simulations on the ICs with all the test points inserted in terms of the critical path delays and compare them with the original values before any test points are added. The results are shown in Figure 3.10. We can see from the results that the average delay increase is within 5% of the original delay.

¹We obtain the transistor count data for the flip-flop implementation from [71]

Table 3.3: Overhead of test point insertion in terms of the number of test points (i.e., area overhead) and the number of measured paths (i.e., cost of test)

Benchmark	# Gates	# TP	# Paths
C499	202	36	58
C880	383	25	41
C1355	546	68	58
C1908	880	82	49
C2670	1193	65	46
C3540	1669	155	125
C5315	2307	354	313
C7552	3512	296	155

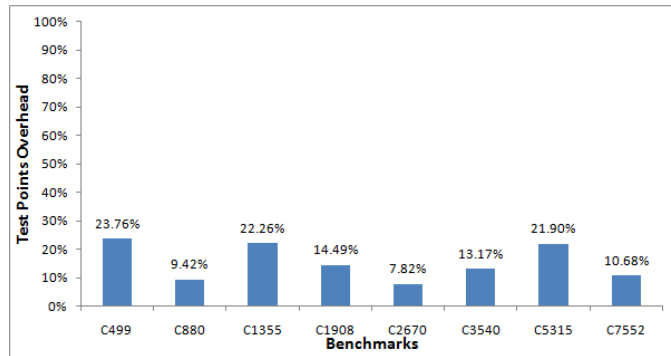


Figure 3.9: Area overhead of the inserted test points.

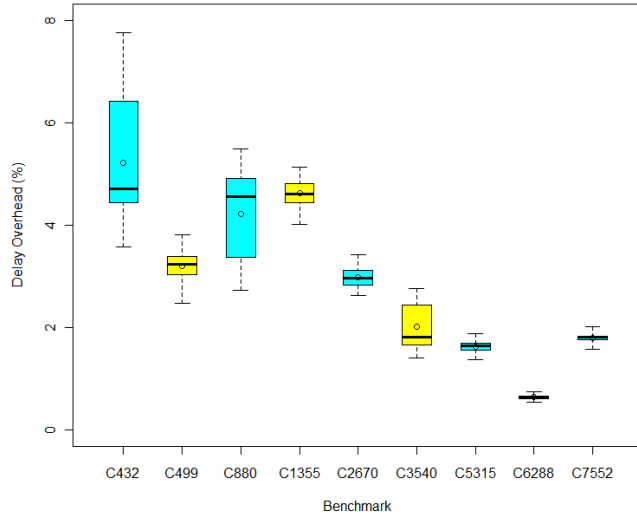


Figure 3.10: Delay overhead under process variation.

3.6 GLC Related Work

In this section, we briefly review directly related GLC research. The existing GLC techniques can be classified into four major groups: (1) direct measurements approaches; (2) schemes that employ FPGA reconfiguration; (3) approaches that create and observe special IC structures and specialized circuitry; and (4) non-destructive techniques that conduct global measurements and deduce scaling factors of each gate by solving a system of equations.

Direct measurement techniques use atomic force microscopes (AFM), electric line measurements (ELM), and optical instruments to directly measure critical dimensions (e.g. effective channel length) [33]. They are very accurate, have a wide range of speeds (e.g. AFM-based techniques are much slower than ELM), and their application is often restricted to the measurements of critical dimensions.

FPGA GLC techniques iteratively create blocks of clock measurement cir-

cuitry and isolate a block under characterization to conduct delay characterization of gates and wires [17].

The third group of techniques populate chips with simple structures such as ring oscillators and delay lines that can be easily characterized in terms of gate delay through clock sweeping and counting techniques [28]. The main limitation of these two types of techniques is that they can be applied only to specific types of designs.

Finally, non-destructive GLC techniques can be divided into two classes. The first group does not impose any assumptions about spatial correlation of gate scaling factors [6][66][107][89][96]. The second class uses spatial correlation, transformations, and techniques such as compressed sensing to reduce cardinality of the system of equations [48].

CHAPTER 4

Hardware Trojan Detection and Diagnosis

A hardware Trojan (HT) [81][43] is a malicious modification to an integrated circuit. The alteration caused by HTs may impact the functionality of the IC, change the original characteristics (e.g. propagation delay or leakage power), or even leak confidential informations from the hardware. Due to the increasing trend of outsourcing, today's IC design and manufacturing has become a global business. However, this results in an increased level of security concerns, as the untrusted foundries have complete access to the hardware during the manufacturing process and may conduct malicious modifications. Therefore, HT detection after manufacturing is essential to ensure the security and integrity of the manufactured ICs.

HT detection is much more challenging than IC testing, because attackers tend to hide the HTs from commonly used detection techniques. For example, attackers may embed a very small HT that is activated only when a rare activation condition is satisfied. Such a HT would compromise the traditional functional testing method, as it is extremely difficult for the test vectors to activate and capture the embedded HTs .

In this chapter, we discuss in details our research findings in hardware Trojan detection and diagnosis using GLC and consistency-based analysis. We begin with the study on challenging hardware Trojan attack models to define the problem domain and motivate our detection and diagnosis techniques. Then, we

discuss our consistency-based HT detection and diagnosis methods.

4.1 Hardware Trojan Attack Model

In order to motivate our HT detection and diagnosis techniques, we first study the challenging hardware Trojan attack models that could possibly be created by an attacker. In particular, we investigate the following three attack models: (1) A one-gate HT attack model that hides in the target circuit by minimizing the impact on leakage energy, switching energy and delay [95][97]; and (2) A customizable HT attack model that generates undetectable attacks by leveraging the finite state machine [98].

4.1.1 One-gate HT Model and Benchmark

4.1.1.1 Overall HT Creation Flow

Our idea in creating challenging HT models is to embed HTs that induce minimum observable variations into the target design. In order to achieve this goal, we employ a one-gate HT trigger that switches the malicious circuitry on and off during the IC operation. In order to increase the difficulty level for detection, the one-gate HT trigger powers on the malicious circuitry only when a rare event occurs, which is defined and activated by the attacker. In this way, the only observable variation before the activation of malicious circuitry is the single HT gate embedded in the circuit. Therefore, to further complicate the detection attempts, the attacker would hide the single HT gate in the circuit and make it difficult to be detected by the commonly used detection methods.

Figure 4.1 shows the overall architecture of the one-gate HT attack model. We embed a single AND gate that serves as a HT trigger. The HT trigger

drives a power control network which either powers on or powers off the malicious circuitry. The malicious circuitry is activated only when the output of the HT trigger is 0; otherwise, the power supply keeps off, which makes the malicious circuitry invisible in terms of leakage or switching power.

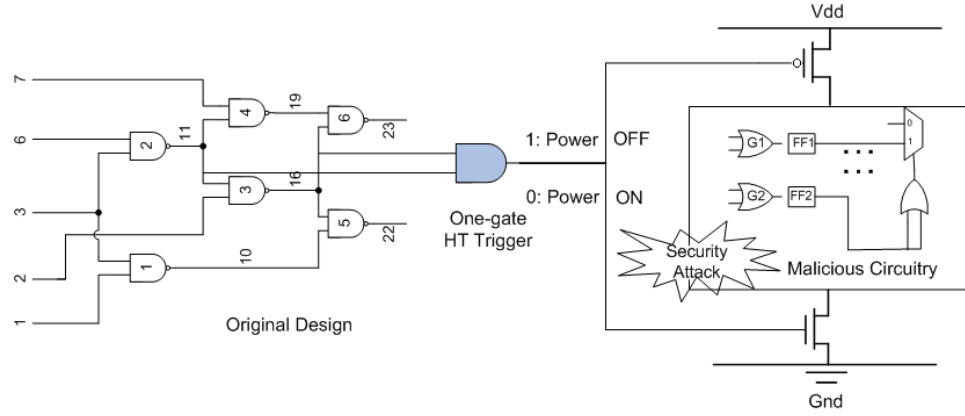


Figure 4.1: Overall architecture of the one-gate HT attack model.

We consider three possible HT creation models that an attacker may consider to minimize the possibility of detection. The proposed HT models correspond to the three most commonly used side-channels for HT detection, namely switching power, leakage power, and delay. Figure 4.2 shows the overall flow of creating the three types of one-gate HTs: (1) For the switching-based HT placement, we design an iterative low switching identification algorithm that searches for the most rarely switching locations in the target design; (2) In the leakage-based HT model, we develop an aging-based leakage power reduction scheme to minimize the observable variations in leakage power; and (3) For the consideration of timing-based HT, we employ a backtracking-based algorithm to identify the reconvergent paths in the circuit, where delay variations caused by the HT is not observable. Then, we combine the three sets of locations and find a number of specific locations that minimize the observability of all three properties. Finally,

we create challenging HT benchmarks by embedding the one-gate HT trigger at one of these locations.

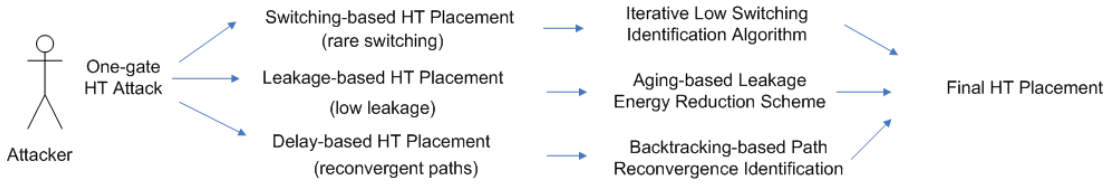


Figure 4.2: Overall flow for HT creation and placement.

4.1.1.2 Rare Switching HT Benchmark

In the switching power-based HT model, the goal is to insert the HT in such a way that it can be switched only by a rare set of input vectors. Consequently, there is a limited probability for the one-gate HT to exhibit any switching activity during the normal IC operation; on the other hand, the attacker can apply the rare input vectors to activate the malicious circuitry at any time. Figure 4.3 shows our simulation results regarding the switching activities of all gates on ISCAS benchmark C499. Our observation in this example is that all gates can be switched by a certain set of input vectors. Also, there exist gates that switch very often (e.g., more than 50% of the time) and, similarly, there are a small set of gates that have relatively low (but non-zero) switching activities (e.g., less than 5% of the time).

We develop a low switching identification algorithm that iteratively searches the locations on the original design and finds those locations that lead to the most rare switching activities. Then, we connect the obtained input signals to a single HT gate that is expected to have low switching activity. Algorithm 4 describes the detailed algorithm for finding such a gate set on the target circuit

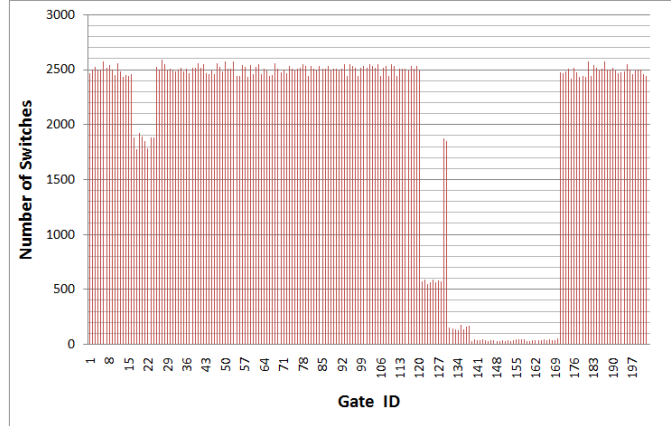


Figure 4.3: Switching activities of all gates on ISCAS benchmark C499, under the application of 5000 pairs of input vectors.

and placing the HT gate.¹ We start with the random simulation results, as shown in Figure 4.3, and find the most rarely switching gate in the design. Next, we iteratively add one more gate from the design to the candidate group. This gate is the most correlated with the existing gates in the group and has the least switching activity. The algorithm terminates after K iterations and provides us with K locations in the circuit that can drive a rarely switching HT trigger.

In order to further reduce the probability of switching and enable an effective evaluation mechanism, we divide the gates in the target circuit into several independent groups, where the gates in different groups have no overlap in terms of their transitive fan-in inputs.

Therefore, one can simulate the switching probability of a gate by varying only the transitive fan-in inputs of its group. Then, if we select only one gate (signal) from each group as the inputs of the HT gate, the overall switching probability

¹*switching*(\cdot) is the function to find out the switching activity of gate g_i via simulation of random input vectors; and *SAT*(\cdot) is the procedure to determine whether the specific gates are switchable via SAT problem solving. The details of the SAT approach is introduced in Chapter 3.3.4.

Algorithm 4 Iterative searching algorithm for placing rare switching HT.

Input: Netlist Net ;**Output:** A set of locations L that result in rare switching one-gate HT;

```
1: Find the most rarely switching gate  $g_0$  via simulation of random input vectors;
2: Insert  $g_0$  into  $L$ ;
3: for  $i \leftarrow 1; i < K; i++$  do
4:   for all Gates  $t$  that are controlled by the transitive fan-in of  $L$  do
5:     if  $switching(g_i) < switching(L) \ \&\& \ SAT(L + g_i)$  is solvable then
6:        $g_i \leftarrow t$ ;
7:     end if
8:   end for
9:   Insert  $g_i$  into  $L$ ;
10: end for
```

of the HT gate can be calculated as the product of all the switching probabilities in each group, since the groups do not share any common primary inputs and thus their switching activities are independent.

We formulate the problem of finding such rare switching locations in the target circuit as a maximum independent set problem. For the first step, we extract an undirected graph G from the netlist of the target circuit as $G = (V, E)$, where V is composed of all the gates in the circuit, i.e., $V = \{g_i | i = 1..n\}$, and E is the set of edges between gates that share at least one transitive fan-in input, i.e., $E = \{e_{ij} | g_i \text{ and } g_j \text{ share at least one transitive fan-in input, } i, j = 1..n\}$. Then, the rare switching locations we are looking for in the circuit are those in the maximum independent set of G , i.e., the maximum number of gates that do not share any transitive fan-in inputs. Maximum independent set is a well known NP-complete problem. Many approximation algorithms have been proposed. For the

discussion of this dissertation, we employ the algorithm proposed by Dharwadker et al. [27] to solve for the rare switching locations.

4.1.1.3 Low Leakage HT Benchmark

The leakage power-based HT model corresponds to the HT detection techniques that leverage whole circuit or gate-level leakage power tracing. In this case, the idea for hiding the one-gate HT is to minimize its leakage power consumption. Therefore, the embedded HT gate would cause a limited variation in leakage power and has a high probability of hiding under the measurement errors in the existing leakage power-based detection approaches.

Our implementation of such a low leakage HT gate is based on the observation that the gate-level leakage power decreases exponentially with the increase in the threshold voltage (following Equation (2.1)), and that the threshold voltage can be increased by IC aging process (following Equation (2.7)). Therefore, our idea is to intentionally age the embedded HT gate in the post-silicon stage to reduce its leakage power to the greatest extent.

We develop a satisfiability (SAT)-based approach to determine the input vectors that can stress the transistors and age the HT gate. Since the output signal of each gate can be expressed as a boolean expression of the input vectors, SAT can determine the input vectors that generate a specific signal pattern. SAT is one of the first known NP-complete problems. Several very high quality SAT solvers are readily available for delivering fast and accurate SAT solutions [29]. We leverage the SAT solutions to find the aging input vectors that stress the HT gate at the expectant location in the design.

One of the consequences of the aging-based low power HT creation is that it may cause a delay degradation, due to the aging of the one-gate HT as well as a set

of other gates in the circuit by applying the selected input vectors. The increased delay may be observable by a timing-based HT detection approach. To address this issue, we compensate for the delay degradation due to aging by employing adaptive body bias (ABB). ABB has been proposed as an effective approach to compensate for the PV impact on performance and power consumption. It provides the ability to manipulate the transistor threshold voltage through the body effect and thus enables either a forward or a reverse body effect to change threshold voltage [22]. Here we use ABB to manipulate the threshold voltage of critical gates (e.g., gates that are on the critical path), so that the variation in the circuit delay can be compensated for.

4.1.1.4 Timing-based HT Benchmark

The delay-based HT model utilizes the limitation in delay measurements that only the delay of one single path is measurable from a specific input to a specific output. Furthermore, as discussed in Chapter 3.3, in the cases where there are multiple parallel paths between an input/output pair, it is difficult to map the delay measurement to one of the paths that are in parallel. Therefore, the HT gate can be well hidden within one of the parallel paths without being discovered by the existing delay-based characterizations.

Based on this thought, we develop a backtracking-based search algorithm to find out all the possible parallel paths in the target circuit for HT insertion (i.e., Algorithm 2 in Chapter 3). In particular, we analyze the structure of the netlist and identify the reconvergence points between each pair of input and output. Here we define reconvergence points as the node in the netlist that is the end point of more than one paths. In the case of reconvergence, none of the paths are measurable in terms of delay, because it is not clear which path is being

measured even though one can measure the end to end delay from a specific input to the reconvergence point. As long as a path is not measurable, it can serve as a difficult case for delay-based HT detection method. The reconvergence identification algorithm converts the netlist of the design to a direct graph. Then, the problem of reconvergence identification converts to a graph theory problem that searches for all the nodes that have an in-degree of at least 2.

4.1.1.5 Summary of HT Benchmarks

The three HT models provide us with a systematic way of evaluating an arbitrary HT placement strategy in terms of the difficulty levels for detection. For example, if a single HT gate is embedded at one of the reconvergent paths, where the leakage power consumption is lower than the measurement resolution and the switching probability is small, it would create an ultra challenging case for the HT detection techniques.

Following this idea, we define the first systematic benchmarking strategy for creating and quantifying the HT attacks with various difficulty levels. The difficulty level of a HT attack model can be evaluated using a triplet $\langle d, l, s \rangle$, where d is a boolean variable indicating whether the HT gate is observable via delay measurement (i.e., whether it is on one of the reconvergent paths); l is a boolean variable representing whether the leakage power of the HT gate is below the resolution of the leakage power characterization, and s is the switching probability of the inserted HT gate at the specific location. We can test and evaluate a HT detection approach using the proposed benchmark, by observing the most difficult level of HT that it can successfully detect.

4.1.2 Customizable HT

4.1.2.1 Customizable HT Overview

The existing HT research [81][88][95][87] targeted only on Trojans that are physically present and thus observable on the target IC, either in the form of additional malicious components or modifications toward the target circuit. Although these types of HTs can be well hidden under the target circuit, the difficulty level for detection is limited due to the following two reasons. (1) the embedded HT would result in at least one type of variation in the observable properties of the IC, including but not limited to physical structures (e.g., layout and wiring) and side channels (e.g., delay and power); and (2) the HTs under consideration are identical on different chips of the same design due to the high cost of customizing the design and manufacturing for HT insertion. As a result, once one chip compromised by HTs is detected during the IC test, all other chips under attack can be easily identified in a straightforward way.

We argue that it is completely feasible to create challenging HTs and bypass the existing detection schemes that are subject to the above limitations. It is rather important to investigate on these HT attacks and motivate security primitives from a completely new angle. Based on these thoughts, we develop a zero-overhead, customizable HT model that an attacker could leverage to create untrusted CAD tools and trigger undetectable security attacks. Our undetectable HTs have the following features:

Zero-overhead. Our HT model leverages the redundant states (called HT states) in the finite state machine (FSM) of the target circuit for security attacks, which does not require any additional hardware to trigger the HT during normal IC operations and, therefore, exposes no observable variations in the IC

properties.

Customizable. The proposed HT model induces different and customizable security attacks on different ICs without introducing additional manufacturing costs. Consequently, even if one instance of the HT is detected, it is extremely difficult for the detection procedures to prove the presence of HTs, nor can they generalize the found instance to other chips under test. We achieve this goal by employing post-silicon device aging caused by the negative bias temperature instability (NBTI) effect [9]. The attacker could intentionally age the target ICs after manufacture in such a way that unpredictable delay faults occur at runtime to transition the IC from normal states to the HT states.

Therefore, the main ramification of our proposed HT model is that it forces the detection mechanisms to move from traditional detection to sequential synthesis. Not only the cost for detection is significantly increased, but also the fundamental paradigms in the existing detection approaches have to be revisited and reconsidered in order to achieve reliable HT detection schemes.

4.1.2.2 Motivational Example of the customizable HT

Figure 4.4 shows a motivational example of our proposed undetectable HT model. Figure 4.4(a) is the finite state machine of a mod-3 up/down counter, including two inputs (x_1, x_0) that control the counter to stop, count up, and count down; and 4 states that can be implemented by 2 flip-flops. Among all 4 states, only 3 of them are valid states of the counter, i.e., representing the count number 0, 1, and 2. The shaded state S_3 is a redundant state (or don't-care state) that cannot be reached from any other states using any inputs. An attacker could leverage S_3 to trigger a variety of attacks, such as leaking confidential information or consuming higher energy.

Figure 4.4(b) demonstrates the design of the sequential circuit based on the FSM, which shows the transition from normal states (i.e., states S_0 , S_1 , and S_2) to the don't-care state (i.e., state S_3), so that the desired HT attack can be triggered at runtime. Our approach is to intentionally age (i.e., stress the corresponding transistors) a certain set of gates and trigger delay faults at the circuit output. For example, the attacker could age gate G_8 so that the signal transmitted to F_2 is delayed. It is possible that the delayed signal for F_2 causes delay fault, e.g., both F_1 and F_2 stay at signal 1, which transitions the circuit into the HT state.

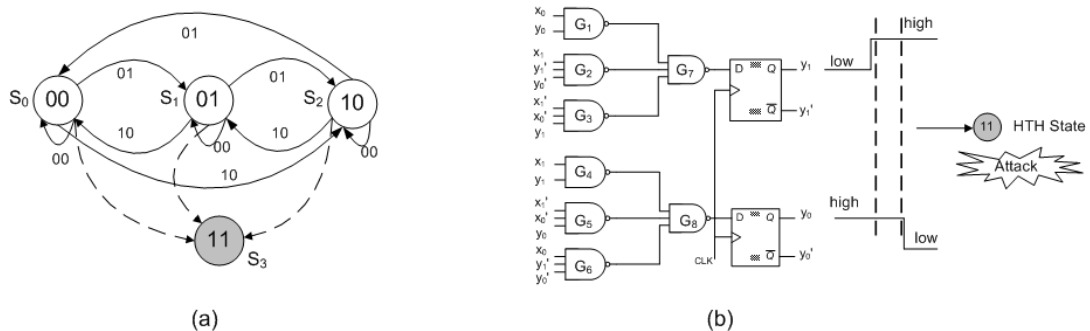


Figure 4.4: Motivational example of the undetectable hardware Trojan horses: (a) finite state machine of a mod-3 up/down counter, which includes 3 normal states (i.e., S_0 , S_1 , and S_2) and 1 redundant state (i.e., S_3); and (b) demonstration of the HT state transition using device aging.

The trigger of delay fault and thus the state transition is fully customizable, in the sense that the attacker can selectively age different components for different chips post-silicon, which transitions the target circuit to different HT states from different normal states. Even in small designs, there are exponentially many combinations of transitions that can be leveraged by the attacker to complicate and obfuscate the attacks.

4.1.2.3 Feasibility Study and Validation for Customizable HTs

The feasibility of the proposed HT attack is based on the assumption that there are large numbers of redundant states available in the target circuit. We argue that the assumption holds for the following two reasons. Firstly, the design of modern sequential ICs often results in large numbers of redundant states for the consideration of performance and ease of integration. Secondly, even if the original design specification does not indicate enough don't-care states, the attacker could easily minimize the FSM [103] to create equivalent designs that include many redundant states.

4.1.2.4 Consequences on HT Detection

As a consequence of the undetectable HT model, the traditional HT detection mechanisms have to be revisited to accommodate the elevated difficulty level for ensuring a trusted IC system. In order to achieve this goal, we argue that the current HT detection approaches [81], which rely on the monitoring of the end system in the post-silicon stage, have to be moved to sequential synthesis at the design time. In other words, the detection process must examine the redundant states generated by the untrusted CAD tools and exclude the possibility of HT attacks early at the design time, which is an extremely difficult task.

Our idea to address the problem is to enforce a specified system at design time, where all or a part of the don't-care states are either explicitly removed or incorporated as a well defined state. In this way, we can limit the freedom of manipulating the FSM that is exposed to the untrusted tools. The downside of this solution is that it may compromise the performance gains obtained from the don't-care states. Therefore, a careful design is required to balance the tradeoff between performance and security of the system.

4.2 HT Variable-based Hardware Trojan Detection and Diagnosis

In this section, we discuss our HT detection and diagnosis methods by introducing an additional HT variable to the GLC process. The HT variable serves as an indicator of the inconsistency caused by the HT trigger.

4.2.1 HT Detection Using HT Variable

Our starting observation is that regardless of the HT type, switching activity, or placement strategy, any gates in the circuit would increase the total leakage energy. However, just observing the leakage energy is not sufficient due to the presence of process variation. The key insight is that the extra HT gates introduce a systematic bias in the total leakage power and, therefore, enable detection of any HT by using systematic leakage power measurements.

Our idea is to introduce an extra component in the power measurement equations that captures the systematic bias caused by HTs. Since we do not have any information about possible HTs, such as their types, locations, or input signals, we abstract all HTs into a single variable called HT variable. In the process of HT detection, we add this HT variable to each of the linear equations regardless of whether or not any HTs exist, which we do not actually know before the HT detection procedure. We keep other parts of the linear equations unchanged. In particular, the equations after adding the HT variable are the following, as modified from Equation (3.2):

$$z + K \cdot s = \tilde{p} + e \tag{4.1}$$

where z is the HT variable, which serves as the indicator of HTs.

As discussed in Chapter 4.1, the most difficult case for HT detection using leakage energy is when only one extra gate is added in the circuit, because it causes the least bias in leakage power and can be best hidden under PV or other sources of errors. Therefore, hereafter we only discuss and demonstrate the case where a single HT gate is used as the trigger.

In order to better illustrate our HT detection technique, we show a HT detection example in Figure 4.5 on ISCAS benchmark circuit C17. We assume that the attacker may have added an extra HT gate in the circuit. We use a NAND gate in this example as the HT gate because NAND gates have the lowest leakage power among all the gates, and thus this is the most difficult case for HT detection. Our linear program to detect the HT includes 8 equations obtained from the application of 8 input vectors. We also add an extra variable z in each linear equation of leakage power measurement. In this example, after solving the system of linear equations using a LP solver, we find that the value for HT variable z is 256.4; this indicates that the target circuit contains malicious circuitry. For the consideration of possible false positives in HT detection, we further test our approach on an instance of C17 circuit without any HTs. In that case, the obtained value for variable z is 0, which indicates that there is no systematic bias in the leakage power compared to its normal value, and thus there are no HTs. The accuracy of our HT detection technique is ensured by our GLC approach using thermal conditioning, as discussed in Chapter 3, which provides accurate characterization results for all the gate-level scaling factors when there is no malicious circuitry.

Furthermore, in order to evaluate the reliability of our HT detection scheme, we repeat our simulation 500 times and plot the probability density function of the HT variable in Figure 4.6. In all 500 runs, the HT variable is 0 when no HTs exist, and it is a large value between 220 and 440 when HTs are present. The

results show a large gap in the values of the HT variable between the two cases. Therefore, the HT variable serves as a reliable indicator for the inserted HTs.

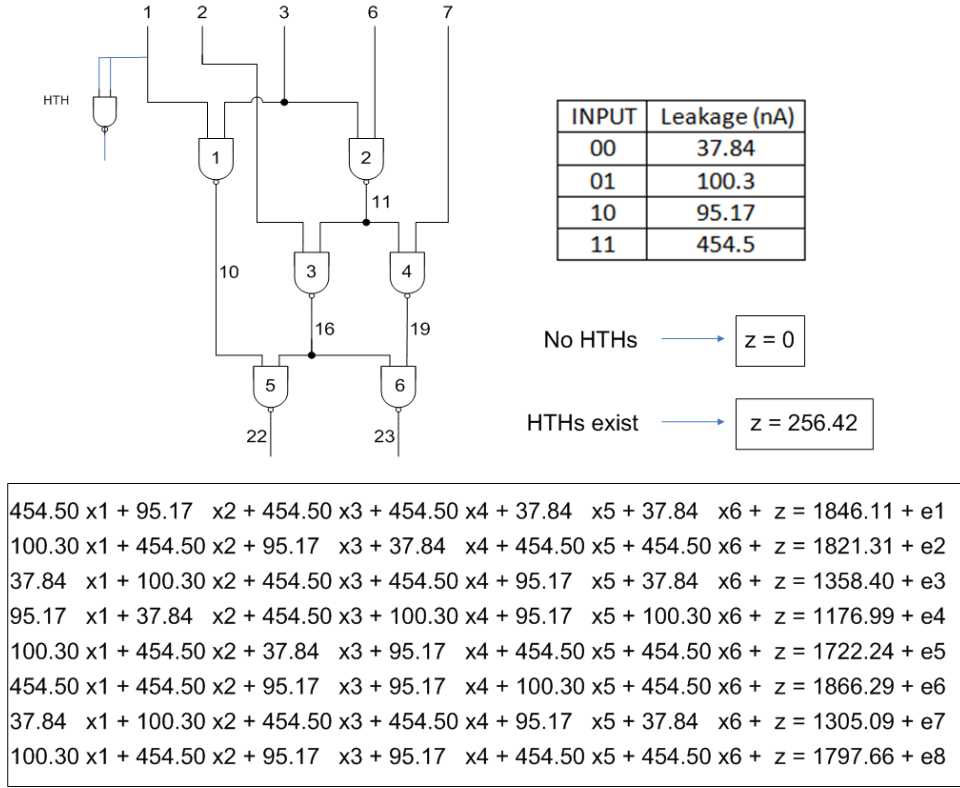


Figure 4.5: Example of GLC-based HT detection scheme on benchmark C17. The coefficients (nominal leakage power values) [104] are shown in the lookup table. We add one extra HT variable z to the system of measurement equations as the indicator of HTs. e_i ($i = 1, 2, \dots, 8$) represents the leakage power measurement errors. The solution of z is zero when no HTs are present, and it is a large value (256.4) in the case where HTs exist.

4.2.2 HT Diagnosis Using HT Variable

HT diagnosis is the process through which we infer the detailed information about the detected HTs, including their types, locations, and input signals. Our generic

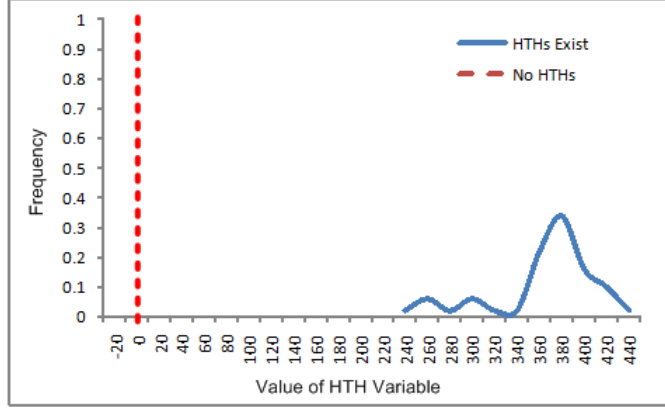


Figure 4.6: Probability density function of HT variable for 500 runs of HT detection on benchmark C17. For all the 500 runs, the value of the HT variable is 0 in the case where there are no HTs, and it is a large value between 220 and 440 when HTs are present.

approach is exhaustive search, in which we first identify the type, location and input signals of the HT, and we verify our identifications by employing additional constraint manipulation based on Equation (4.1), where we add a new HT variable according to our identifications:

$$z + k_{ht} \cdot s_{ht} + K \cdot s = \tilde{p} + e \quad (4.2)$$

where $k_{ht} \cdot s_{ht}$ is the new item we added for HT diagnosis. k_{ht} is the leakage coefficient of the HT gate, which is dependent on the type, location, and input signals of the HT that we have identified. s_{ht} is the variable representing the PV scaling factor of the HT gate. After solving the LP, if the current identification is correct, we will obtain z close to 0, and s_{ht} to be the estimated scaling factor for the HT. Otherwise z is a large value that represents the discrepancy caused by an incorrect identification.

We show the HT diagnosis procedure in Algorithm 5. We examine each

potential location using our generic GLC approach. As soon as we find that the HT value is much lower than those for other cases, we know that we have found the location of the HT. The key observation is that we conduct HT diagnosis on a per segment manner, as discussed in Chapter 3.4. Therefore, even if HTs may have multiple inputs (e.g. NAND gate), the running time is still relatively low because of the small segment size.

Algorithm 5 HT diagnosis algorithm.

Input: 1. Circuit with HT for HT diagnosis;
 2. λ , threshold value of HT variable that indicates the presence of HT

Output: L_{ht} , locations (inputs) of the HT on the circuit

- 1: $L_{ht} = \emptyset$;
- 2: **for** each segment of the circuit **do**
- 3: **for** each input i in the segment **do**
- 4: Assume HT is embedded at input i ;
- 5: Take measurements and formulate a linear program in the form of Equation (4.2) with the HT variable var_{ht} ;
- 6: Solve the LP;
- 7: **if** $var_{ht} < \lambda$ **then**
- 8: add i to L_{ht}
- 9: **end if**
- 10: **end for**
- 11: **end for**
- 12: Return L_{ht} ;

4.3 Consistency-based Hardware Trojan Detection and Diagnosis

Our goal in this section is to address the detection and diagnosis of HTs using consistency analysis based on GLC [94] without employing additional HT variables. Our idea is based on the fact that a circuit containing HTs would cause systematic bias in the total leakage power consumption, no matter where the HT is, how it is constructed, and even whether it is activated or not. With our GLC process, since there are no variables in the system of equations (shown in Equation (3.2)) to represent the HTs, the systematic bias in the total leakage power would create inconsistencies in the equations, and the bias would be reflected in the scaling factors of regular gates in the circuit. By observing the bias in the leakage power scaling factors, we are able to detect HTs embedded in the circuit.

4.3.1 Consistency-based Hardware Trojan Detection

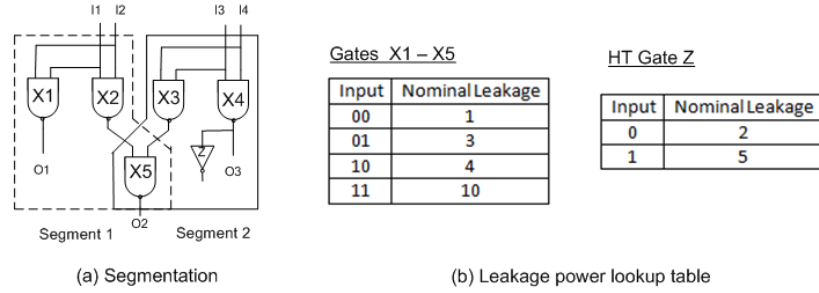
There are two key challenges with the consistency-based HT detection approach. Firstly, we do not assume that we have a golden model of the circuit that does not have any HTs. Therefore, it is difficult to observe the bias in the scaling factors caused by HTs, as there is no standard scaling factors to compare with. Secondly, since the number of gates in modern IC designs is up to the magnitude of millions, the size of the system of equations would easily exceed the computational limit of the LP solvers.

We address both challenges using segmentation. The segmentation of an IC is based on the divide-and-conquer paradigm, in which we divide a large IC into multiple small segments and characterize each of them using GLC. As discussed in Chapter 3.4, segmentation can be implemented using input vector

control, i.e., we freeze a certain set of the primary inputs and vary the others. Consequently, only the gates controlled by the varying inputs would possibly change their coefficients in the system of linear equations, while the other gates would have identical coefficients in all the equations. Therefore, we can represent all the frozen gates using a single variable in the system of linear equations.

Furthermore, there are overlapping gates across segments. This provides us with an opportunity to characterize a single (overlapping) gate in multiple sub-circuits (segments), and thus observe possible bias in scaling factors due to the presence of HT. For example, suppose there are two segments A and B with an overlapping gate X, we can characterize the scaling factors of X in both segment A and B, namely α_a and α_b . Our idea is that α_a and α_b will be consistent if there is no HT present in either segment A or segment B, as ensured by the accuracy of GLC in both segments. In the case where a HT exists in either A or B, there exists inconsistencies in the segment that contains the HT, and the resulting scaling factor (α_a or α_b) will be biased to reflect the inconsistencies. In the case where HTs exist in both segments A and B, since the two segments are different in terms of their gates and overall leakage power, the systematic bias caused by the HTs will be different in the two segments, which will again result in different values for α_a and α_b . We use the average discrepancy (d_{avg}) in calculated scaling factors of overlapping segments as an indicator of whether a HT is present or not. d_{avg} is calculated as the average standard deviation of the scaling factors of the same gate in the overlapping segments.

We illustrate our segmentation-based HT detection scheme using an example shown in Figure 4.7. For the sake of brevity and clarity, the circuit has only five NAND gates (named X1 to X5) as shown in Figure 4.7(a). We adopt normalized values as shown in Figure 4.7(b) for their nominal leakage power. Our goal is to



(a) Segmentation

(b) Leakage power lookup table

Case 1: HT-free vs. HT-free

Input Vectors I1 I2 I3 I4	System of Equations (Segment 1)	Input Vectors I1 I2 I3 I4	System of Equations (Segment 2)
0000	$1 \alpha_1 + 1 \alpha_2 + 10 \alpha_5 = 15.3$	0000	$1 \alpha_3 + 1 \alpha_4 + 10 \alpha_5 = 15.4$
0100	$4 \alpha_1 + 3 \alpha_2 + 10 \alpha_5 = 21$	0001	$4 \alpha_3 + 3 \alpha_4 + 10 \alpha_5 = 21.35$
1000	$3 \alpha_1 + 4 \alpha_2 + 10 \alpha_5 = 21.1$	0010	$3 \alpha_3 + 4 \alpha_4 + 10 \alpha_5 = 21.45$
1100	$10 \alpha_1 + 10 \alpha_2 + 3 \alpha_5 = 26.9$	0011	$10 \alpha_3 + 10 \alpha_4 + 4 \alpha_5 = 29.2$
Results	$\alpha_1 = 1.1; \alpha_2 = 1.2; \alpha_5 = 1.3$	Results	$\alpha_3 = 1.15; \alpha_4 = 1.25; \alpha_5 = 1.3$

Case 2: HT-free vs. HT-present

Input Vectors I1 I2 I3 I4	System of Equations (Segment 1)	Input Vectors I1 I2 I3 I4	System of Equations (Segment 2)
0000	$1 \alpha_1 + 1 \alpha_2 + 10 \alpha_5 = 15.3$	0000	$1 \alpha_3 + 1 \alpha_4 + 10 \alpha_5 = 19.4$
0100	$4 \alpha_1 + 3 \alpha_2 + 10 \alpha_5 = 21$	0001	$4 \alpha_3 + 3 \alpha_4 + 10 \alpha_5 = 25.35$
1000	$3 \alpha_1 + 4 \alpha_2 + 10 \alpha_5 = 21.1$	0010	$3 \alpha_3 + 4 \alpha_4 + 10 \alpha_5 = 25.45$
1100	$10 \alpha_1 + 10 \alpha_2 + 3 \alpha_5 = 26.9$	0011	$10 \alpha_3 + 10 \alpha_4 + 4 \alpha_5 = 30.8$
Results	$\alpha_1 = 1.1; \alpha_2 = 1.2; \alpha_5 = 1.3$	Results	$\alpha_3 = 1.15; \alpha_4 = 1.25; \alpha_5 = 1.7$

Case 3: HT-present vs. HT-present

Input Vectors I1 I2 I3 I4	System of Equations (Segment 1)	Input Vectors I1 I2 I3 I4	System of Equations (Segment 2)
0000	$1 \alpha_1 + 1 \alpha_2 + 10 \alpha_5 = 18.8$	0000	$1 \alpha_3 + 1 \alpha_4 + 10 \alpha_5 = 19.4$
0100	$4 \alpha_1 + 3 \alpha_2 + 10 \alpha_5 = 24.5$	0001	$4 \alpha_3 + 3 \alpha_4 + 10 \alpha_5 = 25.35$
1000	$3 \alpha_1 + 4 \alpha_2 + 10 \alpha_5 = 24.6$	0010	$3 \alpha_3 + 4 \alpha_4 + 10 \alpha_5 = 25.45$
1100	$10 \alpha_1 + 10 \alpha_2 + 3 \alpha_5 = 28.3$	0011	$10 \alpha_3 + 10 \alpha_4 + 4 \alpha_5 = 30.8$
Results	$\alpha_1 = 1.12; \alpha_2 = 1.21; \alpha_5 = 1.64$	Results	$\alpha_3 = 1.15; \alpha_4 = 1.25; \alpha_5 = 1.7$

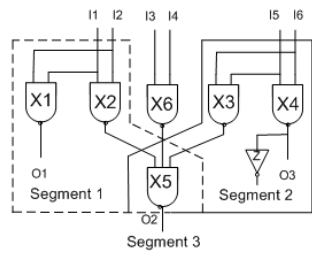
(c) Formulation of systems of linear equations for HT detection

Figure 4.7: Example of the segmentation-based HT detection approach: (a) shows that a circuit with five gates is segmented into two segments, and gate X5 is the overlapping gate of the two segments; (b) shows the nominal leakage power values for all the gates in the circuit; and (c) demonstrates the formulation of systems of linear equations and their solutions in three cases regarding whether a HT is present in each segment. The discrepancy in the results of the overlapping gate (i.e., X5) is an indicator of whether any HT exists or not.

determine whether there is any HT embedded in the circuit. We first partition the circuit into two segments, as shown in Figure 4.7(a). We obtain Segment 1 (gates $X1$, $X2$, and $X5$) by freezing inputs 3 and 4 and by applying different input vectors to inputs 1 and 2. Similarly, we obtain Segment 2 (gates $X3$, $X4$, and $X5$) by freezing inputs 1 and 2.

Next, we conduct GLC for each individual segment. In particular, we apply four input vectors to each segment that provide four sets of nominal leakage values for gates $X1$, $X2$, and $X5$ in Segment 1 and gates $X3$, $X4$, and $X5$ in Segment 2. For HT detection, we show three cases where HT exists or does not exist in Segment 1 and Segment 2. We assume that we do not know whether the circuit has HT in advance, and we form the system of linear equations to conduct GLC for each segment as shown in Figure 4.7(c).

In case 1 (where HT is present in neither Segment 1 nor Segment 2), the values of overlapping gate $X5$ in the two segments are identical. In case 2 (where a single HT gate is present in Segment 1), the two calculated values of $X5$ have a 30.8% discrepancy. Finally, in case 3 (where HT is present in both Segment 1 and Segment 2), the values of $X5$ have a 3.7% discrepancy. These results indicate that the discrepancy between overlapping gates in multiple segments can serve as an indicator for the systematic bias in leakage power caused by embedded HTs. Therefore, we check the GLC results of overlapping gates between pairs of segments in the circuit. As long as the segments can cover all the gates in the circuit, our approach can detect any HTs embedded in the circuit. Furthermore, the use of segmentation ensures the scalability of GLC, since the number of gates being characterized in each system of linear equations is drastically reduced.



Segment 1:

Input Vectors I1 I2 I3 I4 I5 I6	System of Equations
000000	$1 \alpha_1 + 1 \alpha_2 + 10 \alpha_5 = 15.3$
010000	$4 \alpha_1 + 3 \alpha_2 + 10 \alpha_5 = 21$
100000	$3 \alpha_1 + 4 \alpha_2 + 10 \alpha_5 = 21.1$
110000	$10 \alpha_1 + 10 \alpha_2 + 3 \alpha_5 = 26.9$
Results 1:	$\alpha_1 = 1.1; \alpha_2 = 1.2; \alpha_5 = 1.3$

Segment 2:

Input Vectors I1 I2 I3 I4 I5 I6	System of Equations
000000	$1 \alpha_3 + 1 \alpha_4 + 10 \alpha_5 = 19.4$
000001	$4 \alpha_3 + 3 \alpha_4 + 10 \alpha_5 = 25.35$
000010	$3 \alpha_3 + 4 \alpha_4 + 10 \alpha_5 = 25.45$
000011	$10 \alpha_3 + 10 \alpha_4 + 4 \alpha_5 = 30.8$
Results 2:	$\alpha_1 = 1.15; \alpha_2 = 1.25; \alpha_5 = 1.7$

Segment 3:

Input Vectors I1 I2 I3 I4 I5 I6	System of Equations
000000	$1 \alpha_6 + 10 \alpha_5 = 14.2$
000100	$3 \alpha_6 + 10 \alpha_5 = 16.6$
001000	$4 \alpha_6 + 10 \alpha_5 = 17.8$
001100	$10 \alpha_6 + 3 \alpha_5 = 25.9$
Results 3:	$\alpha_6 = 1.2; \alpha_5 = 1.3$

Results 1 + Results 3: {X1, X2, X5, X6} **HT-Free**

Results 1 + Results 2: {X1, X2, X3, X4, X5} possibly **HT-Present**

Results 2 + Results 3: {X3, X4, X5, X6} possibly **HT-Present**



Conclusion: {X3, X4} **HT-Present**

Figure 4.8: Example of consistency-based HT diagnosis. We demonstrate the gate characterization in three segments with overlapping gates. The consistency in Segment 1 and Segment 3 exposes the possible HTs in Segment 2.

4.3.2 Consistency-based Hardware Trojan Diagnosis

The goal in HT diagnosis is to determine the locations of the HTs in the circuit if any exist, so that one can either remove or mask the HTs from the circuit. We design a scalable HT diagnosis scheme based on our consistency-based HT detection method. We have observed that one can detect the existence of HTs using two segments with overlapping gates. However, the HT detection results do not indicate which segment the HTs may be embedded in, and thus it is difficult for the HT masking process to handle the HTs. In order to diagnose the HTs, we introduce a third segment with the same set or subset of overlapping gates and use it as an arbiter for HT diagnosis. Figure 4.8 shows an example of the consistency-based HT diagnosis. We find one more segment (Segment 3) compared to the example in Figure 4.7. The three segments have an overlapping gate $X5$. We vary the controlling inputs of each segment and characterize the scaling factor of all the gates. In the case where the HT is embedded in Segment 2, we have the scaling factor of $X5$ consistent in Segment 1 and Segment 3 (e.g., $\alpha_5 = 1.3$), while that in Segment 2 has a different value (e.g., $\alpha_5 = 1.7$). Then, we analyze each combination of the pair of segments following the rule that an inconsistency in the scaling factor of the overlapping gate indicates possible HTs in either of the segments, while a consistent result ensures that both of the involved segments are HT-free. For example, as shown in Figure 4.8, we conclude that the HTs are present in Segment 2 (i.e., gates $X3$ and $X4$).

Algorithm 6 describes the detailed procedure of the consistency-based HT diagnosis. In each round of the diagnosis, we first characterize three segments with at least one overlapping gate. Then, we compare the scaling factor values of the overlapping gate obtained from the three segments. The one that has a large difference compared to the other two values is in the segment that is possibly

HT-present. In the case where all three scaling factor values have large difference compared to the others, we conclude that multiple HTs are embedded in at least two segments and find more segments that cover the overlapping gates to further diagnose the HTs.

Algorithm 6 Consistency-based HT diagnosis.

Input: Target circuit with embedded HTs;

Output: Segment set Seg , which contains all the segments that are HT-present;

- 1: Detect the existence of HTs;
- 2: Search for S , the three-segment set that covers all the gates in the circuit;
- 3: **for** each S_i in S **do**
- 4: **for** $j = 1 \rightarrow 3$ **do**
- 5: Characterize Segment S_{ij} and obtain scaling factor α_j for the overlapping gate;
- 6: **end for**
- 7: $d_1 = \min\{|\alpha_1 - \alpha_2|, |\alpha_1 - \alpha_3|\}$;
- 8: $d_2 = \min\{|\alpha_2 - \alpha_1|, |\alpha_2 - \alpha_3|\}$;
- 9: $d_3 = \min\{|\alpha_3 - \alpha_1|, |\alpha_3 - \alpha_2|\}$;
- 10: $h = \operatorname{argmax}\{d_1, d_2, d_3\}$;
- 11: Insert S_{ih} into Seg ;
- 12: **end for**
- 13: **return** Seg ;

Furthermore, within the segment where we have confirmed that a HT exists, we employ a variable elimination technique to determine the location of the HT at the gate level. Our intuition is that there must be a gate that drives the HT in the segment, and more importantly, the switching pattern of the HT gate is correlated with the normal gate that drives it. Therefore, they often have linearly

dependent coefficients in the system of linear equations. In this case, if we conduct linear transformation and eliminate the driving gate of the HT, the HT gate will be eliminated as well due to the linear dependency in the coefficients. On the other hand, if we conduct this variable elimination procedure for each individual gate in the segment, and evaluate whether the segment is HT-free after each round of elimination, we are able to conclude which gate is driving the HT.

4.3.3 Self-consistency Analysis via Optimal Subsegments Creation

4.3.3.1 Motivation of Self-consistency Analysis

From the discussions in Chapters 4.3.1 and 4.3.2, we note that the major source of cost in HT detection and diagnosis is the leakage power measurements required in each of the overlapping segments. Despite of the non-instrumentation to the target circuit and the high accuracy, the power measuring devices or techniques [74][73][67] would introduce additional delay or hardware cost to the process.

Therefore, our goal in achieving a highly efficient HT detection and diagnosis scheme is to minimize the number of power measurements that are required to obtain accurate detection and diagnosis results. In order to achieve this goal, we regard the creation of multiple overlapping segments as the key issue in both HT detection and diagnosis processes, since the strategy of segmentation directly impacts the number of measurements that are required to solve the systems of linear equations. In the next subsections, we formulate the problem of minimizing the power measuring cost and propose our solution that delivers an efficient segmentation strategy.

4.3.3.2 Problem Formulation for Segment Creation

The problem of creating multiple overlapping segments that minimize the number of required measurements can be formulated as the following:

Given a netlist of circuit C , find a set of segments Seg such that Seg covers all gates in C and that the total number of power measurements is minimized in order to solve the system of linear equations for each segment s in Seg . In particular, the total number of required equations (measurements) can be formulated as the following:

$$N = \sum_{s \in Seg} (q_s - \sum_{i=1}^{q_s} r_i) \quad (4.3)$$

where q_s is the number of measurements that are required to solve the system of linear equations for segment s and, therefore, $q_s \geq n_s$ (n_s denotes the number of gates in segment s); r_i is the number of times measurement i is reused by other segments except segment s . r_s is the subset of measurements in q_s that are reused in at least one other segment.

From Equation (4.3) we conclude that we cannot reduce N by simply reducing the number of segments or the number of gates in each segment (i.e., n_s), because all of the gates in the circuit have to be covered by the segment set Seg . Thus, our idea for reducing N is to reuse the global power measurements in various segments, i.e., increasing r_s to the greatest extent. Furthermore, we note that the best way to reuse the measurements is by creating a set of sub-segments from the same larger segment. Consequently, the power measurements of the larger segments can be used in each of the sub-segments since the larger segment is a superset of the corresponding sub-segments.

4.3.3.3 Variable Elimination

In the extreme (ideal) case of sub-segment creation, we only measure a small set of leakage power values of the entire circuit, and generate all the sub-segments that are required for HT checking solely from this set of measurements. In this way, the power measurements are reused in the greatest extent. Also, it reduces the complexity of conducting power measurements, since only global measurements are required.

The remaining issue in reusing the power measurements is how we can accommodate the measurements of the larger segment to those of the sub-segments. We address this issue by leveraging a variable elimination technique in the system of linear equations. In particular, we only conduct one set of leakage power measurements toward the entire circuit. Then, we apply linear transformation to the obtained equations so that a specific set of variables can be eliminated from the equations, leaving only the rest of the variables appearing in the equation. Note that this transformation is equivalent to the process of creating a sub-segment from the global segment, which does not require additional leakage power measurements. In this way, the global power measurements can be reused in various sub-segments and thus that the total number of measurements is reduced.

Figure 4.9 shows a motivating example regarding the variable elimination technique to create sub-segments from a global segment. In order to compare our technique with the initial non-optimized approach, we use the same example as Figure 4.7 in terms of HT detection. As shown in the example, we start with a system of equations that include all the gates in the circuit and that are obtained from a fixed set of global power measurements of the entire circuit. Then, we extract two separate sets of equations from the global measurements by conducting linear transformations and eliminating the unneeded variables.

The two smaller systems of equations cover the gates in the two sub-segments, which are the same as the segments in Figure 4.7. However, the advantage in the linear transformation-based segmentation method is that it requires less power measurements in order to characterize all the gates in the segments.

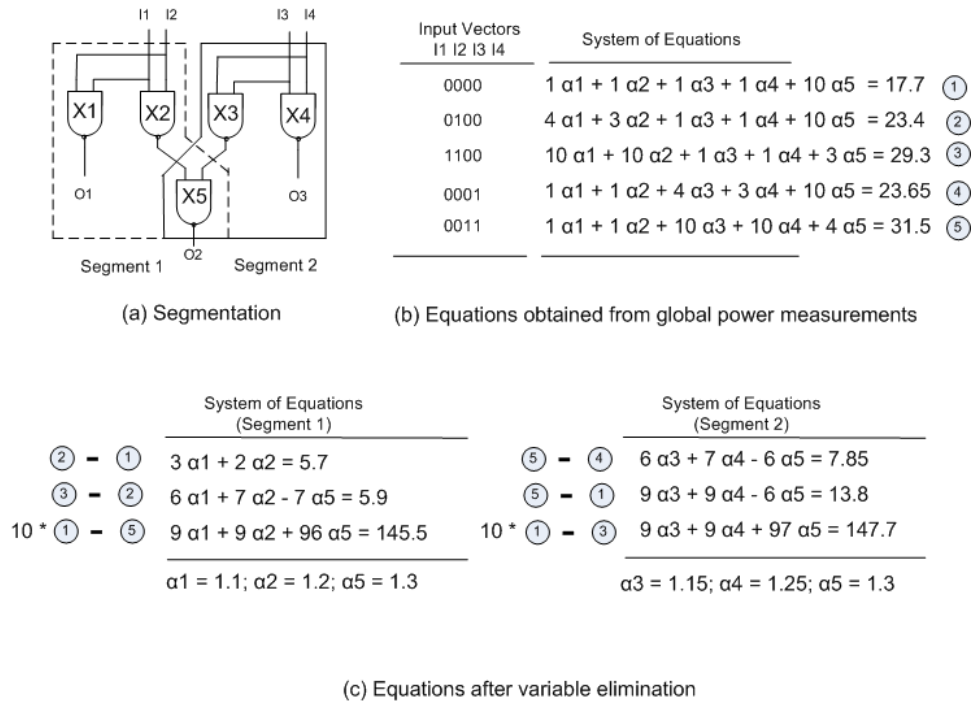


Figure 4.9: Example of the variable elimination technique using linear transformation.

It is important to note that the resulting number of measurements with variable elimination is less than the original method and, meanwhile, the two methods provide the same HT detection results. Furthermore, in large ICs where the partitioned segment is still large and beyond the processing ability of GLC, we employ a variable grouping technique, in which we simultaneously consider two or more variables (gates) during both HT detection and diagnosis. In particular, we can group two or more variables into a single new variable, in the case that these variables have in all equations exactly the same coefficients. This situation

is rather common and its effectiveness can be further enhanced by intentional selection of a subset of equations that satisfy this requirement. From the theoretical point of view, this technique implies that one does not have to characterize each individual gate for accurate HT detection, which ensures the scalability of the approach.

4.3.3.4 Gate Cover Problem

Another important issue in HT detection and diagnosis is that it must cover all the locations in the circuit in terms of searching hardware Trojans. Therefore, the selected sub-segments must cover all the gates in the circuit. Meanwhile, in order to reduce the computational complexity in HT detection and diagnosis, we aim to minimize the number of sub-segments that we select for consistency checking, under the condition that the sizes of the segments are well controlled so that the resulting systems of linear equations are solvable using common linear programming solvers. Taking into consideration the above requirements and goals, we formulate the segment selection problem as a set cover problem:

Segment Selection Problem. Given (1) a netlist of circuit C that contains a set of gates $G = \{g_1, g_2, \dots, g_m\}$ and (2) k sub-segments obtained from the variable elimination process, identify the smallest number of sub-segments whose union contains all gates (i.e., g_1, g_2, \dots, g_m) in G .

We solve the set cover problem using integer linear programming and the approximation algorithm discussed in [83], which provides us with the smallest number of sub-segments to minimize the computation complexity.

4.4 Hardware Trojan Evaluation Results

In this section, we present our evaluation results for our research findings in hardware Trojans, including hardware Trojan attack models, HT variable-based detection and diagnosis, and consistency-based detection and diagnosis.

4.4.1 Effectiveness of Hardware Trojan Attack Models

We evaluate our HT benchmark creation method on a set of ISCAS and ITC benchmarks. For each benchmark circuit, we first embed a single HT at the location determined by the HT attack model discussed in Chapter 4.1. Then, we evaluate the leakage power, switching power of the HT gate, as well as its observability under delay measurements. The combination of the three metrics quantifies the difficulty level of detecting such a HT attack.

4.4.1.1 Low Leakage-based HT

Table 4.1 and Table 4.2 shows the trend of total leakage energy reductions by varying the V_{th} increase during the aging process from 10% to 100%. We observe that the leakage energy can be reduced by up to 28X, which enables the placement of the ultra-low leakage HTs on all circuit locations. Furthermore, we observe that after the delay compensation of the non-HT gates is done using adaptive body biasing, the leakage energy reduction can still be up to 18X. The results indicate that we are able to place the low leakage HT gate without impacting the delay characteristics of the design, which makes the HT difficult to detect using both delay and leakage power-based characterizations. Furthermore, for the larger designs such as C7552 (shown in Table 4.2), we obtain a larger rate of leakage energy reduction.

Table 4.1: Leakage energy reduction via aging for HT benchmark creation (Benchmark C6288).

V_{th} Increase	Without Delay Compensation	With Delay Compensation
10%	2.0	1.9
20%	3.7	3.4
30%	6.3	5.6
40%	9.7	8.2
50%	13.4	10.9
60%	17.0	13.2
70%	20.2	15.0
80%	23.1	16.4
90%	25.8	17.5
100%	28.3	18.5

Table 4.2: Leakage energy reduction via aging for HT benchmark creation (Benchmark C7552).

V_{th} Increase	Without Delay Compensation	With Delay Compensation
10%	2.2	2.0
20%	4.3	4.0
30%	8.7	7.7
40%	16.9	14.4
50%	31.3	25.9
60%	55.2	44.0
70%	91.0	69.9
80%	138.9	102.6
90%	195.9	138.7
100%	256.8	173.7

4.4.1.2 Rare Switching-based HT

Figure 4.10 shows our simulation results for rare switching-based HT creation. The box plots show the statistical distributions of the switching activities for all gates in the ISCAS'85 benchmark circuits, obtained from the simulation of 10,000 randomly generated input vectors for each design. For each box in the plot, the lower and upper edges correspond to the 25th and 75th percentiles of the distribution. The line in the middle of each box indicates the median of the distribution. The smallest and largest points are also shown if they happen outside a range from the box. In most cases the switching probability ranges from 20% to 50%, and it is very rare to have gates that can never be switched.

However, after applying our iterative low-switching identification algorithm and feeding the obtained input pins to a single AND gate, we obtain a maximum of 0.78% switching probability while simulating 10,000 input vectors. Also, we have used SAT to show that for each AND gate, there is at least one input vector that could activate the malicious circuitry. Therefore, our results indicate that the attacker can use the rare activation condition to trigger the malicious circuitry during the system operation, while the single HT gate with low switching probability is difficult to detect when the malicious circuitry is dormant.

Table 4.3 shows the switching probability results obtained by applying the maximum independent set algorithm. We evaluated two cases, where the single HT gate is a NAND gate with 5 inputs and 10 inputs. We observe that the probability of switching is 10^1 to 10^4 times lower than the results in Figure 4.10.

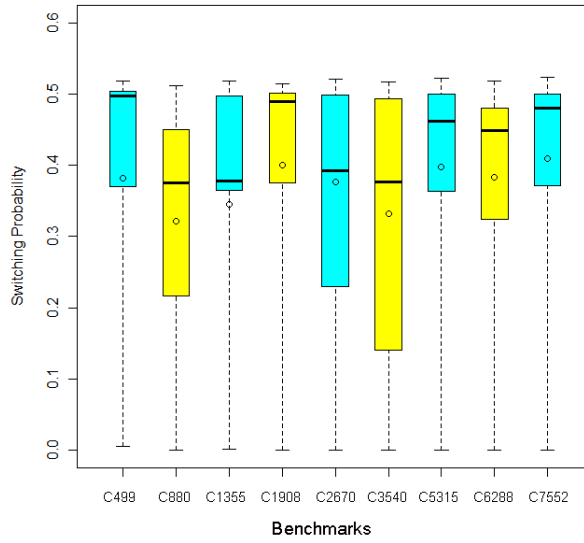


Figure 4.10: Simulation results of switching activities of all gates on ISCAS'85 benchmarks.

Table 4.3: Switching probability of the HT gate using the maximum independent set approach.

Benchmark	# Gates	Prob. Switching (5-input)	Prob. Switching (10-input)
C432	160	4.77E-04	2.63E-09
C499	202	2.15E-05	1.59E-08
C880	383	6.83E-04	8.92E-08
C1355	546	4.04E-08	1.98E-09
C1908	880	7.24E-06	3.02E-09
C3540	1669	2.60E-05	6.78E-10
C5315	2307	5.48E-06	2.05E-10
C7552	3512	5.24E-06	4.13E-10

4.4.1.3 Timing-based HT

Table 4.4 summarizes our simulation results regarding delay characterizable gates on a set of ISCAS benchmarks. As discussed in Chapter 4.1, we cannot characterize the delay of a path if there exist parallel reconvergent path from the input to the output. From the simulation results, we observe that there is no full coverage of all gates in any of the evaluated benchmarks in terms of delay characterization. The highest achieved rate of coverage on the benchmark set is 60%, which still leaves a large portion of the circuit susceptible to HT placement without the risk of being detected.

Table 4.4: Simulation results regarding uncharacterizable gates due to reconvergences. The high percentage of uncharacterizable gates in each design indicates that there is a large number of candidate locations for embedding the non-detectable one-gate HT trigger.

Benchmark	Gates	# Inputs	# Outputs	# Gates with Reconvergence	% Gates with Reconvergence
C499	202	41	32	80	39.6%
C880	383	60	26	208	53.5%
C1355	546	41	32	546	100%
C1908	880	33	25	739	84.0%
C2670	1193	233	140	931	78.0%
C3540	1669	50	22	1542	92.4%
C5315	2307	178	123	1924	83.4%
C7552	3512	207	108	2552	72.7%

4.4.2 HT variable-based HT Detection and Diagnosis Results

We evaluate our HT detection approach on a set of ISCAS and ITC benchmarks. For each benchmark, we simulate two cases: (1) where the HTs do not exist, and (2) where a single HT gate (NAND gate) is embedded at random locations on the target circuit. We repeat the leakage power measurements for all the benchmarks 50 times. The results are shown in Table 4.5. Also, we plot the probability density function (PDF) of the HT variable in Figure 4.11. We observe a large gap between the two cases in terms of the probability distribution of the HT variable. This enables us to draw a decision line between the two situations, with which we achieve zero false positives and zero false negatives in HT detection.

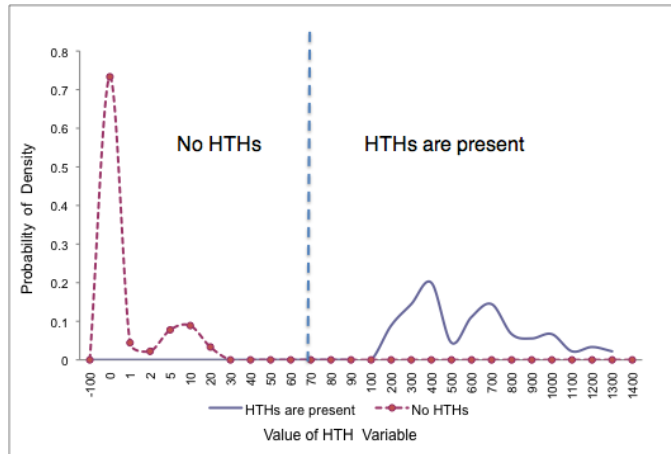


Figure 4.11: PDF of the HT variable in HT detection, integrated with all the ISCAS and ITC benchmarks in Table 4.5. In the case when no HTs exist, the HT variable has a small value from 0 to 11.2. When a single HT gate is present, the HT variable ranges from 151 to 1214. There is a large enough gap between the two cases to enable us to draw a decision line at around 70 to distinguish the two cases.

Table 4.5: HT Detection and Diagnosis on ISCAS and ITC Benchmarks

Design	Gates	GLC Error (%)	With HTs	No HTs
C17	6	0.0057	240 ~ 303	0
C432	160	0.11	582 ~ 603	0 ~ 8.9
C499	202	0.26	174 ~ 254	0 ~ 6.3
C880	383	0.34	151 ~ 231	0 ~ 10.2
C1355	546	0.40	298 ~ 666	0 ~ 4.5
C1908	880	0.98	600 ~ 1022	0 ~ 2.5
C2670	1193	0.75	567 ~ 1182	0 ~ 1.9
C3540	1669	1.72	232 ~ 881	0 ~ 0.2
C5315	2307	0.52	223 ~ 1214	0
C6288	2416	0.13	342 ~ 912	0 ~ 0.4
C7552	3512	0.39	492 ~ 838	0 ~ 1.3
S526	214	0.33	195 ~ 315	0 ~ 11.2
S832	292	0.73	214 ~ 355	0 ~ 10.8
S38584	19253	0.20	381 ~ 1198	0 ~ 6.6
b17	32192	0.60	586 ~ 1210	5.8 ~ 9.7

4.4.3 Consistency-based HT Detection and Diagnosis Results

4.4.3.1 Consistency-based HT Detection

We evaluate the consistency and self consistency-based HT detection methods on a set of ISCAS and ITC benchmarks. We generate the IC instances following the process variation models proposed by Asenov et al. [7] and Cline et al. [24]. In particular, we consider that the threshold voltage follows a Gaussian distribution (e.g., mean is 0.25V, and standard deviation is 0.01V). Also, we assume that the effective channel length follows the quad-tree model [24] that reflects the spatial correlation. For each benchmark, we simulate two cases where HTs are present (i.e., HT-present) and there are no HTs in the circuit (i.e., HT-free). The threat model we consider is the additional gate attack [95], where the attacker embeds one or more small sized gate (e.g. an inverter) into the circuit. The metric we use for identifying HTs is the inconsistency value, i.e., average discrepancy (d_{avg}) of the scaling factors, which is calculated as the average standard deviation of the scaling factors of the same gate in the overlapping segments. We select pairs of segments that have overlapping gates and can cover all the gates in the circuit, conduct GLC of each of the segment, and calculate the d_{avg} value over all pairs.

Table 4.6 shows the inconsistency values obtained from the consistency-based and self-consistency-based HT detection approaches. We observe that there are large gaps (more than 7X) in terms of d_{avg} between the HT-free case and the HT-present case. This enables us to draw a decision line between the d_{avg} values in the two cases and use it to determine whether HTs exist or not.

To be more specific, we plot in Figure 4.12 the distribution of the inconsistency values over all pairs of segments in four of the benchmarks. We observe that there are no overlaps between the inconsistency values in the two cases, which

Table 4.6: HT detection results using consistency and self consistency-based GLC: the values in the “HT-Free” and “HT-Present” columns represent the average discrepancy of the overlapping gates in terms of their scaling factors.

Benchmark	# Gates	HT-Free (Cons.)	HT-Present (Cons.)	HT-Free (Self-Cons.)	HT-Present (Self-Cons.)
C499	202	6.2E-03	2.0E-01	4.1E-03	3.0E-01
C880	383	5.8E-03	7.3E-02	5.7E-02	4.0E-01
C1908	880	2.1E-03	2.3E-01	3.8E-03	2.3E-01
C2670	1193	1.4E-03	1.3E-01	8.1E-04	4.3E-01
C5315	2307	6.2E-03	1.2E-01	5.9E-02	8.5E-01
C7552	3512	4.6E-03	9.8E-02	1.2E-02	5.8E-01
S38584	19253	4.7E-03	2.4E-01	2.3E-03	3.6E-01
b17	32192	5.9E-03	3.8E-01	1.6E-03	6.4E-01

ensures zero false positives and false negatives in the self-consistency based HT detection. In particular, we can determine a decision line (e.g., at the 50% boundary of the average gap) based on the results obtained from a small set of training benchmarks and use it to detect HTs in an arbitrary circuit after manufacturing.

In Figure 4.13 we summarize and compare the measurement costs of the two methods that we have developed, namely the segment selection using the set cover algorithm and the variable grouping technique. We observe that the ratio of the number of measurements and gates scales with the sizes of the circuit using our variable elimination technique.

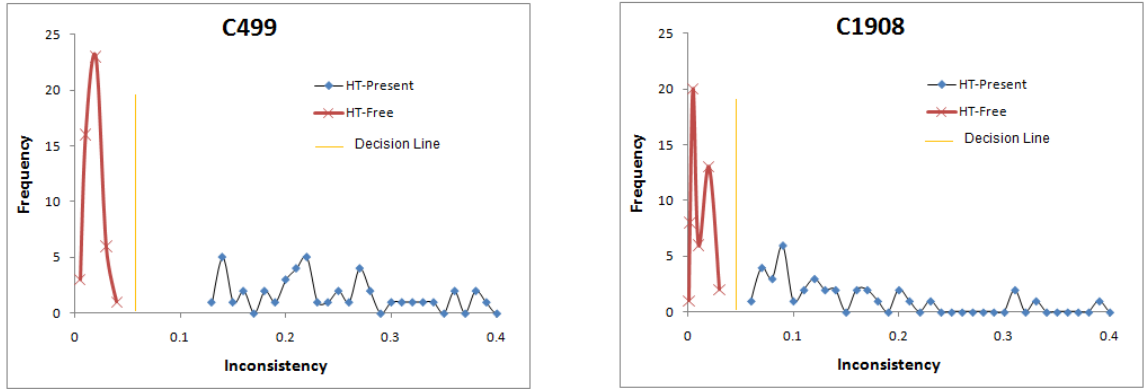


Figure 4.12: Distribution of the inconsistency values in the HT-present and HT-free cases.

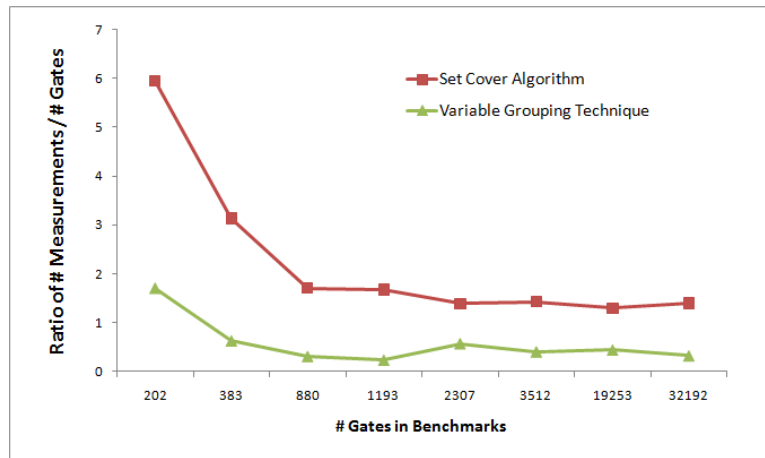


Figure 4.13: Comparison of the numbers of measurements in the consistency-based hardware Trojan detection.

4.4.3.2 Consistency-based HT Diagnosis

We evaluate the consistency-based HT diagnosis approach on a set of ISCAS benchmarks, as shown in Figure 4.14. For each benchmark, we show the scaling factors of the overlapping gates in three segments, where a single HT is embedded in one of the segments (e.g., Segment 3). We observe from the results that the two values of scaling factors from the HT-free segments are consistent with each other, and that in the HT-present segment is either a very high value or a very low value apart from the two consistent values. These results enable us to conclude that the HT is embedded in Segment 3 with zero false positives and zero false negatives.

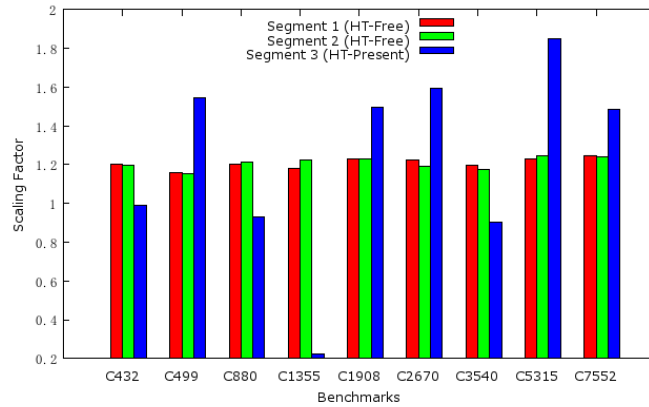


Figure 4.14: Simulation results for the consistency-based HT diagnosis.

4.5 Hardware Trojan Related Work

Agrawal et al. [1] proposed one of the first HT detection techniques in 2007. They construct fingerprints using side channels (e.g., power and temperature) of the circuit for a specific design and authenticate the IC instances based on the fingerprints. The technique is based on the assumptions that there is no process

variation, ICs are available for reverse engineering, and there are no measurement errors in the side channels.

Several early HT detection approaches employed functional test techniques. Functional tests simulate the input vectors on the circuit and monitor the outputs to see whether they match the expected patterns. For example, Wolff et al. [100] proposed the generation of test vectors that maximize the likelihood of detecting rarely switching HT gates. Also, Banga et al. [10] proposed automatic test pattern generation (ATPG) techniques that employ the divide-and-conquer paradigm.

Recently, HT detection methods using side channel-based analysis have been developed [72][76][56][41][65][105][53][31][49]. They characterize the target ICs for their manifestational properties, such as delay and power, in order to detect the embedded HTs. For example, two types of HT detection techniques analyzed pertinent ICs in terms of their delay from one flip-flop to another using either deterministic [56] or statistical methods [41]. A number of HT detection techniques advocate the use of switching power measurements[11]. Researchers from UCLA [70] advocate leakage current-based HT detection techniques.

Tehranipour et al. [81] presented a comprehensive survey of HT detection. There are two most common assumptions in the existing HT detection approaches: (1) there exists a golden model of the target IC; and (2) all the gates have the same process variation properties. In this dissertation, we do not impose these two assumptions for HT detection using our GLC and consistency-based analysis. Furthermore, we employ segmentation-based gate characterization into the process of HT detection, which ensures the scalability of the approach.

4.6 Summary and Discussions of Hardware Trojans

In this section, we summarize and discuss our consistency-based HT detection approach. First, we discuss more on the sources of hardware Trojan attacks; Second, we specify the boundary of the HT scenarios where our approach will be effective and where it may fail to detect the HTs. In particular, our discussion involves the following three aspects, namely size of the IC, type of the hardware Trojans, and type of the IC. Third, we explore the other possibilities of HT detection in addition to the HT trigger that we target on in this dissertation.

4.6.1 Discussions on HT attacks

There are many other research efforts that focus specifically on hardware Trojan attacks [3][45][31]. In this dissertation, we mainly focus on the additional malicious circuitry attack triggered by a single HT trigger. As a matter of fact, it is likely that multiple gates are used by an attacker as the trigger. In that case, the variations caused by the HT triggers is actually larger and easier to be identified. Also, the HT gates will be likely to appear in more segments than the single HT trigger case, making the probability of detecting the HT higher using the consistency-based approach. However, the multiple-trigger case does make the HT diagnosis process more complex, as all the locations on the IC must be examined after the HT is confirmed to exist, and there is no chance for the diagnosis to terminate early once any HT triggers have been found. However, the complexity of the procedure is still linear by using our consistency-based approach.

Also, in reality, HTs can be embedded in various stages in the IC design and manufacturing flow. For example, during IC design, the CAD tools may have been compromised that would generate HT-infected designs. Also, during IC

manufacturing, an untrusted foundry may embed malicious components into the design. We observe that the manufacturing time HT embedded by a foundry is the most challenging for detection, as it can bypass all the design verification steps. Therefore, in this dissertation, we mainly focus on the manufacturing-time hardware Trojans.

4.6.2 Boundaries of the HT Detection Approach

4.6.2.1 Size of the IC

In my simulation, I evaluated the consistency-based HT detection and diagnosis approach using benchmarks with up to 32,192 gates (i.e., ITC'99 benchmark b17). This is the upper bound of the supported IC size that I claimed in the dissertation.

However, the scalability of our consistency-based approach is guaranteed by the segmentation and variable elimination techniques. Also, the modern large-scale IC designs often contain independent and naturally segmented components, such as arithmetic logic unit (ALUs) and third party IPs. This phenomenon makes it possible to employ a multi-level divide-and-conquer paradigm and ensure the scalability of the approach.

4.6.2.2 Type of the HT attack

In this dissertation, I only focused on the detection of two types of hardware Trojans, including (1) the additional malicious circuitry triggered by a single HT trigger (discussed in Chapter 4.1.1), and (2) energy hardware Trojans triggered by adaptive body biasing (discussed later in Chapter 6.2). The reason why I focused on these two types of HTs is that, compared to other HTs, they are they

impose less instrumentation to the target IC and thus can be more easily hidden from the detection approaches. Therefore, we assume that these two types of HTs are challenging and representative for the discussion of HT detection techniques.

In addition, I proposed a new customizable and zero-overhead hardware Trojan attack by leveraging the redundant states in the finite state machine (discussed in Chapter 4.1.2), in order to motivate new research efforts in the HT detection space. The detection of such a customizable HT is left as the future work of this dissertation.

Finally, I do acknowledge that there are many other types of HTs discussed in the community, such as [3][45][31]. It is not possible for a HT detection approach to effectively address all types of HT concerns. This motivates us to investigate research benchmarks for hardware Trojans that would evaluate and motivate the HT detection research, as discussed in Chapter 4.1.1.

4.6.2.3 Type of the IC

In the dissertation, I only discussed the detection and diagnosis of hardware Trojans embedded in combinational and sequential application-specific integrated circuits (i.e., ASICs). So far, my approach does not support other types of system components, such as FPGAs and memory cells.

4.6.3 Target of HT Detection

In addition to the HT detection method targeting on the HT trigger, it is also possible that detecting the HTs via the “malicious circuitry” part. As a matter of fact, it is even easier to do, since the malicious circuitry is much larger and more observable than the single HT trigger. However, this is under the assumption that

the malicious circuitry is already activated or at least powered on. In reality, as I discussed in the dissertation, an advanced attacker is likely to power off the malicious circuitry during normal IC operation and, in the meantime, make the activation (i.e., power-on) of the malicious circuitry extremely rare, for example, with some special trigger condition that is only known and controllable by the attacker. In this case, the chance of having the malicious circuitry activated and detectable is very low. However, it provides us with another possible option for HT detection, which may trigger a new wave of detection approaches. For example, one may detect the HTs by observing the representative patterns that a malicious circuitry may depend on, such as the power gating structure. Although the applicability of this idea still needs to be investigated, we agree that this is a good topic that help complete the scope of the dissertation and motivate future research efforts.

CHAPTER 5

Hardware Metering and Digital Rights Management

In this chapter, we discuss the application of our GLC and consistency analysis approach in the domain of hardware metering and digital rights management (DRM). In particular, we develop a robust passive hardware metering approach based on the characterization of gate-level physical properties [91], which provides us with a unique and stable ID for each IC. Then, we conduct statistical analysis to the IC IDs to quantitatively identify the unauthorized IC manufacturing [86].

5.1 Hardware Metering Using Physical GLC

5.1.1 IC Metering Overview

With the rapid growth of integrated circuit (IC) outsourcing, *IC metering* [50][51][2][4] has become an important procedure in deterring or detecting the unauthorized IC production. More formally, IC metering or hardware metering refers to tools, methodologies, and protocols that enable post-fabrication tracking of the ICs. The metering approaches proposed thus far can be classified into two categories, namely passive metering and active metering. In passive metering, the ICs are individually identified, either in terms of their functionalities, or by other forms of unique identification. The identified ICs may be matched against their record

in a pre-formed database that could reveal unregistered ICs or overbuilt ICs (in case of collisions). The advantage of passive metering is that the intrinsic process variation of the legacy chips, without any modifications, can be exploited to identify and track each individual chip [4]. For a more comprehensive review of hardware metering, we refer the readers to recent surveys on the topic [46] [47]. In active metering, not only the ICs are specifically identified, but also parts of the IC functionalities can be only accessed, locked (disabled), or unlocked (enabled) by the design house or intellectual property (IP) owners via exploiting the design details that are not transferred to the foundry.

Our goal in IC metering is to characterize the physical level properties of the sampled chips and quantify the process variation model for all the manufactured chips. We take into account both the manifestational test properties (e.g., power and delay) and physical device properties (e.g., threshold voltage and effective channel length). From the power and delay models (i.e., Chapter 2.1 and Chapter 2.2), we observe that the conventional gate-level manifestational properties are impacted by many factors, which make the property values unstable and unpredictable. For example, the temperature (T) impacts leakage power exponentially, which means that the leakage power would have a large variation when the temperature varies due to IC activities or environmental factors. Therefore, the manifestational properties are not appropriate for the purpose of IC identification and, therefore, we consider using physical level properties as the IDs for the chips.

5.1.2 IC Metering based on Physical Level Characterization

Our flow of IC metering is shown in Figure 5.1. We first conduct gate-level characterization to determine the power/delay of each gate on the sampled chips,

which is done by solving a system linear equations using linear programming. Then, we conduct physical level characterization to calculate the V_{th} and L_{eff} of each gate, based on the manifestational properties and the models shown in Chapter 2.1 and Chapter 2.2. This is a nonlinear programming process since the models of power and delay are nonlinear with V_{th} and L_{eff} . Finally, we conduct process characterization to determine the parameter values in the PV model of all the manufactured chips.

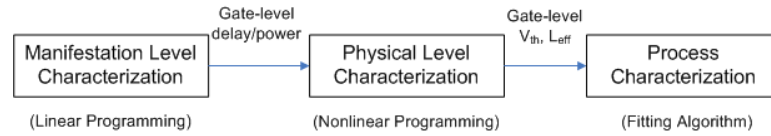


Figure 5.1: Flow of IC metering.

5.1.2.1 Manifestation-level Characterization

We use the GLC method discussed in Chapter 3.2 and Chapter 3.3 to characterize the manifestational properties in the presence of process variation. In the GLC method, the power and delay models are expressed in a linear format assuming that the variation of all the physical level properties is represented by a single PV scaling factor s . Then, the value of s can be obtained by solving a system of linear equations, as shown in Equation (3.2).

5.1.2.2 Physical Characterization based on Thermal Conditioning

From the characterization results from the manifestational properties, we are able to formulate a nonlinear equation based on Equation (2.1) in the following format:

$$P_{leakage} = \frac{A}{L} \cdot T^2 \cdot e^{\frac{C-V_{th}}{BT}} \quad (5.1)$$

where L and V_{th} are the two variables that we are characterizing. A , B and C are transistor level parameters in the leakage power model that we assume as constant values.

Equation (5.1) provides us with a nonlinear equation that relates L_{eff} and V_{th} to the manifestational properties (leakage power). We can obtain the leakage power value from the characterization as discussed in Chapter 5.1.2.1. However, with only one nonlinear equation, we are not able to solve two variables L_{eff} and V_{th} . Therefore, we must find a way to add additional variations to the leakage power model, so that a system of nonlinear equations can be obtained. We achieve this goal by varying the temperatures of the circuit using thermal conditioning. According to Equation (5.1), leakage power has a non-linear relation with temperature T . Therefore, we can use thermal conditioning to control the temperatures and obtain multiple leakage power nonlinear equations for each single gate. By applying different T to the IC and repeat the manifestational property characterization in terms of leakage power, we can formulate a system of nonlinear equations. Then, we solve the nonlinear equations using a nonlinear program solver and obtain characterization results for V_{th} and L_{eff} .

5.1.2.3 Process Characterization

In process characterization we aim to find out the quantified PV model parameters as discussed in Chapter 2.3 for all the manufactured chips. In particular, for the quad-tree model of L_{eff} , we characterize the Gaussian distribution parameters for the ΔL at all levels. For the model of V_{th} , we calculate the mean and variance in the Gaussian distribution.

For the V_{th} distribution, we can refer to a Gaussian fitting tool that can provide distribution parameters (mean and variance). Then, we use the obtained parameters as the estimation of those for the entire chip population. For the quad-tree model of L_{eff} , the problem becomes more complicated because it is a sum of multiple Gaussian distributions on multiple levels, and there is no direct way to break down the compound distribution and obtain parameter values for each single distribution. In order to solve the problem, we develop a decomposition algorithm and use a divide-and-conquer approach to keep fitting the sampled L_{eff} (compound distribution) to individual distributions. The objective in this process is to fit the individual distributions to Gaussian distributions as accurate as possible, i.e., optimize the approximation error provided by the Gaussian fitting tool for each individual distribution. Our solution is based on the fact that a Gaussian distribution is infinitely divisible, i.e., a Gaussian distribution X with mean μ and variance σ can be decomposed to multiple Gaussian distributions X_i with mean of μ_i and variance of σ_i , where the following equations hold:

$$\sum_i \mu_i = \mu \quad (5.2)$$

$$\sum_i \sigma_i^2 = \sigma^2 \quad (5.3)$$

Based on this divisibility feature of Gaussian distribution, we design a decomposition algorithm of process characterization. We start from the highest level (root) of the quad-tree and conduct a breadth-first search of the tree. At each node, we guess and verify the constant component of the leaf node with the requirement that the remainder obtained by subtracting this constant component from the L_{eff} value should follow a Gaussian distribution, which is the L_{eff} value of the lower level nodes of the current node.

5.1.3 Coincidence Estimation

An important and challenging step in IC auditing is to be able to distinguish each chip from the others. Due to the possible measurement and characterization errors in the IC metering process, there are possibilities of false positives and false negatives. The former means that we count chips that are not of our design as ours, and the latter means the opposite. Our goal in coincidence estimation is to measure the probabilities of false positives and false negatives, so that we can estimate their impacts on the accuracy of IC auditing.

We develop a Bayesian-based approach to calculate the probability of coincidence when only a single gate on each chip is considered. Then, we employ a majorization technique to conduct worst case analysis, which assumes that all the gates in the circuit have the worst case (i.e., the highest) probability of coincidence. From this analysis, we obtain an upper bound of the probability of coincidence and use it to analyze the impact on IC auditing.

5.1.3.1 Bayesian-based coincidence analysis

Since our IC metering is based on the characterization results of L_{eff} and V_{th} , there is a possibility that two gates that are not from the same chip would have the same measured L_{eff} and V_{th} due to either measurement errors or characterization errors.

Our IC auditing process works in the following way. We take a sample chip from the market and conduct IC metering to obtain L_{eff} and V_{th} , and we label this chip according to the L_{eff} and V_{th} values. Then, we put this chip back to the market and continue to take other samples. It is possible that the another sample is not the same chip as the previous labeled chip but we characterize

them as similar L_{eff} and V_{th} (i.e., false positives), or it is the same chip as the previous labeled chip, but we have different L_{eff} and V_{th} measurements (i.e., false negatives).

We employ Bayesian-based probability analysis [63] to calculate the probability of coincidence. Taking the false positive case as an example, we have the following Bayesian-based calculation:

$$P(H|D) = \frac{P(D|H) \cdot P(H)}{P(D)} \quad (5.4)$$

where H is the event that a gate matches with at least one other gate according to either L_{eff} or V_{th} measurement. D represents the event that we have a certain set of L_{eff} or V_{th} measurements for N sampled chips. Therefore, $P(H|D)$ represents that the probability that a gate matches with other gates under the condition that we have that certain set of measurements; and $P(D|H)$ is the probability of having the certain set of measurements under the condition that the gate matches with some other gates. We assume that $P(D|H)/P(D)$ forms a normalization constant that does not vary with the variation of D . We calculate $P(H)$ in the following way by using the rationale in the well known birthday paradox problem [32]:

$$P(H) = 1 - \prod_1^N (1 - P_i) \quad (5.5)$$

where P_i is the probability that a certain gate i matches with one another gate j . The value of P_i depends on the position of the L_{eff} or V_{th} in the whole distribution. Figure 5.2 shows our simulation results of P_i in terms of L_{eff} . We can see that the gates with L_{eff} around the mean value ($L_{eff} = 1.2$ in the figure) of the distribution have a relatively large P_i .

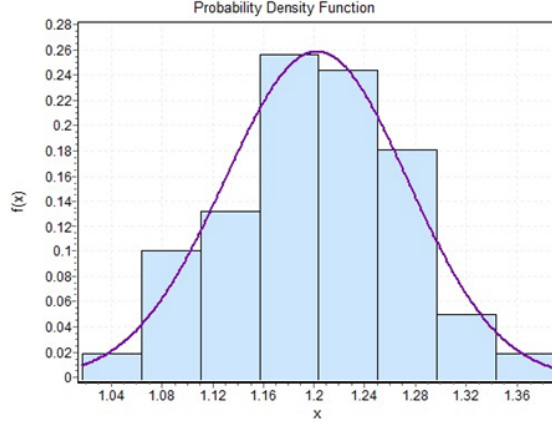


Figure 5.2: Probability that a gate has coincidence with other gates in terms of L_{eff} (benchmark C432 with 160 gates; mean value of L_{eff} is 1.2).

Putting it all together, we have the following estimate for $P(H|D)$:

$$P(H|D) \propto 1 - \prod_1^N (1 - P_i) \quad (5.6)$$

We use two approaches for determining whether two chips match with each other: (1) an extreme method, in which we claim two chips are identical as long as there is an overlap between the distributions of their measured values; and (2) a threshold approach, in which only when the overlap between two distributions exceeds a threshold value do we assume they are identical.

5.1.3.2 Majorization and Worst Case Analysis

As mentioned in the previous subsection, the probability P_i varies over the absolute value of L_{eff} or V_{th} , and it reaches the highest value if the sampled chip is at the mean value of the entire distribution. In order to conduct coincidence estimation considering all gates on a chip, we must take into account the variations of P_i for all the gates. In our coincidence estimation process, we approximate

the value of each P_i using a majorization technique. In other words, we use the highest possible P_i (that of the mean value L_{eff} or V_{th}) to represent all the P_i values. In this way, we indeed overestimate the probability of coincidence and obtain an upper bound value for the worst case analysis.

5.1.3.3 Summary of Coincidence Estimation

From the results in Table 5.1, we can conclude that the worst case probability of coincidence is small enough to hold a large number of chips (e.g., in millions), and the probabilities of false positives and false negatives are close to zero. This conclusion enables us to assume that all the chips are distinguishable from each other and we can label them uniquely without overlaps. This is important in the next step of our IC auditing process, because the sampling and re-sampling in the IC auditing approach are based on replacement.

5.2 IC Auditing Using Statistical Analysis

5.2.1 IC Auditing Overview

IC digital rights management (DRM) [50][5] has drawn a great deal of attention in the recent years with the fast growth of outsourcing in the IC industry. In the current model of IC manufacturing, the IC design companies deliver their designs to IC foundries without having any control over the manufacturing process. In this process, it is likely that an untrusted IC foundry fabricates a larger number of ICs than it was authorized to produce. Such a misconduct has become a crucial concern in the IC industry, with illegal copies of ICs costing the design companies billions of dollars annually.

The existing IC metering approaches can detect intellectual property (IP)

violations in IC manufacturing. However, they cannot quantify the number of counterfeit chips. This makes it extremely challenging to charge the parties that use the IPs. To address this issue, we propose a new concept of IC auditing, in which we aim to provide a quantitative estimation of the number of chips released to the IC market. Our strategic objective is to create a new IC auditing technique using statistical analysis. In particular, we employ an animal counting model that predicts the total population from a partial sampling and labeling of the chips in the market.

5.2.2 IC Auditing Model

Based on the IC metering with near-zero false positives and false negatives, we are able to conduct IC auditing using a sampling approach. Our IC auditing scheme is based on the animal counting techniques proposed in the statistical field [78] [21]. The main idea is to predict the total population of a kind of animals by capturing and recapturing samples. In this section, we show how our IC auditing problem is adapted to the animal counting model and how we solve the IC counting problem based on the model.

The animal counting problem was first studied for estimating the dynamic of biological populations. One of the widely used approaches is the capture-recapture method [78] [21], in which samples are taken and labeled at periodic intervals. Then, the total population can be predicted from the number of captured, and more importantly, recaptured animals in each sample. For example, in the fish counting problem discussed in [78], the following information is recorded for each sample i : (i) the total number of fish (t_i); (ii) the number of new fish (d_i); and (iii) the number of recaptures (r_i). Next, the probability of obtaining such a sample can be calculated by using binomial distribution:

$$p_i = \binom{t_i}{r_i} \left(\frac{M_i}{N}\right)^{r_i} \left(1 - \frac{M_i}{N}\right)^{d_i} \quad (5.7)$$

where N is the total number of fish, and M_i is the number of labeled fish when the i th sample is drawn. Assuming all the samples are taken randomly and independently, the probability of obtaining n samples with specific t_i , d_i , and r_i is the product of p_i : $P = \prod_{i=1}^n p_i$. Then, by using maximum likelihood analysis, Schnabel [78] gives the equation that holds for N and M_i :

$$\sum_{i=1}^n \frac{d_i M_i}{N - M_i} = \sum_{i=1}^n r_i; \quad (5.8)$$

Schnabel [78] solves the equation and gives an approximation solution of N as the following:

$$N = \frac{\sum_{i=1}^n t_i M_i}{\sum_{i=1}^n r_i} \quad (5.9)$$

Equation (5.9) indicates that the predicted number of fish is a function of t_i , M_i , and r_i . All of these parameters can be obtained easily from the sampling and labeling process.

Our IC auditing problem is similar with the animal counting problem, in both required inputs and outputs. However, we must analyze the assumptions behind the problem and verify that our IC counting problem still makes the assumptions hold. We note that the fundamental assumptions that are required by the animal counting model include the following: (1) there must be a method to uniquely label the captured samples; and (2) the sampling model must be with replacement so that the captured samples can be recaptured, which provides an indicator on how large the total population is.

From the discussion in Chapter 5.1.3, the first assumption holds because the probability of coincidence becomes extremely small when we consider all the gates on the chip. For the second assumption, we make our IC auditing process spread into the IC marketing period. In other words, we collect IC samples periodically and put them back into the market after each sampling period. This would make our auditing process long, but it is doable. Furthermore, the number of samples can be adjusted according to the required accuracy of the prediction results.

Based on the above analysis, we apply the animal counting technique to our IC auditing problem. Here we use the same symbols of t_i , d_i , and r_i as in the animal counting model. The number of chips can be predicted by Equation (5.9).

5.2.3 IC Auditing Post-processing

The accuracy of the prediction results can be impacted by many factors, such as the degree of independence of the samples and the approximation errors in the animal counting model. In order to improve the accuracy of IC auditing, we employ a statistical method, namely maximum likelihood estimation to post-process the data after many runs of the sampling experiments have been conducted. Then, we apply goodness-of-fit tests [55] on the data from each run, and estimate the statistical distribution of the predicted results over different runs. According to the distribution that each result follows, we obtain its approximate density function, i.e., $p(N)$, and set our estimated value of N to be the one that maximizes the likelihood function:

$$\tilde{N} = \operatorname{argmax}_N \log p(N) \quad (5.10)$$

5.2.4 IC Auditing Validation

We can validate our prediction results in two ways. One is to experiment it directly on a known set of chips. By comparing the actual number of chips and our predicted results, we can draw a conclusion on the accuracy of our prediction model. Figure 5.3(a) shows one of validation results, in which we apply our IC auditing approach to unknown sets of chips with up to 1500 chips. For each set of chips, we plot and compare our prediction results with the actual number of chips. We observe from Figure 5.3(a) that the estimated N is close to the actual N , but the distance between them is increasing as the actual N grows. This is due to the fixed number of samples and sample sizes, which are not enough when the population is large.

Another method for validation is to conduct statistical analysis on multiple runs of the sampling process. In particular, we repeat the experiment many times and compare the results of each run in terms of the variance of the predicted results. If the variance is within a small enough range, it indicates that our prediction model converges and is stable. Figure 5.3(b) shows our validation results based on this method. We repeat the experiment 500 times for a known set (800) of chips. The plotted results of predicted number of chips indicate that they are within the range of 25% of the actual number of chips.

5.3 Hardware Metering Evaluation Results

We simulate our IC metering and auditing schemes on a set of ISCAS benchmarks. We use leakage power as the manifestational test properties in the simulation because every gate on the circuit has leakage power regardless of its activities. This provides us with more variabilities in metering and labeling the gates.

5.3.1 IC Metering

We use the manifestation-level characterization results as well as thermal conditioning to formulate a system nonlinear equations. In this simulation, we use 20 nonlinear equations (temperatures) per gate and obtain V_{th} and L_{eff} for each gate by solving the system of nonlinear equations. We solve the nonlinear equations by using the Gauss-Newton method provided by Matlab. The PV model we use in the simulation is the quad-tree model as discussed in Chapter 2.3.

Figure 5.4 and Figure 5.5 show the accuracy of our characterization results for L_{eff} and V_{th} , respectively. In each benchmark, we characterize each gate and compare the characterization results with the actual values to calculate the accuracy of our characterization. We plot the relative characterization errors for all gates in histograms and fit them into a distribution as shown in the curves. We can see from the curves that we have less than 1% of average errors and less than 5% of maximum errors except for few outliers. We consider these error distributions in the next steps where we conduct coincidence estimation and IC auditing.

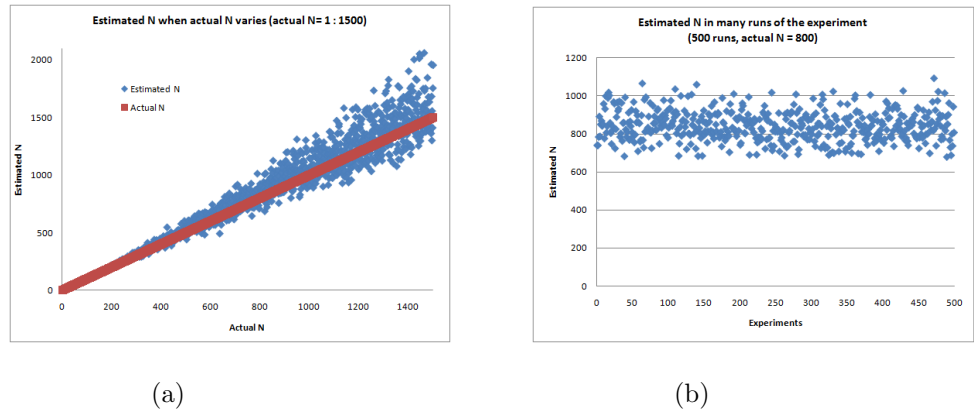
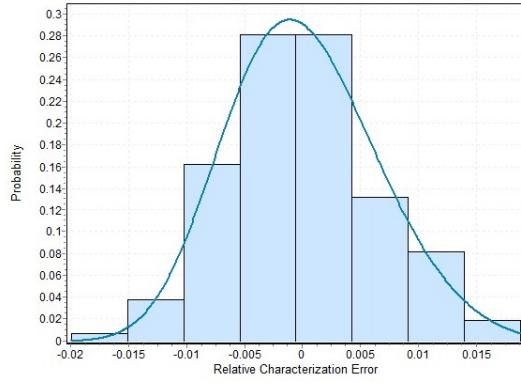
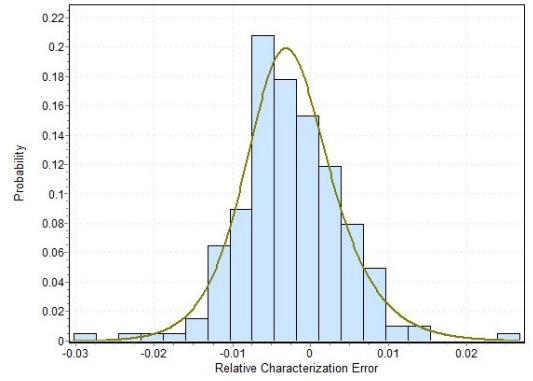


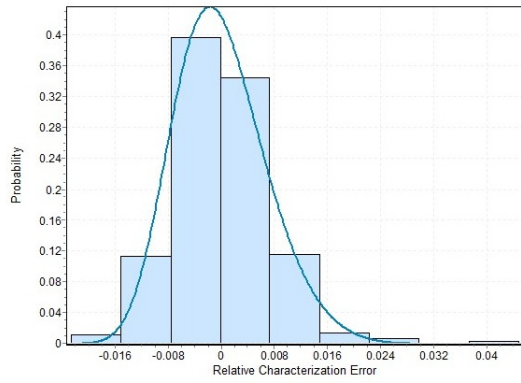
Figure 5.3: Validation of our IC auditing scheme: (a) on known sets of chips; N varies from 1 to 1500; (b) on 500 runs of the analytical simulation; N is fixed to 800.



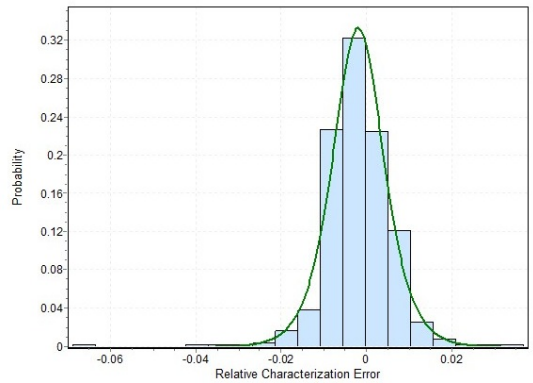
(a) Benchmark C432 (160 gates)



(b) Benchmark C499 (202 gates)

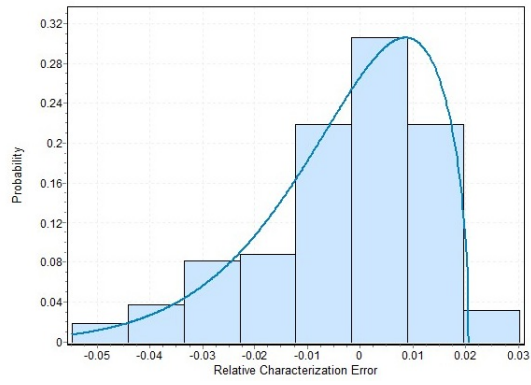


(c) Benchmark C880 (383 gates)

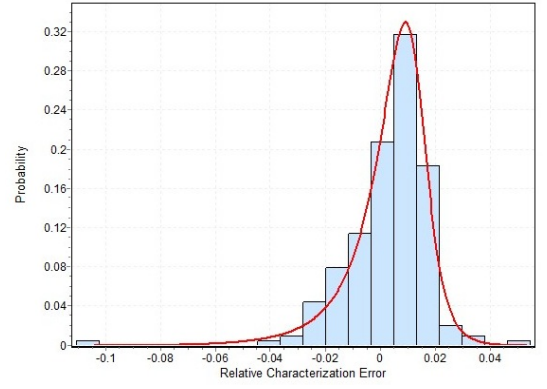


(d) Benchmark C1355 (546 gates)

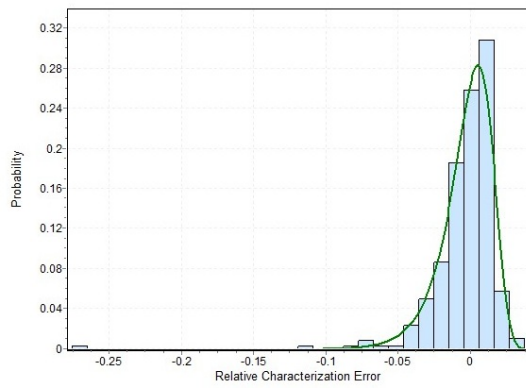
Figure 5.4: Accuracy of L_{eff} characterization on a set of ISCAS benchmarks.



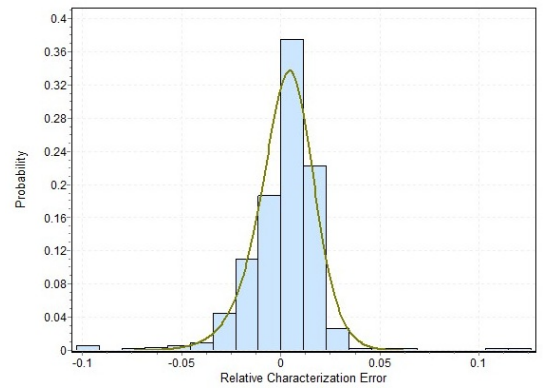
(a) Benchmark C432 (160 gates)



(b) Benchmark C499 (202 gates)



(c) Benchmark C880 (383 gates)



(d) Benchmark C1355 (546 gates)

Figure 5.5: Accuracy of V_{th} characterization on a set of ISCAS benchmarks

Table 5.1: Accuracy of coincidence estimation. “FP” and “FN” stand for “False Positives” and “False Negatives”, respectively.

Benchmark	GLC Error (%)	Extreme Method		Threshold Method	
		FP (%)	FN (%)	FP (%)	FN (%)
C432	-2.0 ~ +1.5	6.9	0	5.5	64
C499	-2.0 ~ +3.0	9.8	0	7.8	64
C880	-6.0 ~ +4.0	19.0	0	15.4	64
C1355	-6.0 ~ +2.0	15.4	0	12.4	64
C1908	-3.2 ~ +3.2	12.4	0	10.0	64
C2670	-3.0 ~ +3.0	11.6	0	9.4	64
C3540	-3.0 ~ +3.0	11.6	0	9.4	64
C5315	-3.0 ~ +3.0	11.6	0	9.4	64
C6288	-2.0 ~ +3.0	9.8	0	7.8	64
C7552	-3.2 ~ +2.4	11.6	0	8.6	64

5.3.2 Coincidence Estimation

We perform coincidence estimation on the same set of ISCAS benchmarks and characterize the probabilities of false positives and false negatives when using both the extreme method and the threshold method (with a 20% threshold). Table 5.1 shows the results when considering one single gate on each chip. The extreme method gives zero false negatives and false positives from 6.9% to 19.0%, while the threshold method has lower false positives from 5.5% to 15.4% and constant false negative values depending on the threshold value.

Given the coincidence estimation for having only one single gate considered on each chip, we can calculate the probabilities of false negatives and false positives that consider all gates on the chip by using Equation (5.6). We find that the probability of coincidence becomes extremely small (e.g., 10^{-95} for benchmark C432) because of the large number (e.g., at least 160) of gates in the benchmark circuits. Considering the fact that there are many more (in millions or more) gates on a single chip in modern IC design, the probability of coincidence is very low even if there is a huge number of chips in the market.

5.3.3 IC Auditing

In our IC auditing simulation, we evaluate the IC counting model in terms of the prediction accuracy. Also, we estimate the impact of the sampling parameters, such as the number of samples and the sample size, as well as the impact of the total number of chips on the prediction accuracy.

5.3.3.1 Prediction Error vs. Number of Chips

We simulate our IC auditing scheme on different numbers of chips in order to find out how the total number of chips would impact the prediction accuracy. In Figure 5.6 we show the relative prediction errors when the number of chips varies from 1 to 2000, the number of samples is fixed at 20, and the size of each sample is 20. We observe that the relative prediction error becomes higher as the number of chips increases, but it is always below 15%. Also, we observe that the variance of the prediction error grows as the number of chips increases. This is due to the insufficient samples compared to the total number of chips. We will discuss the impact of the number of samples later in Chapter 5.3.3.2. Also note that the results in Figure 5.6 are obtained without post-processing, i.e., each experiment is conducted only once, which is another reason why the variance of the prediction error increases.

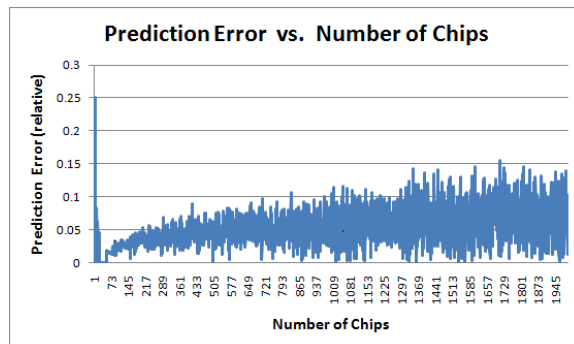


Figure 5.6: IC auditing results: prediction error vs. total number of chips (the number of samples is fixed to 20; the sample sizes are fixed to 20; the total number of chips varies from 1 to 2000; and no post-processing of the prediction results is performed).

Table 5.2 shows our simulation results on a large number of chips (up to 100 million). In this set of simulation, we set the sample rate (the ratio between the

number of sampled chips and the total number of chips) as 0.1%, 0.5%, or 1.0% of the total number of chips. Also, we repeat each experiment 100 times and conduct MLE post-processing towards the collected results. We observe that the estimation error decreases as the increase of the number of chips with the same sample rate. Also, a sample rate of 0.5% can provide us with estimation errors below 5% for large numbers of chips (10^7 or 10^8).

Table 5.2: IC auditing on large numbers of ICs.

Number of ICs	Total Sample Rate	Estimation Error
10^6	0.5%	15.0%
10^6	1%	3.77%
10^7	0.1%	29.9%
10^7	0.5%	2.16%
10^8	0.1%	6.34%
10^8	0.5%	5.04%

5.3.3.2 Prediction Error vs. Number of Samples

We find in our simulation results that the number of samples taken plays an important role in the eventual prediction accuracy. In order to find out more about the impact of the number of samples, we perform simulations on 1600 chips while varying the number of samples from 10 to 50, with 20 chips in each sample. We show the results in Figure 5.7. We can observe that the prediction accuracy keeps improving as the number of samples increases. This verifies our intuition that the prediction becomes more accurate with more information from the samples.

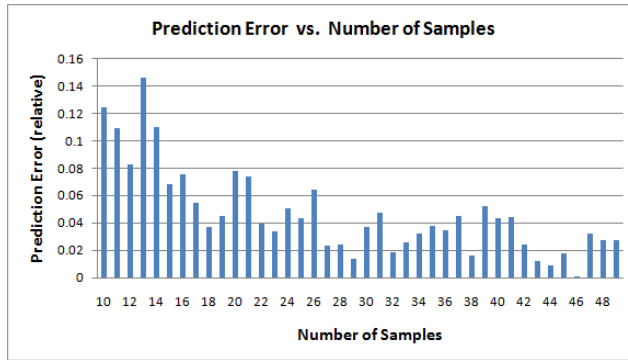


Figure 5.7: IC auditing results: prediction error vs. number of samples (the number of chips is 1600; the number of samples varies from 10 to 50; and the sample sizes are fixed to 20 chips.)

5.3.3.3 Prediction Error vs. Size of Samples

We further investigate the possible impact of the sample sizes by conducting a set of simulations on 1600 chips, with a fixed number of samples (e.g., 20) and varied sample sizes (e.g., from 10 to 50). We show the results in Figure 5.8, where there are no improvements in the prediction accuracy as we increase the sample sizes.

5.4 Hardware Metering Related Work

Metering and auditing have been recently studied in the area of web applications, such as for client counting for client/server management [64] and click fraud prevention [57]. Similarly, in the area of IC design and manufacturing, there are several active or passive IC metering schemes that have been proposed. Some of them require instrumentation in the design and manufacturing process, which are called extrinsic metering; the others utilize the existing IC characteristics for metering without modifying the design flow, which are called intrinsic metering.

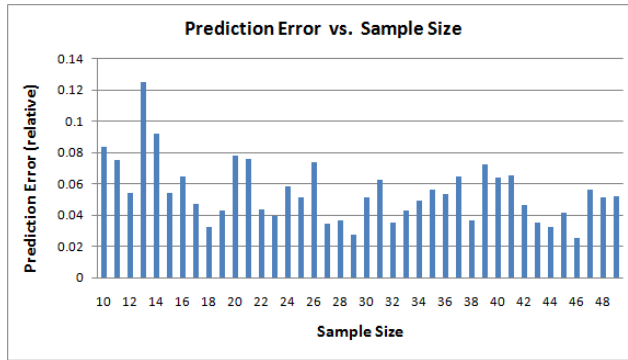


Figure 5.8: IC auditing results: prediction error vs. sample sizes (the number of chips is 1600; the number of samples is fixed to 20; and the sample sizes vary from 10 to 50.)

5.4.1 Extrinsic IC Metering

Extrinsic IC metering introduces extra hardware/software components to the chips, in order to make a unique identification for each chip and use it to detect IP violations. Extrinsic IC metering methods maybe either active or passive. Fingerprinting schemes [18] [42] assign a unique fingerprint on each IP that the manufacturer is allowed to use. The manufacturer is supposed to use each IP once when producing the chips. Therefore, each chip would have a unique fingerprint compared to the other chips. Then, the design company can detect the IP violation by finding out the chips with the same unique fingerprint. Another extrinsic metering scheme [51] adds a small programmable component in each design which can be configured in a unique way for each chip during the manufacturing process. The foundry reports to the design company all the IDs of the manufactured chips. To detect IP violation, the design company would conduct a random sampling in the market and record the number of unreported chips. From a statistical analysis based on collision probabilities computed by the Birthday paradox, the number of unauthorized chips can be estimated.

Extrinsic metering approaches can detect IP violations but they require a high instrumentation to either the design or manufacturing process. It complicates the IC design process and increases the cost of each chip. Also, there are still security concerns in this scheme, because the design company do not have control over the manufacturing process, it is possible that untrusted manufacturers modify the assigned fingerprint or ID and compromise the IP protection scheme.

5.4.2 Intrinsic IC Metering

Intrinsic IC metering approaches do not interrupt the IC design and manufacturing process. Instead, they characterize the existing properties of the chips and assign a unique ID obtained from the characterization results of each chip. The IDs are used in the same way as in the extrinsic metering scheme. Koushanfar et al. [50] proposed a CAD-based intrinsic passive IC metering approach. It characterizes each gate of an IC in terms of its delay on critical path. Because of the existence of process variation, the delay values of the gates are different even if they are from the same design. Therefore, the delay value can be used as a unique ID of the IC. Alkabani et al. [4] proposed a nondestructive approach for gate-level characterization and a hardware metering protocol based on the characteristics. They analyze the probability of collision of IDs in presence of intra- and inter-chip correlations.

The intrinsic metering approaches avoid the instrumentation to the IC design and manufacturing process and are still able to generate unique IDs for the chips. However, they would require high accuracy in the gate-level characterization results. Also, the existing approaches did not provide quantified solutions in terms of the number of chips that a foundry may have produced.

CHAPTER 6

Remote In-field Wireless Security Techniques

In this chapter, we discuss the application of our GLC and consistency analysis techniques in wireless security applications. We focus on addressing the issue of security attacks in remote in-field wireless systems, where we must collect power profiling data in a trusted and secure manner.

6.1 Wireless Security Challenges

Recently, wireless communication, computation, and sensing devices, such as mobile phones, laptops, and tablets, have been experiencing exceptionally explosive growth. For example, every second more than 30 cell phones are sold worldwide. In addition, emerging industrial sensor networks are both economically and strategically important. Furthermore, wireless security imposes a technically challenging set of objectives and requirements. For example, side channel and fault induction security attacks [36][37][50][68][84][93] are much more likely on cell phones and, in particular, on sensor nodes that may be deployed in unprotected or even hostile environments. Also, operational conditions and numerous design constraints such as low energy, low power, and low cost impose difficulties on security requirements.

As a consequence, wireless security has emerged as a premier research and development issue. Numerous important aspects have been addressed, such as

key management schemes in mobile ad hoc networks [19] and distributed sensor networks [30], secure routing protocols [8] and localization algorithms [54] to prevent wireless sensor attacks, and privacy protection in RFID systems [75].

However, none of these important contributions address the detection of hardware Trojans (HTs) [43][81] in the wireless systems. HTs are in a sense the most powerful way to completely compromise the security of any wireless or other devices, because they enable the attacker to bypass the system security mechanisms, access any storage element, change access rights of any program, and abuse (e.g., induce high energy consumption) or destroy any piece of hardware. Following the discussions in Chapter 4, in this chapter, we investigate the attacks and defense techniques for Hardware Trojans in wireless systems. In particular, we focus on the discussion of an energy attack model that targets on the most important and limiting resource (i.e., energy) in the wireless systems. In addition, we discuss how we detect such an attack remotely during the system operation and, more importantly, how we ensure the security and robustness of the online remote detection approach.

6.2 Energy Hardware Trojans in Wireless Systems

6.2.1 Energy Hardware Trojan Overview

The existing HT detection efforts targeted only on HTs that cause direct security attacks, such as implanting a backdoor in the circuit and extracting confidential information from the system at runtime [40][45]. However, we note that the security toward one of the most crucial and fundamental components in embedded systems, namely energy consumption, has rarely been discussed until now.

It is well acknowledged that power efficiency is one of the most important

design objectives of all embedded systems. Although huge efforts have been put on energy reduction and optimizations at the design time, the security aspects of energy efficiency for embedded systems have been seldom discussed. We argue that the embedded systems that are assembled using ICs from untrusted foundries are vulnerable to energy attacks, which aim to increase the energy consumption and thus reduce the lifetime of the system. For example, an untrusted foundry of a cell phone chip may have embedded malicious components in the hardware that leak additional energy from the phone, create hot spots on the chip, and eventually make the phone malfunction.

In this section, we aim to analyze and address the potential security concerns raised by the new type of hardware Trojans, namely energy HTs. To achieve this goal, we first develop and evaluate the brand new attack model to create challenging energy attacks that are capable of bypassing existing detection mechanisms. We focus on the two most important design components of an energy HT: (1) HT action, which indicates the malicious behavior conducted by the energy HT; and (2) HT trigger, which is the runtime condition that activates the HT to perform the HT action. For HT action, the idea in designing such an energy attack model, which is also the best interest of an attacker, is to maximize the damage to the target circuit but camouflage it under certain known factors or phenomena. In particular, we exponentially increase the energy consumption of the target circuit using adaptive body biasing (ABB) [22] that was originally invented for IC optimizations. Meanwhile, we tend to camouflage the damage or, in other words, attribute the impact of HT action, to environment temperatures that vary frequently and also have huge impacts on the leakage power [59]. Consequently, even if the detection approach is able to capture the energy hike, it is extremely difficult to distinguish between the impact of natural temperature variations and the malicious attack. For the HT trigger, the design strategy is to minimize its

observability on the target design, so that it can bypass the security checks. In our proposed energy HT model, we hide the HT trigger under the target circuit by minimizing its observable power and delay. Also, we minimize the probability of activation by associating the trigger with a combination of multiple input signals or a set of consecutive states in the finite state machine.

6.2.2 Energy Hardware Trojan Attacks


In this section, we discuss powerful energy attacks in both the sleep mode and the system operation mode.

6.2.2.1 Sleep-mode Energy Attack: Input Vector Manipulation

For most of the wireless systems and applications, the circuit of the system would stay in the sleep mode for a large portion of the time. For example, in the case of a cell phone, the major components are only exercised once it is on a voice call. Similarly, in a wireless sensor network, the communication circuitry only operates during the data collection process. Consequently, the input vectors that are being applied during the IC sleep mode becomes crucial with regard to the leakage energy consumption, because of the fact that the leakage energy of a logic gate highly depends on the input vectors [104]. For example, as shown in Figure 6.1, the leakage energy of a NAND gate can vary up to 12 times with different input vectors. Although this phenomenon can provide us with an opportunity for leakage energy reduction, it is more easily leveraged by an attacker for energy attack.

Figure 6.2 shows a motivational example of energy attack using input vector manipulation. In this case, the HT component applies an input vector that sets the maximum number of gates in the high energy state, which results in the

Input	Leakage Current
0	100.3 nA
1	227.2 nA



Input Vector	Leakage Current
00	37.84 nA
01	95.17nA
10	100.3 nA
11	454.5 nA




Figure 6.1: Leakage current of an inverter and a NAND gate under different input vectors [104].

highest energy consumption (i.e., 2.96 times of the minimum energy consumption). If this situation continues over time without being identified by the user or tester of the system, it will cause several times more energy consumption, which is considered significant in a power hungry system.

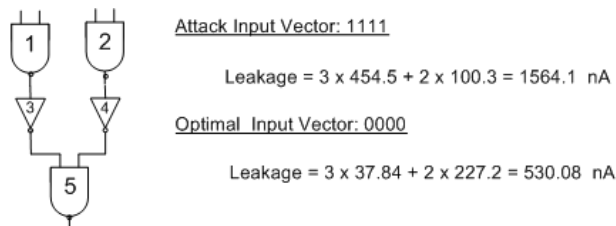


Figure 6.2: Example of energy attack via input vector manipulation. The energy consumption caused by the attack input vector is 2.96 times compared to the optimal input vector.

6.2.2.2 Operation-mode Energy Attack: Forward Adaptive Body Bias

During IC operation, the supply voltage plays an important role to the total leakage power consumption. According to Equation (2.1), the leakage energy of a transistor increases exponentially with the increase of supply voltage. We argue that this phenomenon can be leveraged for powerful energy attacks, since

an exponential energy increase could cause huge impacts to the wireless system. In particular, we argue that an attacker could possibly apply a forward adaptive body bias (FBB, or forward ABB) voltage that, instead of compensating for the process variation as in the normal use of body biasing techniques [22][35][102], would increase the energy exponentially in the circuit under attack.

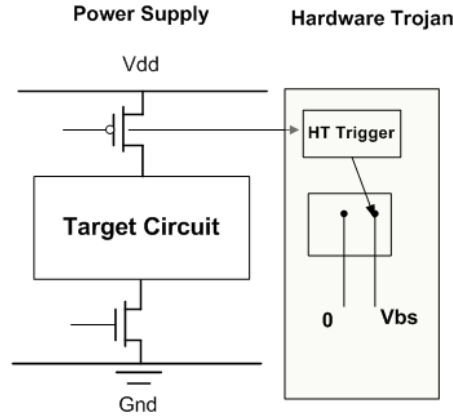


Figure 6.3: Example of energy attack using forward body biasing.

With this consideration, we implement a sample malicious circuitry that triggers the FBB-based energy attack, as shown in Figure 6.3. The shaded part of the circuit is the malicious component, or HT, embedded by an adversary. During the operation mode, once triggered, the HT can select and apply a forward body bias voltage, which increases the supply voltage and maximizes the leakage energy without compromising the functionality of the system.

6.2.2.3 HT Triggers: Rare Activation of Attacks

In the previous two subsections, we have shown that HT-based energy attack could cause huge energy increase, either linearly in the sleep mode or exponentially in the operation mode. However, the attacks would not take effective unless the triggers are well hidden from the common detection approaches. In this sub-

section, we discuss in details how an attacker could design the trigger such that the resulting HT attack has a low probability to be detected by the HT detection attempts. The intuitions of hiding the HT trigger, from the attacker’s perspective, include the following: (1) hide the HT trigger in the circuit both physically and in terms of their observable properties, such as delay and power; and (2) minimize the activation probability of the HT.

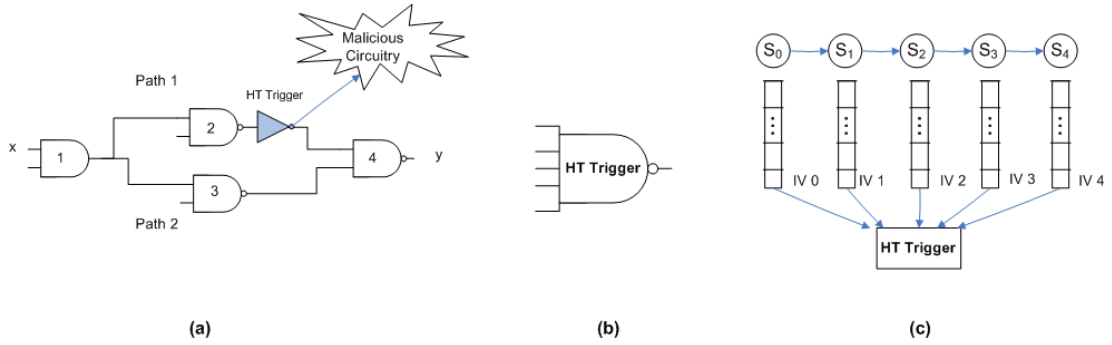


Figure 6.4: Example of hiding the HT triggers: (a) a HT trigger embedded in the reconvergent paths where delay is non-observable; (b) a rarely switching HT trigger driven by multiple inputs; and (c) a HT trigger activated by a 5-state finite state machine, which reduces the activation probability exponentially.

In order to bypass the most commonly used side channel-based detection approaches, an attacker aims to place the HT trigger in such a way that it is non-observable via the commonly considered side channels, including delay, leakage power, and switching power. In order to achieve this goal, we add only one single gate in the target circuit that serves as the trigger. With this single gate HT trigger, we ensure that any resulting delay or power variation is minimum to increase the difficulty level for detection. Furthermore, to further complicate the side channel-based detection approaches, we place the single gate HT trigger at a circuit location where the delay and power are non-observable or difficult to measure. For example, in Figure 6.4(a), the delay of the HT trigger is non-

observable due to the parallel reconvergent paths. One can measure the delay between the two endpoints x and y . However, it is not possible to determine whether the measured delay is for path 1 or path 2 and, therefore, the HT can be hidden under the delay measurements.

Based on the HT trigger placement that is difficult to detect, we further reduce its activation probability to bypass the security checks that are based on generated test vectors, such as automatic test pattern generation (ATPG) [10]. The idea is to set the activation condition in such a way that it is only known to the attacker and very rarely triggered during a normal IC operation or test. We achieve this goal by using two approaches. Firstly, we select the fan-in gates from the target circuit in such a way that the HT trigger is rarely switched, as shown in an example in Figure 6.4(b). The activation probability of the NAND HT trigger is $1/2^n$, where n is the number of inputs that can be customized by the attacker to balance the trade-off between the size of the HT trigger and the activation probability. Secondly, we leverage sequential elements (i.e., flip-flops) that create temporal-based activation conditions in a finite state machine (FSM). In this way, the activation probability of the HT trigger can be further reduced exponentially based on the results from the first approach. For example, as shown in Figure 6.4(c), the 5-state FSM serves as the activation condition, which triggers the HT only when all 5 states are satisfied in 5 consecutive clock cycles. As a result, the activation probability is $\prod_{i=1}^m P_i$, where P_i is the activation probability of the vector in state i , and m is the number of states.

6.2.3 Energy Hardware Trojan Detection

Based on the investigation of the powerful energy attacks, we develop defense approaches for energy attacks by leveraging power profiling and temperature char-

acterizations. The idea is to sample the total leakage power consumption of the wireless system and characterize the gate-level temperature profile by assuming that there is no energy attack. In the case where there is indeed malicious energy attack, the characterized temperature profile will not meet the normal spatial and temporal thermal distributions on an IC. We identify the possible discrepancy in temperatures quantitatively by defining a hardware Trojan indicator concerning the spatial inconsistency of the gate-level temperatures. Note that our energy HT detection approach requires no instrumentation to the wireless system, which can be conducted remotely without having direct physical access to the target device.

6.2.3.1 Energy Paradox

The most straightforward detection approach towards the energy attack is by sampling the energy profiles of the operating wireless systems on a regular basis and observe the abnormal energy increase. Although remote sampling and data collection is a common practice for wireless system performance or status monitoring, the collected power profile is not an effective indicator for energy attacks, due to the following energy paradox, which can be leveraged by energy attackers:

Energy Paradox. Due to the exponential dependence of leakage energy on temperatures, as indicated in Equation (2.1), it is not certain to the system user whether the energy increase is caused by normal temperature variations or malicious energy attacks. As a matter of fact, it is common that the target wireless system, such as a wireless sensor network, is deployed in a hazardous environment where the temperature varies in an unknown pattern. In this case, the power profiling approach by itself is not sufficient to reach a conclusive judgment of whether any energy attack exists or not.

6.2.3.2 Gate-level Temperature Characterization

In order to address the energy paradox and obtain accurate energy HT detection results, in the case of a high energy profile, we must measure or characterize the temperature of the target IC to either exclude its impact or report that the energy increase is due to temperature. Several approaches have been proposed in monitoring the temperatures of IC systems, such as FEA [106], power blurring [44], and sensors [82]. However, the FEA and power blurring approaches work at the IC design stage without the taking account of the impact of process variation and are not resilient for post-silicon attacks. The sensors-based approaches provide real-time measurements of temperatures, but they require additional sensor circuitry in the target IC, which greatly increases the complexity and cost of the system.

We develop a non-destructive gate-level temperature characterization approach using power profiling, which does not require additional hardware circuitry being added to the target IC. The approach is based on the physical-level GLC concerning threshold voltage (V_{th}) and effective channel length (L_{eff}). We show the flow of temperature characterization in Algorithm 7. Firstly, before the release and deployment of the wireless system, we characterize the gate-level V_{th} and L_{eff} at room temperature, where we assume the temperature T in Equation (2.1) as a constant value. Then, after the system has been deployed and in operation, we take M power measurements and characterize the temperature (T_i) of each gate using Equation (2.1) based on the V_{th} and L_{eff} that are already known. Finally, we conduct online security checking to determine the presence of energy attack using the HT indicator, as defined and discussed in the next subsection.

Algorithm 7 Gate-level temperature characterization via power profiling.

- 1: *Post-silicon*:
 - 2: Gate-level characterization to solve V_{th} , L_{eff} of each gate at room temperature following Equations (2.1) and (2.2);
 - 3: *Runtime*:
 - 4: Take M power measurements via sampling;
 - 5: **for all** Gates g_i in the circuit **do**
 - 6: Solve for temperature T_i following Equations (2.1);
 - 7: **end for**
 - 8: *Security Check*:
 - 9: Conduct security check on temperature T_i over all gates;
-

6.2.3.3 Energy HT Indicator

The problem we face in inspecting the characterized temperature profile for HT detection is that the normal temperature profile, or the “golden model”, is not available in the case of wireless systems. It is because the system is often deployed in unknown environments (e.g., wireless sensor network) or has a mobile nature and a high probability of environmental changes (e.g., smart phones). Therefore, the online temperature security inspection cannot be done via simple comparisons.

We solve the problem by defining a HT indicator that represents the temperature inconsistency over gates that are adjacent to each other in the target IC. Our intuition is that the heat transfer process would create spatial correlations in the temperatures of gates that are physically close to each other. Therefore, if we ever observe that there is an abnormally large deviation between the temperatures of two or more adjacent gates, it is an indicator that the energy increase is not likely caused by temperature changes but by malicious energy attacks. This

is based on the assumption that it is computationally impossible for an attacker to emulate the heat transfer model and impose energy attacks following exactly the same pattern.

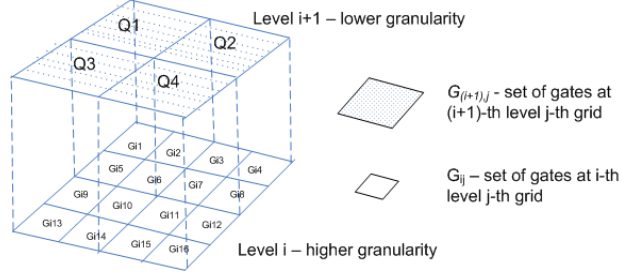


Figure 6.5: Principal component analysis (PCA) model to define the hardware Trojan indicator.

We define the HT indicator using the principal component analysis (PCA) models [24] that were originally used for modeling spatial correlations in IC process variations. As shown in Figure 6.5, we group the gates into multiple grids at different levels in order to capture the inconsistency of the temperatures between various boundaries of adjacent gates. At each specific level, we define the HT indicator as the average standard deviation, over all grids, of temperatures among all gates within each grid. In particular, at the i -th level, the HT indicator H_i can be calculated as the following:

$$H_i = \left(\sum_{j=1}^{N_i} \text{stddev}(G_{ij}) \right) / N_i \quad (6.1)$$

where N_i is the number of grids at level i , and G_{ij} is the set of gates in the j -th grid at level i . We use H_i to evaluate the temperature deviation over adjacent gates at different granularities. Depending on the sizes and physical properties of the circuit under test, different levels of H_i plays different roles in the final evaluation of the temperature deviations. Therefore, we define the following

weighted function for the overall HT indicator:

$$H = \left(\sum_{i=1}^L w_i H_i \right) / L \quad (6.2)$$

where L is the number of levels that we divide using the PCA model, and w_i is the weight factor at level i concerning the physical properties of the circuit.

6.3 Online Security Attack and Defense

In Chapter 4, we have discussed the attacks and detection methods regarding the one-gate HT model (as shown in Figure 4.1) in the post-silicon stage. In this section, we focus on analyzing the consequences of deploying such a HT in remote wireless systems, such as wireless sensor networks, where there is no physical access to the system. In particular, we discuss the possible mechanisms that an attacker may leverage to conduct online attacks that could bypass the security checks via remote power profiling. Also, we propose the corresponding defense schemes to ensure the security of the wireless system. The discussion of the attacks and defense techniques also apply to the power profiling process for the energy HT detection, as discussed in Chapter 6.2.

Due to the extremely limited overhead and rare activation of the HT trigger, it is often the case that the HT triggers cannot be detected in the post-silicon stage. Therefore, it is essential to detect the HT attacks in-field after they are activated. During the system operation, the attacker must trigger and power up the malicious circuitry in order to activate the HT attack. Once the malicious circuitry is activated, one can easily detect the abnormality, since the malicious circuitry often contains a large number of gates as well as complicated structures in order to accomplish advanced security attacks, such as leaking confidential information

or making the device malfunction. However, it is still possible for the attacker to manipulate the behavior of the wireless system to further bypass the online security checks after the activation of malicious circuitry.

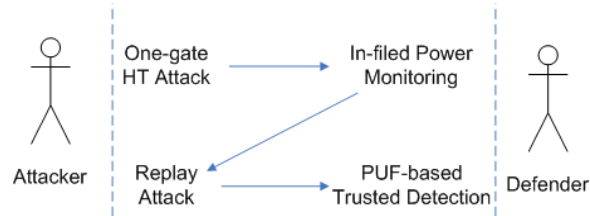


Figure 6.6: Online attack and defense model.

6.3.1 Attack and Defense Models

Figure 6.6 demonstrates the attack and defense models after the HT-embedded wireless system enters the operation mode. The attacker would activate the malicious circuitry by inducing the system to run an application that satisfies the rare activation condition. Then, as a defense, the defender can sample and monitor the power profile of the remote wireless system and observe the variations that may be caused by the activation of the malicious circuitry. However, it is possible for the attacker to conduct a more advanced attack, namely replay attack, that tricks the online monitoring scheme with outdated power profiles that do not reflect any variations caused by the malicious circuitry. In order to resolve the replay attack, we develop a physically unclonable function (PUF)-based trusted HT detection technique that authenticates each sample of power profile with specific time and location information and, therefore, any attempts to report replayed power profiles would be detected.

Algorithm 8 In-field power monitoring for detecting one-gate HT attack during system operation.

- 1: Designer implements a test trigger into the design that monitors the activity of the privileged area for security attacks;
 - 2: Attacker embeds the HT gate and the malicious circuitry in the wireless system;
 - 3: The wireless system passes post-silicon test, since the malicious circuitry powered off;
 - 4: The wireless system starts operating;
 - 5: Defender collects power profile during the initialization period as the baseline profile;
 - 6: The wireless system operates normally for a period of time t ;
 - 7: Attacker triggers the one-gate HT and activates the malicious circuitry;
 - 8: The test trigger activates the power meter to measure the power profile and reports it to defender;
 - 9: Defender observes abnormal variation in power profile caused by the activated malicious circuitry;
 - 10: Defender terminates the operation of the wireless system that is under HT attack;
-

6.3.2 Online Detection by In-field Power Measurements

During the operation mode of the wireless system when the malicious circuitry can be possibly activated, we employ in-field power metering techniques [39][58][79] to keep track of the power profile. The micro power meter that is integrated into the wireless system is capable of measuring the real-time power profiles and reporting to the remote administrator for further assessment. In order to reduce the cost of conducting such power measurements, we employ a test trigger gate to monitor the activity of the privileged area in the design. The test trigger is activated and the power meter starts measuring the power profile only when the privileged area is suspected to be attacked. When this situation occurs, the power profile data is sent to the administrator for further analysis to confirm the existence of HT attacks.

Algorithm 8 describes the detailed procedure of in-field power measurements for HT detection. The power meter in the wireless system first collects a set of power samples at the beginning of the system operation, which can serve as a baseline for the normal power profile. Once the test trigger gate is activated, the administrator would be able to collect instant power profiles from the power meter and determine whether there is any HT attack being conducted. We consider the variation of power profile as an indicator of HT attack, since the activated malicious circuitry would consume a relatively large amount of power and cause a surge in the leakage power profile compared to the baseline.

6.3.3 Online Replay Attack

We note that the straightforward HT detection technique via in-field power profiling can still be bypassed by the attacker. For example, it is possible that the attacker conducts replay attack [80], in which a set of normal leakage power

profiles are pre-recorded and reported to the monitoring system constantly. Algorithm 9 illustrates a typical case of replay attack, which results in the failure of detection. Note that the attacker may start or terminate the replay procedure at any time, or vary the power profile considering environmental factors and the workload on the wireless system to generate more trustworthy power reports.

Algorithm 9 Online replay attack that bypasses the in-field power sampling approach.

- 1: Attacker embeds the HT gate and the malicious circuitry in the wireless system;
 - 2: The wireless system passes post-silicon test as the malicious circuitry is powered off;
 - 3: Defender enables the online in-field power profiling process;
 - 4: The wireless system starts operating;
 - 5: The wireless system operates normally for a period of time t ;
 - 6: Attacker records the power profiles f_t within the time period t ;
 - 7: The wireless system starts responding with power profiles f_t anytime when there is a profiling request;
 - 8: Attacker triggers the one-gate HT and activates the malicious circuitry;
 - 9: Defender observes normal power profile f_t constantly;
 - 10: The wireless system is compromised by the activated malicious circuitry;
-

6.3.4 Trusted Detection Using Physically Unclonable Functions

Considering the possible online replay attack, we develop a trusted HT detection approach based on the use of physically unclonable functions (PUFs) [34][60]. A PUF is a specially designed circuitry in which the prediction of output signals from known inputs is computationally infeasible, unless one has physical access

to the PUF. Figure 6.7 shows a sample PUF design [61], where the complexity of determining the output vectors grows exponentially as the increase of the number of levels in the design. Since there is a huge difference between the simulation time (e.g., in the magnitude of nanoseconds) and the prediction time (e.g., in the magnitude of seconds) for obtaining the output signals, PUFs can be used as a security key for identity authentications in many applications [12][61][69][99].

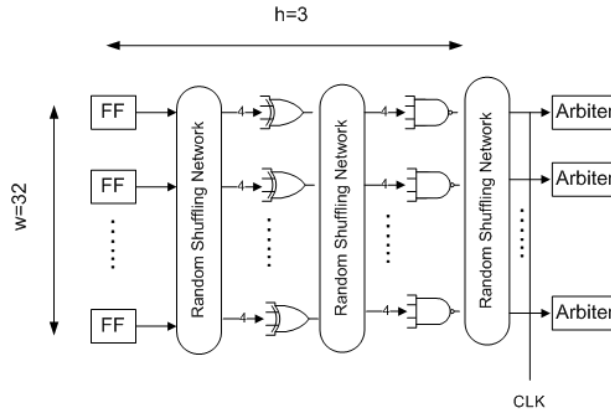


Figure 6.7: PUF architecture [61].

However, a direct use of PUF with randomly generated challenge bits cannot resolve the replay attack, since we must ensure that the collected power profile in the monitoring process are those generated from the specific sensor at the specific time frame. This requires us to associate each sample with both time and location information and take into consideration of the (time, location, power) triplets at the checking time. For the time stamp, we leverage the secure navigation signals that can be received synchronously from integrated GPS systems [52] at both the remote wireless system and the local administration site. For the identification of sensors, we leverage the fact that each PUF exhibits different delay characteristics due to process variation. Consequently, the output signals are different for the PUFs on different sensors, since they are highly dependent on the accumulated

delay at each level of the design. Figure 6.8 shows our design of the PUF system for resolving the replay attack toward the remote wireless system.

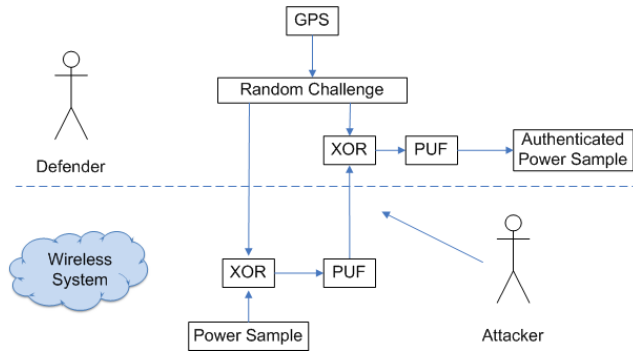


Figure 6.8: PUF-based trusted HT detection.

6.4 Wireless Security Experimental Results

We evaluate our HT attack and detection models on a set of ISCAS and ITC benchmarks. We model the process variations of the designs following the Gaussian distribution presented in [7] and the quad-tree model presented in [24].

6.4.1 Effectiveness of Energy Hardware Trojan Attack

We evaluate the effectiveness of the energy HT attack by quantifying the energy increase caused by the energy HT in the circuit under attack. In particular, we insert a HT trigger that selects ABB voltages up to 1.0V and evaluate the energy increase. Figure 6.9 shows the energy increase due to the attack on a set of ISCAS benchmarks. We observe that the energy consumption grows exponentially with the linear increase of the ABB voltage, creating huge impacts on the circuit under attack.

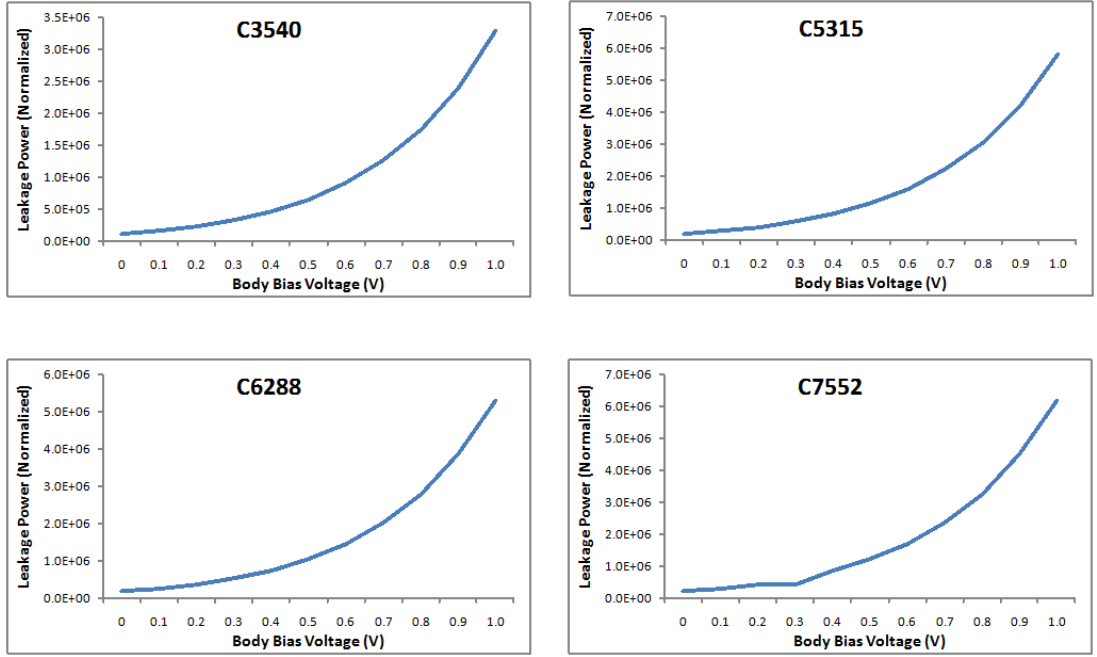


Figure 6.9: Leakage power increase due to forward ABB attack.

6.4.2 Effectiveness of Temperature-aware Hardware Trojan Detection

We evaluate the effectiveness of our temperature-aware HT detection approach from two aspects. Firstly, we evaluate the accuracy of the gate-level temperature characterization, which is an indicator of how accurate we can capture the abnormal temperature variations. Then, we evaluate the HT detection approach by comparing the HT indicator values in two cases where HTs are present and where there are no HTs, in order to determine the false positives and false negatives in HT detection.

6.4.2.1 Accuracy of Gate-level Temperature Characterization

We evaluate the accuracy of the gate-level temperature characterization by comparing the characterized temperatures and the actual temperatures and quan-

tifying the average characterization errors. Figure 6.10 shows the distribution of the characterization errors for each gate in a set of ISCAS benchmarks. The results indicate that characterization errors of all gates are controlled within the 2% mark except for very few outliers gates. Also, the accuracy does not decrease with the increase of the circuit size, indicating the scalability of our detection approach.

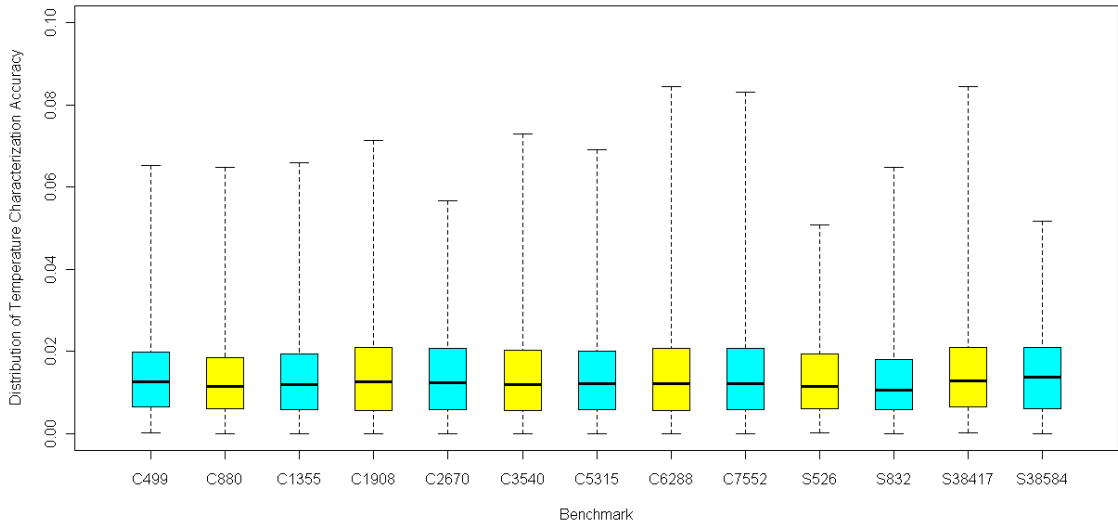


Figure 6.10: Accuracy of temperature characterization.

6.4.2.2 Effectiveness of Detection Using HT Indicator

In order to evaluate the effectiveness of HT detection, we characterize the gate-level temperature profiles in two cases where there are no energy attacks (i.e., HT-free) and where there are forward ABB-based energy HTs embedded and triggered in the target circuit (i.e., HT-present). For each case, we calculate the value of the HT indicator as defined in Chapter 6.2.3 to observe the difference between the two cases. In our simulation, we use 3 levels of grids in the PCA model (i.e., $L =$

3) and use evenly assigned weight factors for the w_i (i.e., $w_i = 1/3, i = 1, 2, 3$) in calculating H . Figure 6.11 shows our evaluation results of the HT indicator H in both the HT-present and HT-free cases. There is an obvious and large difference between the HT indicators in the two cases. In the HT-present case, the HT indicator is significantly larger than that of the HT-free case. Therefore, the HT indicator is an effective metric for differentiating the two cases, which provides us with zero false positives and zero false negatives in the detection of energy HTs.

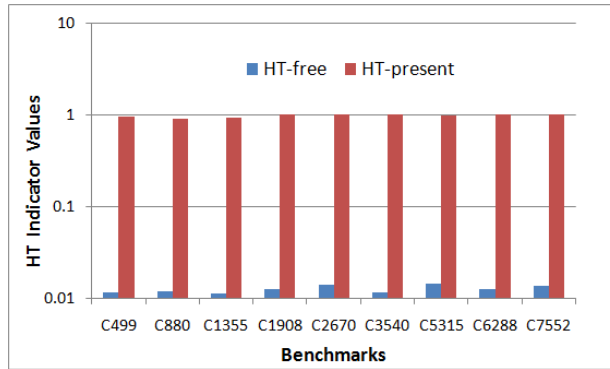


Figure 6.11: Energy HT detection results: HT indicator values in HT-free and HT-present cases.

6.4.3 Effectiveness of PUF-based Online In-field Detection

In order to evaluate the PUF-based in-field HT detection method, we simulate the implemented PUF design using random challenge bits and observe for the randomness of the output signals. Our idea is that if the output signals are random (i.e., in the optimal case, with 50% probability being 1 and 50% being 0), the prediction attempt within any reasonable amount of time will fail, under the consideration that the complexity of prediction grows exponentially with the number of output pins. Figure 6.12 shows our simulation results, where the probabilities of being 1 for many outputs are close to 50%.

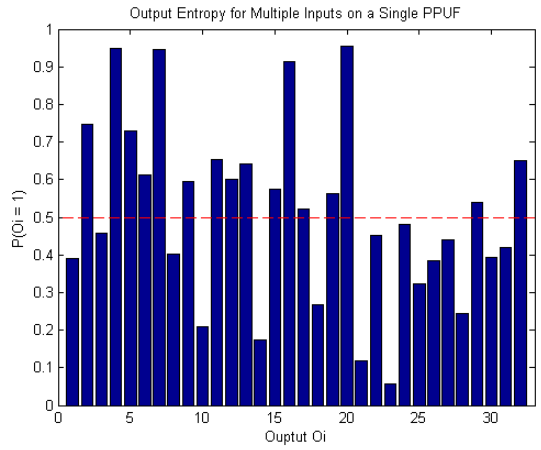


Figure 6.12: The probability of output bit O_i being 1 in the PUF ($w=32, h=3$) following the architecture in Figure 6.7.

CHAPTER 7

Concluding Remarks

We have developed a group of techniques for addressing the security concerns in IC systems arising from the ever increasing trend of transistor scaling and IC outsourcing. The foundation of our techniques is the consistency-based analysis that identifies the abnormal behavior of malicious IC components.

We began the study with a non-intrusive gate-level characterization approach that identifies the gate-level IC properties, such as power and delay. In order to achieve this goal, we employed IC control techniques, such as input vector control, test point insertion, and thermal conditioning, which posed additional and independent variations to the overall IC side channels. In this way, we were able to create linearly independent equations concerning the global IC properties and those of the individual gates. Furthermore, we ensured the scalability of the proposed approach by leveraging a group of scalability techniques that partition the IC into smaller segments.

After obtaining the GLC results in various segments, we conducted consistency analysis and captured the inconsistent behavior of the malicious components. Based on the consistency analysis, we demonstrated a group of system security applications, including hardware Trojan detection and diagnosis, IC metering and digital rights management, and wireless security techniques. We showed that the consistency-based analysis provides an efficient and scalable solution for various IC system security scenarios.

However, we acknowledge that it is likely for a professional attacker to conduct more advanced attack after learning the details of the defense techniques. This observation is in general true in most of the security attack/defense scenarios, which requires that the security primitives must evolve constantly with the emergence of new attacks. It motivates us to continue with our hardware security research efforts in order to achieve a secure and reliable market for the IC industry. Also, more importantly, our goal in this hardware security research is not to provide a one-shot perfect security solution, but to increase the difficulty and cost for an attacker to implement an attack that could cause damage or concerns to the target IC systems.

REFERENCES

- [1] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar. Trojan detection using IC fingerprinting. In *IEEE Symposium on Security and Privacy (S&P)*, pages 296–310, 2007.
- [2] Y. Alkabani and F. Koushanfar. Active hardware metering for intellectual property protection and security. In *USENIX Security Symposium*, pages 20:1–20:16, 2007.
- [3] Y. Alkabani and F. Koushanfar. Extended abstract: Designer’s hardware trojan horse. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 82–83, 2008.
- [4] Y. Alkabani, F. Koushanfar, N. Kiyavash, and M. Potkonjak. Trusted integrated circuits: A nondestructive hidden characteristics extraction approach. In *International Workshop on Information Hiding (IH)*, pages 102–117, 2008.
- [5] Y. Alkabani, F. Koushanfar, and M. Potkonjak. Remote activation of ICs for piracy prevention and digital right management. In *International Conference on Computer-Aided Design (ICCAD)*, pages 674–677, 2007.
- [6] Y. Alkabani, T. Massey, F. Koushanfar, and M. Potkonjak. Input vector control for post-silicon leakage current minimization in the presence of manufacturing variability. In *Design Automation Conference (DAC)*, pages 606–609, 2008.
- [7] A. Asenov. Random dopant induced threshold voltage lowering and fluctuations in sub-0.1 μm MOSFET’s: A 3-D “atomistic” simulation study. *IEEE Transactions on Electron Devices*, 45(12):2505–2513, 1998.
- [8] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An on-demand secure routing protocol resilient to byzantine failures. In *ACM workshop on Wireless security (WiSe)*, pages 21–30, 2002.
- [9] A. Baba and S. Mitra. Testing for transistor aging. In *VLSI Test Symposium (VTS)*, pages 215–220, 2009.
- [10] M. Banga and M. Hsiao. A region based approach for the identification of hardware Trojans. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 40–47, 2008.

- [11] M. Banga and M. Hsiao. VITAMIN: Voltage inversion technique to ascertain malicious insertions in ICs. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 104–107, 2009.
- [12] N. Beckmann and M. Potkonjak. Hardware-based public-key cryptography with public physically unclonable functions. In *International Workshop on Information Hiding (IH)*, pages 206–220, 2009.
- [13] R. Bellman. On the approximation of curves by line segments using dynamic programming. *Communications of the ACM*, 4(6):284, 1961.
- [14] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, and V. De. Parameter variations and impact on circuits and microarchitecture. In *Design Automation Conference (DAC)*, pages 338–342, 2003.
- [15] F. Brglez, D. Bryan, and K. Kozminski. Combinational profiles of sequential benchmark circuits. In *International Symposium on Circuits and Systems (ISCAS)*, pages 1929–1934, 1989.
- [16] F. Brglez and H. Fujiwara. A neutral netlist of 10 combinational benchmark circuits and a target translator in FORTRAN. In *International Symposium on Circuits and Systems (ISCAS)*, pages 677–692, 1985.
- [17] M. Brown, C. Bazeghi, M. Guthaus, and J. Renau. Measuring and modeling variability using low-cost FPGAs. In *International symposium on Field programmable gate arrays (FPGA)*, pages 286–286, 2009.
- [18] A. Caldwell, H. Choi, A. Kahng, S. Mantik, M. Potkonjak, G. Qu, and J. Wong. Effective iterative techniques for fingerprinting design IP. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(2):208–215, 2006.
- [19] S. Capkun, L. Buttyan, and J. Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 2(1):52–64, 2003.
- [20] S. Chakravarthi, A. Krishnan, V. Reddy, C. Machala, and S. Krishnan. A comprehensive framework for predictive modeling of negative bias temperature instability. In *International Reliability Physics Symposium (IRPS)*, pages 273–282, 2004.
- [21] A. Chao. Estimating the population size for capture-recapture data with unequal catchability. *Biometrics*, 43(4):783–791, 1987.

- [22] T. Chen and S. Naffziger. Comparison of adaptive body bias (ABB) and adaptive supply voltage (ASV) for improving delay and leakage under the presence of process variation. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 11(5):888–899, 2003.
- [23] B. Cheng, S. Roy, A. Brown, C. Millara, and A. Asenov. Evaluation of statistical technology generation LSTP MOSFETs. *Solid-State Electronics*, 53(7):767–772, 2009.
- [24] B. Cline, K. Chopra, D. Blaauw, and Y. Cao. Analysis and modeling of CD variation for statistical static timing. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 60–66, 2006.
- [25] J. Cong, H. Li, S. Lim, T. Shibuya, and D. Xu. Large scale circuit partitioning with loose/stable net removal and signal flow based clustering. In *International Conference on Computer Aided Design (ICCAD)*, pages 441–446, 1997.
- [26] R. Datta, G. Carpenter, K. Nowka, and J. Abraham. A scheme for on-chip timing characterization. In *VLSI Test Symposium (VTS)*, pages 24–29, 2006.
- [27] A. Dharwadkar. *The Independent Set Algorithm*. Createspace, 2011.
- [28] N. Dreger, A. Chandrakasan, and D. Boning. A test-structure to efficiently study threshold-voltage variation in large MOSFET arrays. In *International Symposium on Quality Electronic Design (ISQED)*, pages 281–286, 2007.
- [29] N. Een and N. Sorensson. An extensible SAT-solver. In *International Conferences on Theory and Applications of Satisfiability Testing (SAT)*, pages 333–336, 2004.
- [30] L. Eschenauer and V. Gligor. A key-management scheme for distributed sensor networks. In *ACM Conference on Computer and Communications Security (CCS)*, pages 41–47, 2002.
- [31] M. Farag, L. Lerner, and C. Patterson. Interacting with hardware Trojans over a network. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 69–74, 2012.
- [32] P. Flajolet, D. Gardy, and L. Thimonier. Birthday paradox, coupon collectors, caching algorithms and self-organizing search. *Discrete Applied Mathematics*, 39(3):207–229, 1992.

- [33] P. Friedberg, Y. Cao, J. Cain, R. Wang, J. Rabaey, and C. Spanos. Modeling within-die spatial correlation effects for process-design co-optimization. In *International Symposium on Quality of Electronic Design (ISQED)*, pages 516–521, 2005.
- [34] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas. Silicon physical random functions. In *ACM Conference on Computer and Communications Security (CCS)*, pages 148–160, 2002.
- [35] J. Gregg and T. Chen. Post silicon power/performance optimization in the presence of process variations using individual well-adaptive body biasing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 15(3):366–376, 2007.
- [36] M. Hicks, M. Finnicum, S. King, M. Martin, and J. Smith. Overcoming an untrusted computing base: Detecting and removing malicious hardware automatically. In *IEEE Symposium on Security and Privacy (S&P)*, pages 159–172, 2010.
- [37] T. Huffmire, B. Brotherton, G. Wang, T. Sherwood, R. Kastner, T. Levin, T. Nguyen, and C. Irvine. Moats and drawbridges: An isolation primitive for reconfigurable hardware based systems. In *IEEE Symposium on Security and Privacy (S&P)*, pages 281–295, 2007.
- [38] ITC99 benchmark,
<http://www.cerc.utexas.edu/itc99-benchmarks/bench.html>.
- [39] X. Jiang, P. Dutta, D. Culler, and I. Stoica. Micro power meter for energy monitoring of wireless sensor networks at scale. In *International Conference on Information Processing in Sensor Networks (IPSN)*, pages 186–195, 2007.
- [40] Y. Jin, N. Kupp, and Y. Makris. Experiences in hardware Trojan design and implementation. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 50–57, 2009.
- [41] Y. Jin and Y. Makris. Hardware Trojan detection using path delay fingerprint. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 51–57, 2008.
- [42] A. Kahng, D. Kirovski, S. Mantik, M. Potkonjak, and J. Wong. Copy detection for intellectual property protection of VLSI designs. In *International Conference on Computer-Aided Design (ICCAD)*, pages 600–604, 1999.

- [43] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor. Trustworthy hardware: Identifying and classifying hardware Trojans. *IEEE Computer Magazine*, 43(10):39–46, 2010.
- [44] T. Kemper, Y. Zhang, Z. Bian, and A. Shakouri. Ultrafast temperature profile calculation in IC chips. In *International Workshop on Thermal investigations of ICs*, pages 1–5, 2006.
- [45] S. King, J. Tucek, A. Cozzie, C. Grier, W. Jiang, and Y. Zhou. Designing and implementing malicious hardware. In *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, pages 1–8, 2008.
- [46] F. Koushanfar. Integrated circuits metering for piracy protection and digital rights management: An overview. In *ACM Great Lakes symposium on VLSI (GLSVLSI)*, pages 449–454, 2011.
- [47] F. Koushanfar. Hardware metering: A survey. In *Introduction to Hardware Security and Trust*, pages 103–122. Springer, 2012.
- [48] F. Koushanfar, P. Boufounos, and D. Shamsi. Post-silicon timing characterization by compressed sensing. In *International Conference on Computer-Aided Design (ICCAD)*, pages 185–189, 2008.
- [49] F. Koushanfar and A. Mirhoseini. A unified framework for multimodal submodular integrated circuits Trojan detection. *IEEE Transactions on Information Forensics and Security*, 6(1):162–174, 2011.
- [50] F. Koushanfar and M. Potkonjak. CAD-based security, cryptography, and digital rights management. In *Design Automation Conference (DAC)*, pages 268–269, 2007.
- [51] F. Koushanfar, G. Qu, and M. Potkonjak. Intellectual property metering. In *International Workshop on Information Hiding (IH)*, pages 81–95, 2001.
- [52] M. Kuhn. An asymmetric security mechanism for navigation signals. In *International Workshop on Information Hiding (IH)*, pages 239–252, 2004.
- [53] C. Lamech and J. Plusquellic. Trojan detection based on delay variations measured using a high-precision, low-overhead embedded test structure. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 75–82, 2012.
- [54] L. Lazos and R. Poovendran. SeRLoc: secure range-independent localization for wireless sensor networks. In *ACM workshop on Wireless security (WiSe)*, pages 21–30, 2004.

- [55] P. Lewis and E. Orav. *Simulation Methodology for Statisticians, Operations Analysts, and Engineers (Volume I)*. Chapman & Hall/CRC, 1998.
- [56] J. Li and J. Lach. At-speed delay characterization for IC authentication and Trojan horse detection. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 8–14, 2008.
- [57] M. Liedtke. Google to pay \$90m in “click fraud” case. *Washington Post Magazine*, March 2006.
- [58] M. Malinowski, M. Moskwa, M. Feldmeier, M. Laibowitz, and J. Paradiso. CargoNet: a low-cost micropower sensor node exploiting quasi-passive wakeup for adaptive asynchronous monitoring of exceptional events. In *International Conference on Embedded Networked Sensor Systems (SenSys)*, pages 145–159, 2007.
- [59] D. Markovic, C. Wang, L. Alarcon, T. Liu, and J. Rabaey. Ultralow-power design in near-threshold region. *Proceedings of the IEEE*, 98(2):237–252, 2010.
- [60] S. Meguerdichian and M. Potkonjak. Device aging-based physically unclonable functions. In *Design Automation Conference (DAC)*, pages 288–289, 2011.
- [61] S. Meguerdichian and M. Potkonjak. Matched public PUF: Ultra low energy security platform. In *International Symposium on Low Power Electronics and Design (ISLPED)*, pages 45–50, 2011.
- [62] P. Michaud and Y. Sazeides. ATMI: Analytical model of temperature in microprocessors. In *Annual Workshop on Modeling, Benchmarking and Simulation (MoBS)*, 2007.
- [63] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge, 2005.
- [64] M. Naor and B. Pinkas. Secure and efficient metering. In *Advances in Cryptology – EUROCRYPT*, pages 576–590, 1998.
- [65] S. Narasimhan, X. Wang, D. Du, R. Chakraborty, and S. Bhunia. TeSR: A robust temporal self-referencing approach for hardware Trojan detection. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 71–74, 2011.

- [66] M. Nelson, A. Nahapetian, F. Koushanfar, and M. Potkonjak. SVD-based ghost circuitry detection. In *International Workshop on Information Hiding (IH)*, pages 221–234, 2009.
- [67] NI PXI-4130 Power SMU: <http://sine.ni.com/nips/cds/view/p/lang/en/nid/204239>.
- [68] M. Potkonjak. Synthesis of trustable ICs using untrusted CAD tools. In *Design Automation Conference (DAC)*, pages 633–634, 2010.
- [69] M. Potkonjak, S. Meguerdichian, and J. Wong. Trusted sensors and remote sensing. In *IEEE Sensors*, pages 1104–1107, 2010.
- [70] M. Potkonjak, A. Nahapetian, M. Nelson, and T. Massey. Hardware Trojan horse detection using gate-level characterization. In *Design Automation Conference (DAC)*, pages 688–693, 2009.
- [71] J. Rabaey, A. Chandrakasan, and B. Nikolic. *Digital Integrated Circuits*. Prentice-Hall, 2003.
- [72] R. Rad, X. Wang, M. Tehranipoor, and J. Plusquellic. Power supply signal calibration techniques for improving detection resolution to hardware trojans. In *International Conference on Computer-Aided Design (ICCAD)*, pages 632–639, 2008.
- [73] R. Rajsuman. Iddq testing for CMOS VLSI. *Proceedings of the IEEE*, 88(4):544–568, 2000.
- [74] S. Sabade and D. Walker. Iddx-based test methods: A survey. *ACM Transactions on Design Automation of Electronic Systems*, 9(2):159–198, 2004.
- [75] A. Sadeghi, I. Visconti, and C. Wachsmann. Anonymizer-enabled security and privacy for RFID. In *International Conference on Cryptology and Network Security (CANS)*, pages 134–153, 2009.
- [76] H. Salmani, M. Tehranipoor, and J. Plusquellic. New design strategy for improving hardware Trojan detection and reducing Trojan activation time. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 66–73, 2009.
- [77] sat4j: <http://www.sat4j.org/>.
- [78] Z. Schnabel. The estimation of total fish population of a lake. *The American Mathematical Monthly*, 45(6):348–352, 1938.

- [79] T. Stathopoulos, D. McIntire, and W.f Kaiser. The energy endoscope: Real-time detailed energy accounting for wireless sensor nodes. In *International Conference on Information Processing in Sensor Networks (IPSN)*, pages 383–394, 2008.
- [80] P. Syverson. A taxonomy of replay attacks. In *Computer Security Foundations Workshop (CSFW)*, pages 187–191, 1994.
- [81] M. Tehranipoor and F. Koushanfar. A survey of hardware Trojan taxonomy and detection. *IEEE Design Test of Computers*, 27(1):10–25, 2010.
- [82] A. Vahdatpour, S. Meguerdichian, and M. Potkonjak. A gate level sensor network for integrated circuits temperature monitoring. In *IEEE Sensors*, pages 652–655, 2010.
- [83] V. Vazirani. *Approximation Algorithms*. Springer, 2001.
- [84] A. Waksman and S. Sethumadhavan. Silencing hardware backdoors. In *IEEE Symposium on Security and Privacy (S&P)*, pages 49–63, 2011.
- [85] X. Wang, M. Tehranipoor, and R. Datta. A novel architecture for on-chip path delay measurement. In *International Test Conference (ITC)*, pages 1–10, 2009.
- [86] S. Wei, F. Koushanfar, and M. Potkonjak. Integrated circuit digital rights management techniques using physical level characterization. In *ACM workshop on Digital rights management (DRM)*, pages 3–14, 2011.
- [87] S. Wei, K. Li, F. Koushanfar, and M. Potkonjak. Provably complete hardware Trojan detection using test point insertion. In *International Conference on Computer-Aided Design (ICCAD)*, pages 569–576, 2012.
- [88] S. Wei, S. Meguerdichian, and M. Potkonjak. Gate-level characterization: Foundations and hardware security applications. In *Design Automation Conference (DAC)*, pages 222–227, 2010.
- [89] S. Wei, S. Meguerdichian, and M. Potkonjak. Malicious circuitry detection using thermal conditioning. *IEEE Transactions on Information Forensics and Security*, 6(3):1136–1145, 2011.
- [90] S. Wei, A. Nahapetian, M. Nelson, F Koushanfar, and M. Potkonjak. Gate characterization using singular value decomposition: Foundations and applications. *IEEE Transactions on Information Forensics and Security*, 7(2):765–773, 2012.

- [91] S. Wei, A. Nahapetian, and M. Potkonjak. Robust passive hardware metering. In *International Conference on Computer-Aided Design (ICCAD)*, pages 802–809, 2011.
- [92] S. Wei and M. Potkonjak. Scalable segmentation-based malicious circuitry detection and diagnosis. In *International Conference on Computer-Aided Design (ICCAD)*, pages 483–486, 2010.
- [93] S. Wei and M. Potkonjak. Integrated circuit security techniques using variable supply voltage. In *Design Automation Conference (DAC)*, pages 248–253, 2011.
- [94] S. Wei and M. Potkonjak. Scalable consistency-based hardware Trojan detection and diagnosis. In *International Conference on Network and System Security (NSS)*, pages 176–183, 2011.
- [95] S. Wei and M. Potkonjak. Hardware Trojan horse benchmark via optimal creation and placement of malicious circuitry. In *Design Automation Conference (DAC)*, pages 90–95, 2012.
- [96] S. Wei and M. Potkonjak. Scalable hardware Trojan diagnosis. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 20(6):1049–1057, 2012.
- [97] S. Wei and M. Potkonjak. Wireless security techniques for coordinated manufacturing and on-line hardware trojan detection. In *ACM conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, pages 161–172, 2012.
- [98] S. Wei and M. Potkonjak. The undetectable and unprovable hardware trojan horse. In *Design Automation Conference (DAC)*, Article No. 144, 2013.
- [99] J. Wendt and M. Potkonjak. Nanotechnology-based trusted remote sensing. In *IEEE Sensors*, pages 1213–1216, 2011.
- [100] F. Wolff, C. Papachristou, S. Bhunia, and R. Chakraborty. Towards Trojan-free trusted ICs: Problem analysis and detection scheme. In *Design, Automation and Test in Europe (DATE)*, pages 1362–1365, 2008.
- [101] E. Wong, E. Young, and W. Mak. Clustering based acyclic multi-way partitioning. In *ACM Great Lakes symposium on VLSI (GLSVLSI)*, pages 203–206, 2003.

- [102] H. Xu, R. Vemuri, and W. Jone. Temporal and spatial idleness exploitation for optimal-grained leakage control. In *International Conference on Computer-Aided Design (ICCAD)*, pages 468–473, 2009.
- [103] L. Yuan and G. Qu. Information hiding in finite state machine. In *International Workshop on Information Hiding (IH)*, pages 340–354, 2004.
- [104] L. Yuan and G. Qu. A combined gate replacement and input vector control approach for leakage current reduction. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 14(2):173–182, 2006.
- [105] J. Zhang, H. Yu, and Q. Xu. HTOutlier: Hardware Trojan detection with side-channel signature outlier identification. In *International Symposium on Hardware-Oriented Security and Trust (HOST)*, pages 55–58, 2012.
- [106] L. Zhang, N. Howard, V. Gumaste, A. Poddar, and L. Nguyen. Thermal characterization of stacked-die packages. In *Semiconductor Thermal Measurement and Management Symposium*, pages 55–63, 2004.
- [107] W. Zhang, X. Li, and R. Rutenbar. Bayesian virtual probe: Minimizing variation characterization cost for nanoscale ic technologies via bayesian inference. In *Design Automation Conference (DAC)*, pages 262–267, 2010.