SECURE AND RELIABLE VLSI DESIGNS

By

G. SUMATHI

(Enrollment No: ENGG02201204007)

INDIRA GANDHI CENTRE FOR ATOMIC RESEARCH
KALPAKKAM

A thesis submitted to the

Board of Studies in Engineering Sciences

In partial fulfillment of requirements

For the Degree of

DOCTOR OF PHILOSOPHY

of

HOMI BHABHA NATIONAL INSTITUTE

NOVEMBER, 2018

HOMI BHABHA NATIONAL INSTITUTE

Recommendations of the Viva Voce Board

As members of the Viva Voce Board, we certify that we have read the dissertation prepared by **Smt. G. Sumathi** entitled "**Secure and Reliable VLSI Designs**" and recommend that it may be accepted as fulfilling the dissertation requirement for the Degree of Doctor of Philosophy.

Chairman: **Dr. B. K. Panigrahi** Date: 26/11/18

Guide/Convener: **Dr. Anish Kumar** Date: 26.11.18

External Examiner: Date: 26/11/18

Member 1: **Dr. V. S. Srinivasan** Date: 26/11/18

Member 2: **Dr. B. Purna Chandra Rao** Date: 26/11/2018

Technology Advisor: **Dr. D. Thirugnana Murthy** Date: 26.11.18

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertations to HBNI.

I hereby certify that I have read this dissertation prepared under my direction and recommend that it may be accepted as fulfilling the dissertation requirement.

Date: 26.11.18

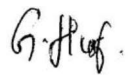
Place: Indira Gandhi Centre for Atomic Research (IGCAR)
Kalpakkam

Dr. Anish Kumar
(Guide)

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at Homi Bhabha National Institute (HBNI) and is deposited in the library to be made available to borrowers under rules of the HBNI.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgement of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the Competent Authority of HBNI when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

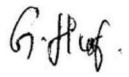


Date : 26.11.18 **(G. SUMATHI)**

Place : Kalpakkam

DECLARATION

I, hereby declare that the investigation presented in the thesis has been carried out by me. The work is original and has not been submitted earlier as a whole or in part for a degree / diploma at this or any other Institution / University.



Date : 26.11.18 **(G. SUMATHI)**

Place : Kalpakkam

LIST OF PUBLICATIONS ARISING FROM THE THESIS

JOURNALS

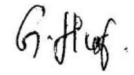
1) **G. Sumathi**, L. Srivani, D. Thirugnana Murthy, Anish Kumar, and K. Madhusoodanan, "Hardware Obfuscation using Different Obfuscation Cell Structures for PLDs," Cryptology and Information Security, Vol. 15, pp. 143-157, 2017.
2) **G. Sumathi**, L. Srivani, D. Thirugnana Murthy, K. Madhusoodanan, and S.A.V. Satya Murty, "A Review on HT attacks in PLD and ASIC designs with Potential Defense Solutions," IETE Technical Review, Vol. 34, pp. 1-14, 2017.
3) **G. Sumathi**, L. Srivani, D. Thirugnana Murthy, and K. Madhusoodanan, "Multi -Corner Timing Analysis based Reliability Estimation of PUFs Implemented in FPGAs," International Journal of Emerging Technology in Computer Science and Electronics, Vol. 23, No. 5, pp. 195-200, 2016.
4) **G. Sumathi**, L. Srivani, D. Thirugnana Murthy, and Anish Kumar, "Hardware Netlist Obfuscation Technique for PLDs with Obfuscation Metric Calculations," in IEEE Trans. VLSI (manuscript under preparation).
5) **G. Sumathi**, L. Srivani, D. Thirugnana Murthy, and Anish Kumar, "Logic Obfuscation Technique for Sequential Designs and Obfuscation Tool Development," in IEEE Trans. CAD (manuscript under preparation).

CONFERENCES/ SYMPOSIUMS

1) **G. Sumathi**, L. Srivani, D. Thirugnana Murthy, K. Madhusoodanan, and S. A. V. Satya Murty, "IP Attacks and Protection Taxonomy: Logic Obfuscation as Solution," in Proc. 9th DAE-VIE Symposium on Emerging Trends in I&C and Computer Systems, IGCAR, Jun. 2016.
2) **G. Sumathi**, L. Srivani, D. Thirugnana Murthy, Anish Kumar, K. Madhusoodanan, and S. A. V. Satya Murty, "Structural Modification based Netlist Obfuscation Technique for

PLDs," in Proc. IEEE International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), Chennai, pp. 1418-1423, Mar. 2016.

3) **G. Sumathi**, L. Srivani, D. Thirugnana Murthy, Anish Kumar, K. Madhusoodanan, and S.A.V. Satya Murty, "Anti-Trojan Design Guidelines for PLD and ASIC Designs in Safety Systems," in Proc. SRESA 2nd National Conference on Reliability and Safety Engineering, Chennai, Oct. 2015.
4) **G. Sumathi**, L. Srivani, D. Thirugnana Murthy, N. Murali, S. A. V. Satya Murty and T. Jayakumar, "DSDPC: Delay Signatures at Different Process Corners based Hardware Trojan Detection Technique for FPGAs," in Proc. IEEE International Conference on Robotics, Automation, Control and Embedded Systems (RACE), Chennai, Feb. 2015.
5) L. Srivani, **G. Sumathi**, and D. Thirugnana Murthy, "Hardware Trojan Horse", in Proc. National Symposium on Nuclear Instrumentation (NSNI), BARC, Mumbai, Nov. 2013.



(G. SUMATHI)

“TO MY MOTHER”

ACKNOWLEDGEMENT

At the outset, I would like to thank my research supervisors, Dr. Anish Kumar, Head, UMS, NDED, MMG and Dr. T. Jayakumar, former Director, MMG, my technology advisor, Dr. D. Thirugnana Murthy, Head, EID, EIG and Smt. L. Srivani, Head, ESS, EID, EIG, IGCAR for their invariable motivation, guidance, encouragement and fortitude during the course of my research tenure at IGCAR. I would like to express my sincere gratitude to all of them for their supervision in many aspects including literature survey, investigational work, preparation of research articles and completion of thesis.

I convey my heartiest appreciation to my doctoral committee Chairmen, Dr. B. K. Panigrahi & Dr. M. Sai Baba and Committee members, Dr. V. S. Srinivasan & Dr. B. Purna Chandra Rao for their support and encouragement during my research work.

My sincere gratitude goes to Dr. Arun Kumar Bhaduri, Director, IGCAR and Dr. S. A. V. Satya Murty and Dr. P. R. Vasudeva Rao, former Directors, IGCAR for providing excellent environment to carry out research work. I would like to thank Shri K. Madhusoodanan, former Director, EIG and Shri N. Murali, former Associate Director, ICG, EIRSG for motivation and support.

My sincere thanks are owed to members of EID and RTSD for providing me technical support and for enriching my ideas. I express my gratitude to IGCAR and fellow colleagues Satya Rajesh Medidi, Anindya Bhattacharya, T. S. Nidhin, V. Rajesh, N. Anil and C. H. Deepak for their timely help, useful technical discussions and suggestions.

I sincerely thank Prof. V. Kamakoti, IITM, Chennai, who has sensitized the research avenue on hardware Trojans and the RISE lab, IITM who has given the permission to make use of Formal Verification tools.

I am grateful for the financial support provided by research fellowship scheme from Indira Gandhi Centre for Atomic Research, Department of Atomic Energy for the duration of the work.

I would like to thank all my friends for helping and supporting me unconditionally on this long and sometimes difficult journey and for being with me whenever I needed. I can't list all the names here, but you are always on my mind. The memories and stories from time we spent together would fill a book thicker than this one.

Last but not the least, my deepest gratitude is to my family for their support which brings belief and hope into my life; to my beloved parents Shri N. Gokulanathan and Smt. G. Rajeswari, my sister G. Revathi, my brother-in-law G. Sathish Kumar and my brother G. Srinivasan. They encouraged and inspired me every time, teaching to never give up no matter how tough the situation is. Last, but most importantly, I would like to express my heartful thanks to my husband, R. Gopi, for his patience and endless love.

Apart from all the people mentioned in this acknowledgment, there are many others who have helped me in various ways during the course of this thesis work. My sincere thanks to all those I may have forgotten to mention.

Thank you everyone.

TABLE OF CONTENTS

SYNOPSIS

Over the past few decades, programmable logic devices (PLD) such as complex PLDs (CPLD) and field programmable gate arrays (FPGA) have been extensively used as the basic building modules in most digital systems due to their features such as high density, field re-programmability and faster time-to-market. In addition, usage of PLDs in a design reduces discrete integrated circuits (IC) population and the associated interconnections on the printed circuit board. This, in turn, increases the reliability of PLD-based systems. However, when features such as unit cost, speed and power are considered, application specific integrated circuits (ASIC) are the most suitable devices. They also address the problem of fast obsolescence associated with PLDs. To develop digital systems using either PLD or ASIC devices, the designer relies more on the outsourced intellectual property (IP) cores, commercial electronic design automation (EDA) tools and offshore fabrication services, which in turn introduce more third-party involvement in the design. Due to these high third-party involvements, hardware description language (HDL) code, IP core and the chip design becomes highly vulnerable to various design security threats. These include: (1) an attacker may steal and claim ownership of the design, resulting in “piracy attack”; (2) an untrusted IP user may perform “duplication or cloning attack” whereas IC foundry may perform “IC overbuilding attack”; (3) an attacker may “reverse engineer (RE)” the IP or IC to understand the functionality and (4) insertion of malicious circuits, also known as “hardware Trojans (HT)” to change the functionality, reduce reliability, denial of service and destroy the chip. In most of the scenarios, IP vendors and the chip or system designers are the key victims of these attacks. Therefore, it is very important to

implement the suitable anti-tamper solutions to avoid such attacks and to ensure that the IP or IC being in use perform only the intended functionality.

To detect HTs inserted at field operating condition of PLDs, but not on one time programmable FPGAs, delay signature based HT detection mechanism is proposed where field extracted delay profile of the netlist is compared with the stored profile at regular intervals to identify tampering. Basic assumption of the work is that results of EDA tools are consistent in the ideal condition; hence, any deviation in delay signature clearly reveals the tampering of design. Also, to effectively reduce false negatives and improve the efficiency, the delay signatures at different process corners technique is proposed. The results of the investigation are as expected that the difference in delay values between with and without HT circuit is increased from slow to fast process corner which enhances the HT detection efficiency.

It is explicit that deriving single detection mechanism for all HT attacks is practically infeasible. Hence, this thesis also focuses on hardware obfuscation techniques which ensure a certain level of design security by preventing against stealing of original design by analyzing and rebuilding during RE. Thus, it increases RE complexity of HDL or IP or IC, in turn, avoids insertion of successful and hard-to-detect HTs. The security analysis of various obfuscation techniques and their applications to the ASIC technology has been published over the last decade. With the expansion of the use of PLDs beyond commercial markets to internet of things, avionics, defense and nuclear applications, designs in PLDs take on additional aspects of safety and national security. Besides, PLD-based critical applications attempt to preserve their indigenous designs as IPs to handle fast obsolescence of PLDs and to upgrade with technology. However, the importance of logic obfuscation technique to PLD-based safety critical designs is not discussed yet. In this thesis, the structural modification based hardware obfuscation

technique is experimented to PLDs using benchmark circuits. The proposed hardware obfuscation method consists of two major implementations such as (1) Finite state machine (FSM) based key mechanism and (2) Obfuscation cell (OC) based lock mechanism. In general, the key mechanism is implemented as separate FSM logic with varying obfuscation states, i.e. varying initialization sequence lengths. The lock mechanism is implemented at selective non-critical nets of target circuit using OC structures. There are two modes of operation in FSM such as obfuscated and functional modes. The control signal, derived from each state of FSM, is stitched with obfuscation nets using OCs. The circuit stays in obfuscated mode upon global reset (i.e. initial state). From initial state, a set of inputs or initialization key sequence must be applied to drive the circuit to functional mode. This enabling key sequence acts as an authentication sequence allowing the circuit to enter into functional mode; otherwise, circuit stays at obfuscated mode and does not perform the required functionality. In particular, the following major contributions are made in this work. (1) Most of obfuscation techniques are implemented using single obfuscation cell structures throughout the design. This, in turn, aids the adversary to find-out the glue logic during image processing-based RE. Hence, it is proposed to use different obfuscation cell structures. (2) To avoid repeatability in netlist obfuscation, it is demonstrated to select obfuscation cells in random for each iteration of logic obfuscation process. (3) It is proposed to randomize the control value generation using true random number generator, so that the output values in obfuscation mode will not be identical in each power on condition. (4) Insertion of obfuscation cells at high fan-out (HF) nets may leave a hint to adversary about logic obfuscation. To avoid this scenario, the inputs of HF-driver modules instead of HF nets are used to insert obfuscation cells. (5) To validate the proposed claims, the complete automation and integration of the logic obfuscation approach with regular FPGA design flow is performed using

"perl" and "tcl" scripting languages. (6) To evaluate the improvement in RE complexity, the novel obfuscation metric is proposed to quantify the percentage of modification introduced by obfuscation techniques. The results are presented from validation of the modeled obfuscation tool.

In addition to proposed HT detection and logic obfuscation mechanisms, implementation of device dependent secret key generation module in PLDs is one of the best solutions to avoid side-channel attacks and various invasive attacks. The physically unclonable functions (PUF) are widely used to address them. A PUF is a challenge-response module which generates unique, reliable and tamper-proof signatures for a given IC based on process variations inherent in the IC manufacturing process, i.e. the output response is totally random and unpredictable. As PUFs are used to generate secret keys, it is necessary to ensure that PUF generates the same response (i.e. 100% reliability) to a given challenge at different instances of time and under different environmental conditions. Reliability is a measure of stability of PUF response bits to a given challenge at different times and operating conditions. Intra hamming distance (HD) is the HD (the number of bits that are different in two strings) between responses generated at different temperatures or voltages and operating temperature or voltage. The hamming distance and reliability estimation calculation of PUFs are performed using multi-corner timing analysis, i.e. by changing the operating voltage and temperature values by simulation software. The proposed method enables the designer to perform simulation based reliability estimation at design stage and this value can support reliability values calculated during experiments. Based on the results, necessary error correction mechanisms shall be applied on PUF output bits to reconstruct exactly the same keys each time under all operating conditions.

The present thesis attempts to understand various hardware security threats in very large scale integration device based safety critical applications to identify key areas of improvement and to suggest feasible solutions for the same. The thesis discusses on development of delay profile based HT detection technique with improved HT detection efficiency and multi-corner timing analysis based reliability calculation of PUFs. As hardware protection mechanism, thesis also briefs on the experiments carried on hardware obfuscation based design security solution for PLDs. Further, it also covers the obfuscation metric calculations, validation of the proposed technique and automation & integration of hardware obfuscation technique using scripting languages.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ADC	Analog-to-Digital Converter
AES	Advanced Encryption Standard
ASIC	Application Specific Integrated circuit
ATPG	Automatic Test Pattern Generation
BMC	Bounded Model Checking
CMOS	Complementary Metal-Oxide-Semiconductor
CPLD	Complex Programmable Logic Device
DAC	Digital-to-Analog Converter
DFS	Design-For-Security
DFT	Discrete Fourier Transform
DPR	Dynamic Partial Re-programmability
DSDPC	Delay Signatures at Different Process Corners
DSP	Digital Signal Processing
EDA	Electronic Design Automation
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GDS-II	Graphic Database System-II
HD	Hamming Distance
HDL	Hardware Description Language
HF	High Fan-out
HT	Hardware Trojan
IC	Integrated Circuit
ICAP	Internal Configuration Access Port
I/O	Input/ Output
IP	Intellectual Property core
ISCAS	International Symposium on Circuits And Systems
ITD	Inverted Temperature Dependence
JTAG	Joint Test Action Group

LUT	Look-Up Table
OC	Obfuscation Cell
OTP	One-Time-Programmable
PAR	Place And Route
PLD	Programmable Logic Device
PROM	Programmable Read Only Memory
PUF	Physically Unclonable Function
RE	Reverse Engineering
REFSM	Reverse Engineering Finite State Machine
RO	Ring Oscillator
RTL	Register Transfer Level
SAT	SATisfiability
SDPC	Simulation at Different Process Corners
SEM	Scanning Electron Microscopy
SEU	Single Event Upset
SoC	System on Chip
SOM	Scanning Optical Microscopy
SRAM	Static Random Access Memory
STF	State Transition Function
TP	Transition Probability
VHDL	Very High Speed IC Hardware Description Language
VLSI	Very Large Scale Integration

1

Introduction

The focus of this research work is to study feasible hardware security threats in very large scale integration (VLSI) device based safety critical applications to identify key areas of improvement and to suggest solutions for the same. This chapter introduces the terms and definitions in the field of "secure and reliable VLSI designs."

Over the past few decades, programmable logic devices (PLD) such as complex PLDs (CPLD) and field programmable gate arrays (FPGA) are extensively used as the basic building modules in most digital systems due to their robust features such as high density, field re-programmability and faster time-to-market. In addition, usage of PLDs in design reduces discrete integrated circuits (IC) population and associated interconnections on printed circuit board. This, in turn, increases the reliability of PLD-based systems. However, when features such as unit cost, speed, power are considered, application specific integrated circuits (ASIC) are most suitable devices. They also address the problem of fast obsolescence associated with PLDs. Hence, it is clearly evident that electronic systems have proliferated over the past few decades to the point that most aspects of daily life are aided or affected by the automation, control, monitoring, or computational power provided by ICs.

As shown in Figure 1.1, a typical PLD design cycle includes programming using hardware description language (HDL), synthesis (netlist generation), simulation, mapping to technology, place and route (PAR), generation of configuration bitstream and finally

programming the target device. In general, ASICs follow the same design flow as PLDs till synthesis by converting the target design using basic digital components. Further, it has various stages such as layout formation using standard cell library, mask generation, chip fabrication and package with post-silicon testing.

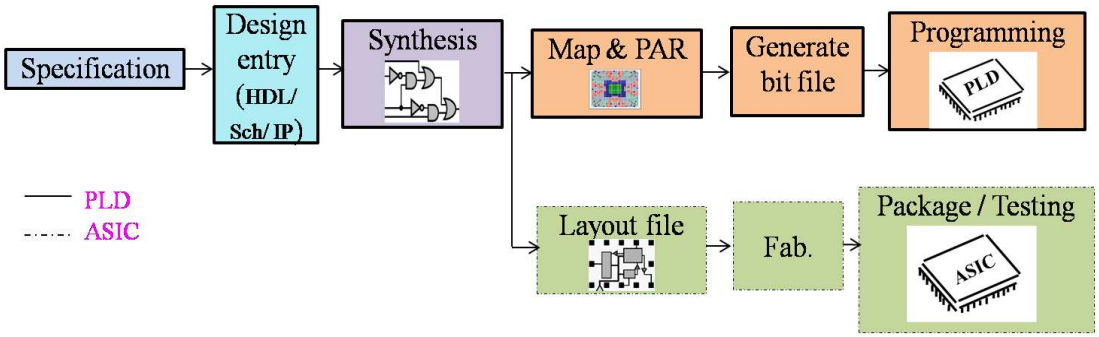


Figure 1.1: PLD and ASIC design cycle

From Figure 1.1, it is very explicit that to develop digital systems with any such devices, the designer relies more on the outsourced intellectual property (IP) cores, commercial electronic design automation (EDA) tools and offshore fabrication services, which introduces more third-party involvement in design. An adversary may use any of these opportunities to introduce various malicious attacks on IPs or ICs. These include:

1. An attacker may steal and claim ownership of the design, resulting in “piracy attack”,
2. An untrusted IP user may perform “duplication or cloning attack” whereas IC foundry may perform “IC overbuilding attack”,
3. An attacker may “reverse engineer (RE)” the IP or IC to understand the functionality and
4. Insertion of malicious circuits, also known as “hardware Trojans (HT)”to change the functionality, reduce reliability, denial of service and destroy the chip.

Therefore, together with featured advantages of PLD and ASIC based digital designs, many security concerns have arisen; especially, the ability to trust these ICs to perform their specified operation (and only their specified operation) has always been a security concern and has recently become a more active topic of research. The increased deployment of such devices in safety critical applications or sensitive areas, such as nuclear power plant, space, military, health care, treasury and border control has also heightened the need to develop the secure and reliable very large scale integration (VLSI) designs that ensures the design and data security. The goal of this thesis is to investigate the potential hardware security threats in VLSI device based safety critical applications, in particular, to identify key areas of improvement in hardware security and to suggest solutions for the same with their associated overhead. This chapter introduces the term, definition and working mechanism of various malicious attacks and their protection methods.

1.1 Hardware Trojans

1.1.1 Hardware Trojan and its Components

HTs are malicious additions or modifications to existing circuit elements and it can be inserted at any stage of ICs life cycle. A typical HT design consists of “HT trigger” and “HT payload” logics [1.1]. While receiving the trigger logic, payload circuitry initiates HT action. Design of HT circuitry involves two basic components:

1. Activation mechanism called “HT trigger” and

2. Introduced effect called “HT payload.”

For a simple design with a moderate gate density, the number of possible HTs with varying activation mechanisms and different effects is very large in number. Usually, an

adversary tries to hide their tampering in such a way that it is extremely difficult to detect with conventional verification or post manufacturing test. Intuitively, it means that HT is manifested or triggered either by signals that have potential rare values (i.e. low testable nodes) or in noncritical or re-convergent paths. Figure 1.2 shows how signal 'S' can be manipulated using a combinational and sequential HT. The dotted region is trigger circuitry and the payload is implemented using XOR gate. On trigger, the original value of 'S' is inverted. Combinational HT triggers on simultaneous occurrence of a set of rare node conditions, while sequential HT depends on the sequence of rare occurrence happening before trigger.

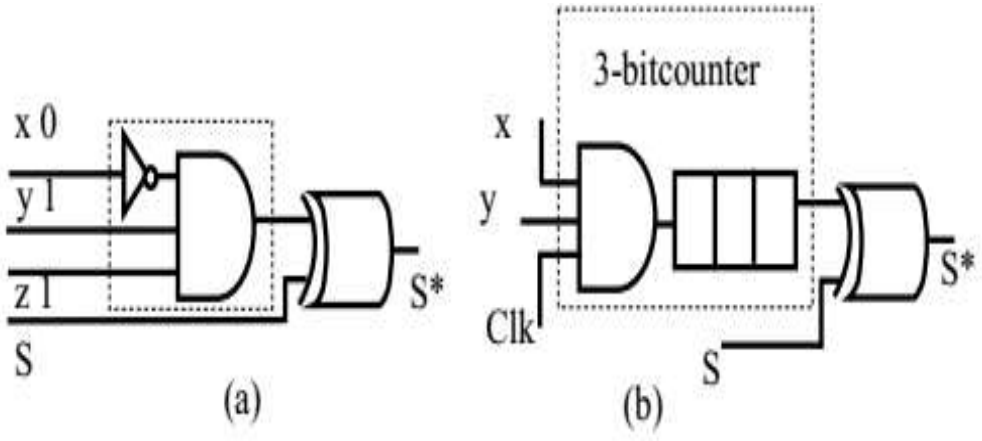


Figure 1.2: Examples of HT design (a) Combinational HT and (b) Sequential HT

In general, HTs are designed either as 'always on' or 'triggered only under certain operating conditions' to disable or destroy the functionality, reduce reliability [1.2], leak valuable information or encrypted key [1.3], or to insert a backdoor within the IC [1.4]. A simple HT design may be as simple as a paragraph change in the specification, an extra line of source code, modification of the silicon die at the fabrication plant, or changes in the CMOS geometries used etc. The effect of HT may be unacceptable and can severely affect the safety of systems.

1.1.2 Taxonomy of Hardware Trojans

Initially, various HT taxonomy models were proposed using different attributes [1.5-1.8]. Afterwards, Rajendran et al. [1.9] exposed a more in-depth discussion of HT taxonomy gathering all probable HT characteristics using the following five attributes: (1) Phase of insertion; (2) Abstraction level; (3) Activation mechanism; (4) Location and (5) Effects as shown in Figure 1.3.

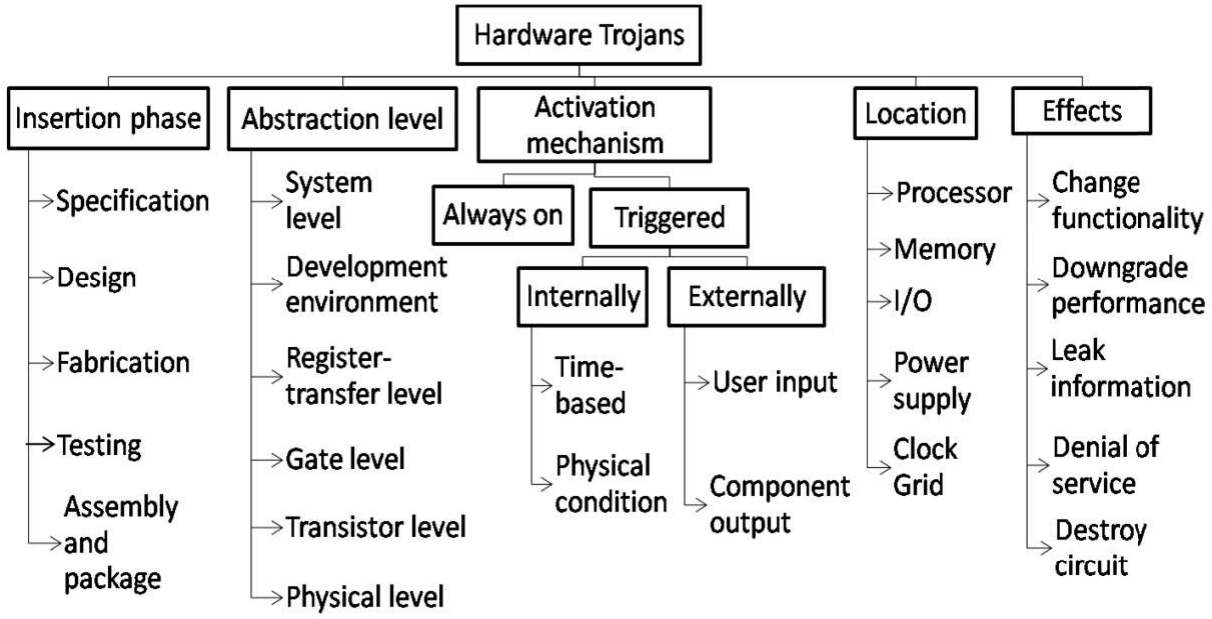


Figure 1.3: Taxonomy of hardware Trojans

Under the phase of insertion, the HTs can be as a hidden specification or, alteration in design characteristics such as digital and timing. Even during the fabrication phase, a mask substitution can also happen like replacing the genuine mask by the fake one. In assembly phase, the tested chip and other HT components are also assembled on a single package. Under abstraction level, the various levels at which HT circuits can be inserted are described. At the system level, the different hardware modules, interconnections and communication protocols

used are covered and HT may be triggered by the modules in the target hardware. The programmable logic devices development environment includes synthesis, simulation, verification and validation tool in which the tools and the associated scripts can be used to insert HT. The register transfer level (RTL) module describes each functional module in terms of registers, signals and Boolean functions; hence HT can be easily designed and inserted at this level.

At the gate level, HT can be inserted to alter the functionality of the design and hence they are also called “functional HT” which can be combinational or sequential module. The transistor level gives the HT designer a control over circuit characteristics like power and timing, because they are the basic components to build logic gates. Since the inserted HT is generally used to alter the functionality of the design, they are also called as “functional HT”. At the layout level, the dimensions and locations of all circuit components are described. HT may be inserted by modifying the size of the wires, distances between circuit elements and reassigning metal layers. HT at this level is described as modifications to the parameters of the IC physical design and hence they are called “parametric HT”. Based on the number of gates inserted, a HT can also be classified as small or big. Also, based on the distribution of the HT in the design, it can be classified into tightly coupled or loosely distributed among modules in design.

In general, HT activation mechanisms are classified as always active or trigger based. The parametric HT in which changes are made at the physical layout is mostly always active HT i.e. it can be activated at any time. The trigger based HT can either be internally or externally triggered. Internal triggering of HT is either time-based or based on physical-condition (such as electromagnetic interference, humidity, altitude and atmospheric pressure.) or reaching specific state of a state machine. External triggering requires external inputs such as user inputs and

outputs of any component that interact with the target device like sensor-triggered, logically-triggered and antenna signal-triggered. The sensor triggered HTs are triggered by physical conditions such as temperature and voltage. Triggering by logic-conditions such as state of the flip-flops, counter, clock signal, data, instruction and interrupts are termed as logic-triggered HTs. Any nearby antenna signal can also trigger the HT circuit. An adversary can design the HT to disable or destroy functionality, reduce reliability, leak valuable information or encrypted key, to deny a service or to insert a backdoor within the IC.

The HT circuitry might be residing on the processing units, memory, I/O, power grid, clock grid or combination of all. In processing unit, they change the execution order of instructions whereas in memory units, they alter the value stored in the memory and they may also block read or write access to certain memory locations. In I/O units, HT may reside on the peripherals of the chip or within the computer. HTs on power supply unit may alter the voltage and current supplied to the chips that cause failure. Using clock grids, an adversary can change the frequency of the clock and insert glitches in the clock supplied to the chip, causing fault attacks. Therefore, it is very clear from the taxonomy of HTs that for a simple digital design with a moderate gate density, the number of potential HT models with varying activation mechanisms and different effects is very large. Eventually, deriving a single detection method for all the probable HTs is practically infeasible.

1.2 Feasible Hardware Security Threats with Potential Attackers

1.2.1 Intellectual Property Cores

Reuse-based IC designs using hardware intellectual property core have become the eminent design practice in semiconductor industries. IP cores are nothing but the reusable unit of

logic, cell, or chip layout that is the intellectual property of one party. It may be licensed to another party or can be owned and used by a single party alone. The hardware IP cores can be in any of following three forms such as:

1. Soft IP, i.e. synthesizable register transfer level descriptions,

2. Firm IP, i.e. gate-level designs directly implementable in hardware and

3. Hard IP, i.e. chip design database as GDS-II.

Based on the designer's requirement, they are purchased as soft or firm or hard IPs and used in the design of complex systems. For example, IPs are widely used as basic building blocks within ASIC or PLD such as FPGA and CPLD designs. Usage of these tested, pre-verified and reusable modules**,** in turn**,** reduces the design**,** verification time and cost dramatically. This is with respect to the commercial aspects of IPs. Apart from this, many critical applications attempt to preserve their indigenous designs as IPs to handle the fast obsolescence of PLDs and to upgrade with technology.

1.2.2 IP & IC Attack Model

It is very clear that to develop digital systems using PLD or ASIC devices, the designer relies more on the outsourced IP cores, commercial EDA tools and offshore fabrication services, which introduces more third-party involvement in the design. Due to these high third-party involvements, the chip design or HDL code or IP core becomes highly vulnerable to various design security threats such as:

1. An attacker may steal and claim ownership of the design, resulting in "piracy attack",

2. An untrusted IP user may perform “duplication or cloning attack” whereas IC foundry may perform “IC overbuilding attack”,

3. An attacker may “reverse engineer” the functionality of an IP or IC and

4. An adversary may insert malicious circuits, also known as “hardware Trojans.”

In most of the scenarios, IP vendors and the chip or system designers are the key victims of these attacks. To avoid such illegal actions, the major victims of IP attack i.e. IP vendors and chip designers need to implement suitable anti-tamper solutions such as watermarking, fingerprinting, hardware metering, encryption and obfuscation. In this section, an attempt is made to analyze the potential IP attackers with their scope on various attacks.

In general, HDL or IP or IC attacks are aimed to claim the ownership (i.e. piracy) or copy and sell others IP cores or perform RE of IPs or ICs either to understand the logic or to insert malicious logics (i.e. HTs). In most of the scenarios, these attacks are attempted by the IP buyers e.g. chip or system designers or by the outsourced foundries. Also, end users who have full control over final products can accomplish such attacks. From the analysis of various IP attacks and attackers, the realistic IP threat model is derived as shown in Figure 1.4. This model highlights the scope of different attackers such as designer, foundry and end user to achieve potential IP attacks. It is clearly evident that, threats such as IP piracy, cloning, RE and insertion of HTs can be successfully achieved by all three attackers as they have full access to IP cores or ICs. As the trend of fabless semiconductor technology emerges, the chip designer exports the GDS-II files to outsourced foundries for fabrication. Using this GDS-II file, the untrusted foundry can fabricate more number of chips than required, which is also called as IC overbuilding. Eventually, these attacks are implemented by fabrication foundries.

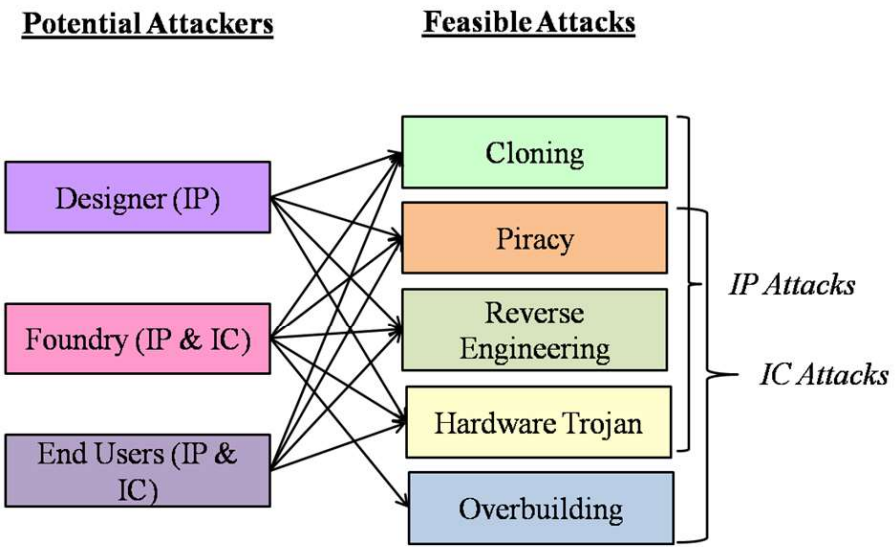


Figure 1.4: Feasible hardware security threats with potential attackers

1.3 Taxonomy of IP & IC Anti-Tamper Solutions

In IP protection taxonomy, the two terms "passive" and "active" are effectively used to denote the attacker's ability i.e. to allow using the illegal copy or not. Basically, passive methods do not prevent the attacker from using the illegal copy, whereas active methods avoid it to an extent. In general, five different hardware IP & IC protection approaches are widely used in IP based digital designs, such as watermarking [1.10], fingerprinting [1.11], metering [1.12], encryption [1.13] and obfuscation [1.14] as shown in Figure 1.5. The watermarking method embeds the digital signatures of IP vendors to ensure their ownership if illegal copy of their IP is found. However, this will not avoid copying and illegal distribution of IPs to further parties. Hence, fingerprinting techniques were proposed which embeds unique digital signatures for each IP buyers in addition to watermarking. By identifying the buyer's signature from IPs, it is possible to detect from which user these IPs are distributed. As watermarking and fingerprinting

techniques can be used only to thwart IP piracy and copying, they are classified under passive techniques.

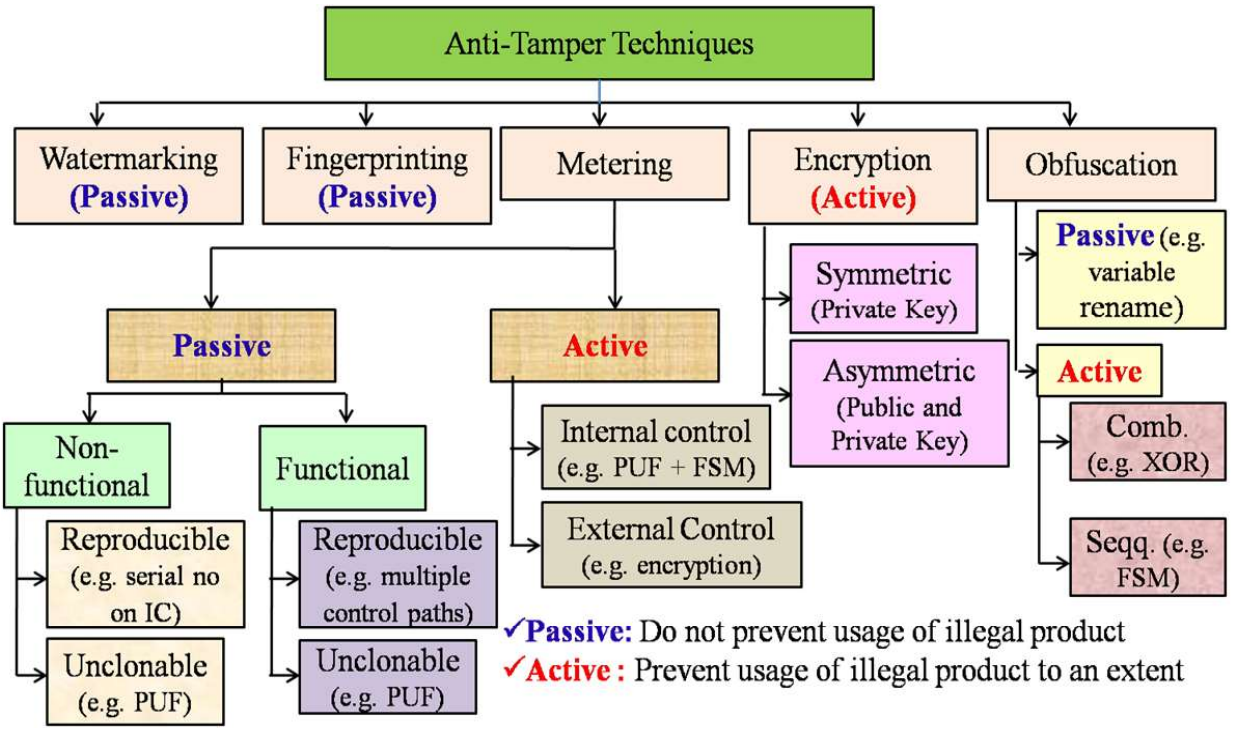


Figure 1.5: Taxonomy of IP & IC Anti-Tamper Solutions

Using the process variations associated with chip fabrications, the physically unclonable function (PUF) modules generate the unique device specific signatures. The chip designer used this concept to tag each ICs fabricated from foundries. The PUF circuits can be used to implement both passive and active methods. Digital signatures stored in external memory or serial number printed on chips could be better non-functional passive metering approach. Unfortunately, they are easily reproducible; hence, delay based PUFs inserted in unused spaces of chips could be used to generate unclonable unique random signatures of each ICs.

The passive functional metering protocol for detection of the unauthorized chips is to monitor and evaluate the chips while they are in use. This is achieved by designing chips with a

single data path but controlled by multiple versions of the same control path specifications. A small part of the chip is retained programmable, so the control path would be programmed into the chip. However, this approach by itself does not strongly deter the attackers, since an adversary with access to one unlocked chip can replicate the programmable memory's content from one chip and then use the information to configure and enable other chips. To make it unclonable, the data for the programmable part should not be replicated on other chips, naturally defending against the overbuilding attacks. The active metering technique covers the obfuscation technique with PUF logics as internal method and data encryption as external approaches, but this is with respect to IC attacks, it doesn't deal with IP protection. In active PUF based solutions, without knowing the appropriate PUFs signatures to unlock the chips, untrusted foundries cannot distribute the ICs to third-parties; hence, hardware metering techniques impede IC overbuilding attacks. Though protection mechanisms of fingerprinting and metering are more or less same, the first one deals with IP copying and the second approach is used to avoid IC overbuilding.

The cryptography theories are well proven models to ensure the design or data security and the same can be used to increase RE complexity of IPs. If single key is used for both encryption and decryption, then it is symmetric key encryption, e.g. advanced encryption standard (AES). In contrast, if two separate keys (say public and private) are used in this process, then it is asymmetric method, e.g. RSA. However, these methods require a separate encryption tools to perform it. In case of logic obfuscation approaches, the original design is modified either by IP vendors or designers in such a way that it is difficult to read and understand the functionality. It is classified as passive and active methods. In case of passive obfuscation, original design is obscured without modifying the functionality.

By inserting additional combinational logics and state elements in finite state machine (FSM), active obfuscation methods hide the original functionality behind secret initialization key. Without the proper key value, circuit will not perform its intended functionality and it is difficult to perform RE. Thereby, it also avoids the insertion of successful HTs. It is explicit that logic obfuscation technique is one of the best solutions among all anti-tamper solutions that completely avoids piracy, cloning, overbuilding and increases the complexity of RE and HT attacks. The hardware obfuscation technique modifies the circuit such that the original functionality is preserved and hidden behind a secret initialization key. Without the key value, it is extremely challenging to understand the functionality.

1.4 Summary

a. In recent years, most of the digital designs are surfaced with hardware security threats like HTs, with effects ranging from a subtle degradation of service to a complete and permanent shut-down of a system.
b. Especially, in applications such as nuclear power plant, space and defense where safety critical systems play an important role, it is mandatory to increase system immunity for the data and design security against HT attacks. Therefore, it is utmost importance to ensure that the chip being in use performs only the intended function.
c. As PLDs and ASICs are two dominant digital devices being used in these applications, this thesis ultimately analyze them for the feasible hardware security threats and tries to propose the suitable anti-tamper mechanisms. Therefore, this work will definitely help the digital system designers or users to understand the severity of various security threats associated with PLD and ASIC life cycles and to incorporate a combination of protection methods at respective stages.

d. In future, there will be an enormous usage of outsourced IP cores as well as much scope to preserve indigenous critical designs as IPs to manage the fast obsolescence of PLDs. Eventually, it is very important to secure the IPs as well as the critical digital designs against malicious attacks. This, in turn, triggers the active research on anti-tamper mechanisms of digital design, which holds a lot of research avenues, is undertaken in academia and industry.

References:

[1.1] Y. Alkabani, and F. Koushanfar, "Extended Abstract: Designer's Hardware Trojan Horse," in Proc. Int. Workshop Hardware Oriented Security Trust, pp. 82-83, 2008.

[1.2] R. S. Chakraborty, I. Saha, A. Palchaudhuri, and G. K. Naik, "Hardware Trojan insertion by direct modification of FPGA configuration bitstream," Design Test Comput., vol. 30, no. 2, pp. 45-54, Apr. 2013.

[1.3] Y. Jin, and Y. Makris, "Hardware Trojans in wireless cryptographic integrated circuits," Design Test Comput., vol. 27, pp. 10-25, 2010.

[1.4] S. Skorobogatov, and C. Woods, "Breakthrough silicon scanning discovers backdoor in military chip," in Proc. 14th Int. Conf. Cryptographic Hardware and Embedded Systems, pp. 23-40, 2012.

[1.5] F. Wolff, C. Papachristou, S. Bhunia, and R. S. Chakraborty, "Towards Trojan free trusted ICs: problem analysis and detection scheme," Proc. Des. Auto. Test Europe, pp. 1362–65, 2008.

[1.6] X. Wang, M. Tehranipoor, and J. Plusquellic, "Detecting malicious inclusions in secure hardware: challenges and solutions," Proc. IEEE Int. Workshop Hardware Oriented Security Trust, pp. 15–9, 2008.

[1.7] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, "Hardware Trojan: threats and emerging solutions," Proc. IEEE Int. High Level Design Validation Test Workshop, pp. 166–71, 2009.

[1.8] M. Tehranipoor, and F. Koushanfar," A survey of hardware Trojan taxonomy and detection," IEEE Trans. Des. Test Comput., Vol. 27, no. 1, pp. 10–25, 2010.

[1.9] J. Rajendran, E. Gavas, J. Jimenez, V. Padman, and R. Karri, “Towards a comprehensive and systematic classification of hardware Trojans,” Proc. IEEE Int. Symp. Circuits and Systems, pp. 1871–74, 2010.

[1.10] D. Kirovski, and M. Potkonjak, “Local watermarks: Methodology and application to behavioral synthesis,” IEEE Trans. CAD of Integrated Circuits and Systems, vol. 22, no. 9, pp. 1277–1284, 2003.

[1.11] D. Holcomb, W. Burleson, and K. Fu. “Power-up SRAM state as an identifying fingerprint and source of true random numbers,” IEEE Trans. Computers, vol. 58, no. 9, pp. 1198–1210, 2009.

[1.12] F. Koushanfar, and G. Qu, “Hardware metering,” in Proc. ACM/IEEE Des. Autom. Conf., pp. 490-493, 2001.

[1.13] J. Guajardo et al., “Physical Unclonable Functions and Public-Key Crypto for FPGA IP Protection,” Field Programmable Logic and Applications, pp.185-195, Aug. 2007.

[1.14] R. S. Chakraborty, and S. Bhunia, “HARPOON: a SoC design methodology for hardware protection through netlist level obfuscation,” in IEEE Trans. on CAD, vol. 28, pp. 1493 - 1502, Oct. 2009.

2

Literature Study and Motivation

The present chapter discusses the detailed and systematic literature study on HT attack with its prevention, detection and diagnosis techniques and hardware obfuscation based anti-tamper solutions. The aim of this research work is to study feasible hardware security threats in VLSI device based safety critical applications to identify key areas of improvement and to suggest solutions for the same. The outline of the thesis is discussed with the scope of the research work.

2.1 Analysis of Potential Hardware Trojan Attacks and Feasible Anti-Trojan Design Guidelines

For a simple digital design with a moderate gate density, the number of potential HT models with varying activation mechanisms and different effects is very large in number. Eventually, deriving a single detection method for all the probable HTs is difficult. Though several literature surveys on HTs have been published, this section aims to review the potential HT threats and defense solutions for PLD and ASIC life cycles i.e. various stages of PLD and ASIC life cycles are analyzed individually for the possible HT attacks. The state of the art HT prevention, detection, and diagnosis techniques are mapped to the valid stages of PLD and ASIC life cycles and a set of anti-Trojan design guidelines is recommended in this chapter. Hence, this work helps the system designers or users to understand the severity of HT attacks associated with PLD and ASIC life cycles and to incorporate a combination of HT preventive, detective and

diagnosis methods at respective stages. That is, adhering to these guidelines improves safety measures and protects safety systems against HT attacks.

2.1.1 Feasible HT Attacks and Defense Solutions for PLD Life Cycle

2.1.1.1 Feasible HTs in PLD Life Cycle

With the increasing globalization trend, most of PLD vendors become fabless suppliers and outsource the design for manufacture. From chip vendors, the system designers procure the devices. Later, following a set of design procedures with the aid of IP cores and EDA tools, designers program the device. Finally, the system with programmed devices is deployed into the environment. Eventually, the life cycle of any PLD has a constant deal with multiple third-party accesses such as vendors during chip design and trade, foundry during manufacture, customers during configuration and field exposure during operation.

To categorize potential HTs allied with various stages of PLDs, the life cycle of PLDs is classified as: (1) Pre-customization phase; (2) Customization phase and (3) Post-customization phase as shown in Figure 2.1. The pre-customization phase deals with the blank i.e., un-programmed devices, which includes design (i.e. vendor) and manufacturing (i.e. foundry) stages. Next phase is the customization phase, where the PLD is integrated into the final system and is programmed to implement its intended functionality. To explain in detail about customization phase, it starts from coding the design using HDL, synthesis (netlist generation), simulation, mapping to technology-specific components, place and route, generation of the configuration bitstream and finally programming the target device. Based on the application, the programmed device is assembled with few other functional blocks such as digital signal processing (DSP), analogue-to-digital converter (ADC), digital-to-analogue converter (DAC)

and memory, and deployed in the field. This field operating condition and associated access come under post-customization phase.

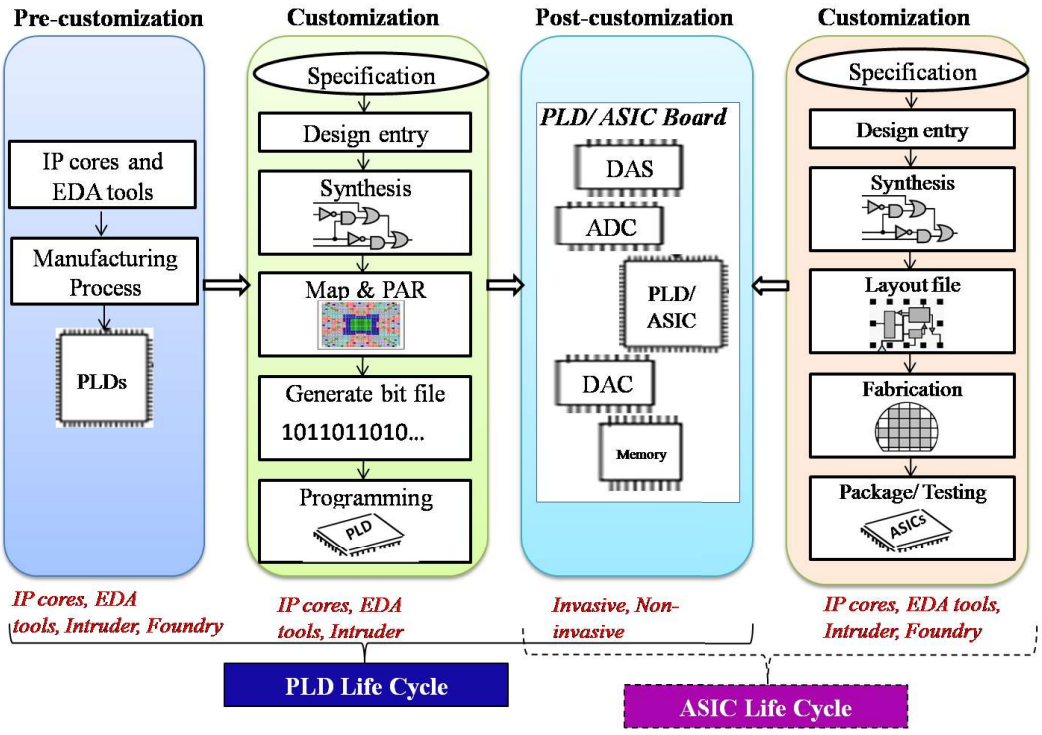


Figure 2.1: Feasible HT threats in PLD and ASIC life cycles

An adversary may introduce HT attacks at any of these phases either to malign the image of a competitive company or to cause a malfunction in electronics used by critical systems or to leak secret information from the device. In most of the scenarios, IP cores, EDA tools, an intruder in chip design team and untrusted offshore foundry are main causes for HT action in the pre-customization phase. Unless the end users have a facility to investigate the chip architecture, the wide range of users trust chip vendors and directly integrates the device with systems without testing. Therefore, an intruder in a PLD design team can exploit this strategy to embed HTs into the device to bypass the software security facilities or keep backdoors for internal access of devices used in mission-critical applications [2.1].

Due to the tremendous advancement in technology, ICs have become more complex in nature. The chip designers are not always able to afford the manufacturing equipment for small feature sizes; As a result, semiconductor companies highly depend on outsourced fabrication foundries around the world. It is widely believed that as the functionality of PLDs is unknown during fabrication, it is very difficult to insert potential HTs in it. However, advanced PLDs or programmable system on chips have inbuilt hardcore macros (like DSP, memory blocks) in addition to programmable fabrics. Therefore, modern PLDs are highly vulnerable to various foundry-based HT attacks. For example, Trimberger [2.2] discussed the risks of untrusted foundries fabricating FPGAs. During fabrication, untrusted insiders or outsiders who can access the manufacturing process or third-party tools used by foundries are the major threatening sources of HT attacks. They can perform intentional modification of process parameter such as nitrate concentration or dopant concentration in the channel or channel length [2.3] or mask layout [2.4] or wire bound area during package. These HTs will be intelligently inserted in such a way that they left undetected by post-manufacturing tests.

In customization stage, outsourced IPs or commercial EDA tools and the associated scripts or untrusted intruder in the design team are leading causes of HT threats. When third-party IP cores are used in the design, it is vital to ensure that the acquired IPs do not hide malicious functions. Because it gives a lot of scope to IP vendors for either inserting malicious codes in HDL or modifying primitives during synthesis or altering PAR. There are more such realistic HT benchmarks available in trust-hub [2.5] and literature [2.6, 2.7] and most of them are implemented at RTL. For most of the PLD-based digital designs, vendor specific or universal EDA tools are highly used at various stages of customization phase. Hence, malicious codes in EDA tools [2.8, 2.9] may add unintended functionality or collect valuable data about FPGA chip

or the system to be built on the FPGA chip. Marchand et al. [2.10] demonstrated effective HT insertion in unused resources of FPGA by executing infection scripts. In addition, an untrusted insider in a design team can accomplish any HT attacks right from specification to device programming stage. However, detection of specification manipulation is extremely challenging due to lack of golden or reference specification.

The vulnerability of PLDs in the post-customization phase is heavily dependent on the application. It involves the physical security of the device (i.e. accessibility to outsiders) as well as the functionality of the device (e.g. military devices are highly vulnerable to HT attack than consumer electronics). Hence, a PLD-based system, developed after a lot of research and development activities should be made secure in all respects. Before inserting a successful malicious logic, adversary performs physical attacks to glean the functionality of chips. Kastner et al. [2.11] classified these attacks as invasive, semi-invasive or non-invasive.

Invasive attacks comprise probing stations, chemical solvents, lasers, ion beams or microscopes to tap the internals of device and gain access to its functionality. It also entails de-packaging to enable probing or imaging of device. Side-channel attacks are semi-invasive, that exploit the leakage of physical information when an application is being executed on a system. A wide range of physical phenomena such as power, timing, electromagnetic radiation, optical, thermal profile and acoustic characteristics of the system are used in this attack. For example, side-channel attacks such as simple or dynamic power analysis can extract device configuration data and encryption keys. Moradi et al. [2.12] demonstrated a successful side-channel attack against the bitstream encryption engines of various FPGA vendors. After a successful attempt of bitstream extraction, an adversary further performs bitstream reverse engineering. It is novel practice to discover the properties of design, e.g. by decoding the bitstream into higher level

abstracts such as netlist or HDL [2.13, 2.14]. Note et al. [2.13] detailed a technique to extract netlist from the bitstream for a particular family of devices. Though bitstream reverse engineering is considered to be practically challenging process, it is possible with additional time, money and sophisticated lab facilities.

In most scenarios, side-channel attacks are a serious threat to cryptographic implementations. Such attacks allow leaking a secret value, e.g., a key, processed inside a cryptographic device from observing physical properties like the timing differences or the power consumption or the electromagnetic emanation. It is practically infeasible to leak IP core or HDL code by side-channel attacks as those are physical or logical representations; they are not data present in the circuit. Imaging or delayering techniques are more adequate for IP theft and there is no successful side-channel analysis in literature that reveals the entire algorithm or IP core or HDL code.

In passive or non-invasive attacks, physical probing of target devices is performed to obtain information about its data or internal working. Perhaps, probing the wires on the board or the pins into the chip is the easiest attack. For example, the bitstream in a volatile FPGA-based system is stored in external non-volatile memory. In such applications, an attacker can either directly read the memory or snoop bus transactions when the FPGA is powered on and the bitstream is loaded into the device.

On the other hand, most of the modern PLDs are provided with additional design features such as read-back, dynamic partial re-programmability (DPR) using joint test action group (JTAG) or internal configuration access port (ICAP) or other programming ports and built-in testability circuits. By extensively using these features, it is possible to extract configuration data

or system information stored or to re-program the device with a malicious program. For example, many FPGAs allow read-back feature that can directly read out the configuration of the device either through JTAG, ICAP or another similar bitstream programming interface. Hence, a successful read-back attack directly acquires the bitstream from the functioning device. DPR is highly encouraged in net-worked FPGA-based systems as it gives functional flexibility by performing the remote update of configuration. Using this feature, an adversary may establish communication with target system and update the device with malicious logics.

2.1.1.2 Potential HT Defense Solutions for PLD Life Cycle

The probable HT attacks on PLD life cycle is discussed so far. It is well understood that deriving single HT defense solution for all kinds of HTs is practically difficult. However, various HT preventive, detective and diagnosis techniques have been proposed over a decade for the different sets of HT threats. To build a highly secure digital system in this insecure world, it is required to understand the state of the art HT threats and incorporate suitable countermeasure techniques in respective stages of chip life cycle. To detect HTs inserted in PLDs during the pre-customization phase, both non-destructive and destructive testing can be done on fabricated devices as listed in Figure 2.2.

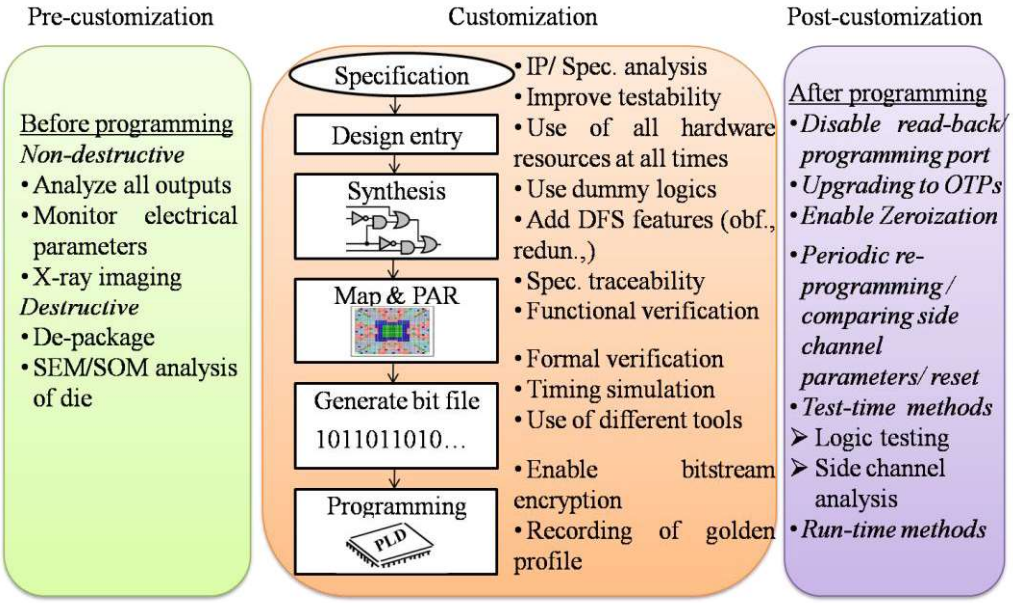


Figure 2.2: Potential HT defense solutions for PLD life cycle

As a non-destructive testing, it is recommended to analyze the un-programmed blank devices for their default output values and also verifying their electrical characteristics as per the datasheet. To detect HTs inserted at die level and wire bound area, X-ray imaging [2.15] is highly useful. Unfortunately, the radiation can still cause damage. In destructive approach, chip de-cap or de-package is carried out first. Furthermore, visual inspection or imaging techniques such as scanning electron microscopy (SEM) [2.16] and scanning optical microscopy (SOM) [2.17] shall be used to identify HTs inserted inside the die. But these techniques require golden ICs to compare as well as they are time-consuming and become more complex with the increase in transistor density. An adversary aims to insert HT such that it will not be detected by post-manufacturing testing. Hence, these methods can also be used by chip vendors or system developers or end users as the chip screening process against HT attacks that might be inserted either by IP cores or EDA tools or intruder or foundry in pre-customization phase.

In customization phase, design tampering at specification level is extremely difficult to detect in later stages; hence, it is a good design practice to perform the functional verification of specifications by trusted outsiders. As mentioned in Section 2.1.1.1, IP cores are primary sources of black boxes in the design. Thus, it is crucial to develop IP trust verification methods to perform complete verification and validation of outsourced IPs before integrating them in design. So far, a lot of research on this topic was successfully carried out [2.18–2.27] either by code or structural analysis or by formal verification techniques.

In general, IP cores are delivered as RTL or VHDL or Verilog codes; hence, code coverage analysis is performed on RTL codes to identify suspicious signals that may be a part of an HT [2.18, 2.19]. Also, IP cores can be automatically analyzed to spot suspicious signals either using controllability or reachability values of signals [2.20] or by identifying gates with low activation probability [2.21]. Later, Zhang et al. [2.22] explained a technique called VeriTrust which marks gates that are not driven by functional inputs as suspicious. That is, the spotted gates are driven by HTs, as they do not perform any computation on functional inputs. Finally, the small numbers of suspicious gates are manually analyzed to determine if they are part of an HT. However, code or structural analysis techniques have the following limitations such as:

1. No guarantee on HT detection;

2. Burden of manual analysis and

3. Analyze only the combinational parts of the design.

Formal verification is an algorithmic-based approach for logic verification that exhaustively proves functional properties of the design [2.19]. Traditionally, it can be performed by equivalence checking or property checking or model checking. The equivalence checking is a

very promising method for comparing functions at different abstraction levels (e.g. specification vs. netlist, netlist vs. design, etc.). Here, the elements such as logical blocks and state transition functions will be translated into Boolean equations which are then converted into reduced ordered binary decision diagram model by co-factoring. Negative equivalence checking results indicate errors in changing the abstraction level or may be evidence for inserted HTs during translation processes [2.23]. However, this method requires the golden reference.

In property checking, the designer can develop a set of functional or security properties that the design should satisfy. Love et al. [2.24] explained the proof-carrying technique where IP vendors construct a valid formal proof to show that the pre-defined properties hold during system operation. The consumer can check the proof that is received with the IP package by running it through the proof assistant interpreter, e.g. Coq language. If the proof is valid, then the consumer can accept the design, knowing that its operation stays within the functional boundaries set by the security properties. However, proof-carrying technique fails to identify whether the design has additional vulnerabilities while satisfying these properties. Subsequently, Rajendran et al. [2.25, 2.26] proposed a bounded model checking (BMC) based formal verification technique to detect HTs in third-party IPs that corrupt registers [2.25] and leak critical information [2.26]. Here, the model checker takes the Verilog code along with the property written as a Verilog assertion, e.g. “no-data-corruption” or “information leakage” and derives a Boolean satisfiability (SAT) formulation for validating or invalidating the property.

Especially, in BMC a property is determined to hold for at least a finite sequence of state transitions. This SAT formulation is fed to a SAT engine and if a satisfying assignment is observed within specific clock cycles, that assignment is a witness against the target property. One can develop a similar set of properties to detect different kinds of HTs. In addition, the

number of clock cycles for which the design is checked can be increased by using an automatic test pattern generator, instead of a BMC, to check for the property. This is because an automatic test pattern generator (ATPG) consumes less memory than a BMC. Unlike proof-carrying method [2.24], the techniques in [2.25, 2.26] detect HTs that do not violate the pre-defined properties and is automated. Reece et al. [2.27] proposed a method to compare two similar but untrusted designs for all possible input combinations to identify the functional differences between them.

To build the trust against HT attacks by EDA tools, the widespread design verification schemes such as specification traceability, functional simulation, timing simulation and formal verification shall be used to verify the results after synthesis and PAR. In addition to the vendor specific EDA tools, it is put forward to use the universal EDA tools by targeting the specific vendor devices. Finally, the results from the vendor specific tools can be verified with the universal tools to detect if any malicious logics are introduced by the EDA tools. On top of these techniques, to avoid HTs inserted at unused resources of PLDs, Potkonjak [2.28] proposed a solution by using all hardware resources at all times, i.e. on all clock cycles. Furthermore, Khaleghi et al. [2.29] used dummy logics to fill the remaining unused resources. This will ultimately allow no room within the hardware for additional HTs. However, concerning the reliability aspect of safety-critical systems, it is not recommended to use all hardware resources at all times and the device level de-rating is followed. Hence, it is suggested to utilize the unused resources of PLDs to design the test logic and it should be in idle state during normal operation. Eventually, this method thwarts insertion of HTs at unused resources and also it retains the device de-rating. Though HT insertions at the unused resources are much concerned by designers, it is also possible to build HT attacks using the logic that already exists in design. To

evade such attacks, Chakraborty et al. [2.30] proposed a logic-obfuscation-based design-for-security (DFS) solution, which is detailed in depth in Section 2.1.2.2.

Since rareness of the internal circuit nets of obfuscated circuit is modified by inserting additional combinational logics and redundant states in FSM, this method increases the difficulty in inserting successful and hard-to-detect HTs. In addition, reverse engineering of EDA tools and development of EDA tools trust evaluation framework [2.31] kind of verification schemes can be adopted in the design flow. Enabling the cryptographic algorithms to generate encrypted bitstreams (mostly provided by all FPGA vendors) improves FPGA design security against snooping of configuration data. Finally, the golden profile of side-channel parameters (explained in Section 2.1.2.2) can be recorded in the customization phase. Periodic monitoring of these values in post-customization phase will be useful to detect whether any design alterations happened in the field environment or not [2.32].

As per the definition of HTs, physical attacks do not come under HT attacks. However, to carry out a successful HT attack, it is required to perform physical attacks on programmed ICs to acquire a complete understanding of the logic. Eventually, techniques that avoid physical attacks will increase the difficulty of a successful HT action. First, defense solutions for semi-invasive and non-invasive attacks are discussed. Later, techniques to avert invasive attacks will be covered. Yu et al. [2.33] provided a solution to the differential-power-based side-channel attacks. Using logic duplication and symmetric routing techniques, the authors ensured that the output power remains the same irrespective of input values. Later, Bogdanov et al. [2.34] investigated the possibility of side-channel resistant implementations of AES-based FPGA encryption engine using state of the art threshold masking techniques.

To avoid non-invasive attacks, in addition to various design security schemes, it is also required to ensure the system security i.e., to deny the access to outsiders. Most of the FPGA vendors support the bitstream read-back facility to verify the programming. However, this feature can also be used to extract the programmed configuration. To avoid this attack, recent FPGA vendors let the designers to disable the read-back option at the final design stage. Another better way to protect the bitstream is to program the volatile FPGA at a secure facility and consistently keep the device powered during field deployment. These drawbacks can also be avoided by upgrading the systems with anti-fuse (one time programmable) and other non-volatile FPGAs. To secure the encryption algorithms by avoiding permanent key storage and management, PUF-based key generation schemes are introduced. The working principle of PUF is explained in Section 2.1.2.2. These PUF modules are also used to protect configuration data stored in low end FPGA families which do not support bitstream encryption [2.35]. With trusted programming environment, the attacker is left with attempting physical attacks on the device to try and determine the contents of the configuration bitstream. More often, invasive attacks, i.e. reverse engineering of FPGAs are performed mostly on antifuse and non-volatile devices. However, determining the state of a single switch is the practically challenging task since the physical change created by programming the fuse is tough to detect. Therefore, probing the state of millions of switches is so prohibitive as to be considered almost impossible. On top of this, most of FPGA vendors pro-vide a technique called zeroization [2.36], which is the practice of erasing sensitive parameters to prevent their disclosure if the system is attacked, or is at an increased risk of unauthorized access.

Though several design security schemes such as encryption and integrity checking of the bitstream are implemented in the design, they do not detect remote re-programming of the

devices. To avoid this scenario, modern FPGAs have a facility to ensure the trustability of remote updates using passwords. For example, flash lock feature in Microsemi FPGAs [2.36] allows locking the device with a particular passcode, which allows the device to be unlocked and reprogrammed by providing the same passcode. Also, a permanent lock of all programming access ports such as JTAG, ICAP can be implemented which turns the device to one-time programmable device and thus very secure from attempts to subvert the function of the target design. It is well understood that even with the above mentioned various FPGA device and design security features, it is not possible to assure that a design is 100% protected against HT attacks.

MalSarkar et al. [2.37] proposed a redundancy-based HT tolerant method which bypasses HT effects if any exist in the design. As a final measure, it is recommended developing the PLD-based critical systems using on-board redundancy, i.e. using PLDs from different technologies and manufacturers. The outputs from both devices can be driven by a simple ASIC-based voting logic that shall be completely verifiable. As the probability of similar and simultaneous HT attacks in redundant devices is very less, this technique bypasses the effect of HTs and also ensures trustable system operation. Besides, periodic reprogramming, measurement of side-channel parameter and its comparison and reset [2.38] (to avoid activation of sequential HTs) of the device ensure that chip performs only intended functionality and nullifies the HT inserted in field respectively. In addition, most of DFS solutions and non-destructive HT detective approaches such as test time and run-time methods explained in Section 2.1.2.2 can also be applied to PLD-based designs.

2.1.2 Feasible HT Attacks and Defense Solutions for ASIC Life Cycle

2.1.2.1 Feasible HTs in ASIC Life Cycle

ASICs are customized chips which are designed and manufactured for a specific functionality. In contrast to PLDs, the chip designer and the customer is one and the same in most of the ASIC-based applications. As a result, the entire life cycle of ASICs can be classified as (1) Customization phase and (2) Post-customization phase as shown in Figure 2.1. Here, it is evident that the customization phase of ASIC is nothing but the combination of pre-customization and customization phases of PLDs. The simplified ASIC design flow starts from HDL coding, synthesis, simulation, layout generation, chip fabrication and finally post-silicon testing. Here, synthesis does netlist generation by compiling and mapping into standard macro cells, whereas layout stage creates GDS-II design file. After-wards, GDS-II files are outsourced to foundries for mask preparation and die fabrication as explained in Figure 2.1.

Furthermore, packaging and post-manufacturing testing are carried out before supplying the devices to chip designers. Later on, ICs are integrated with systems and implemented in the field and this stage is categorized under post-customization phase. The feasible HT attacks in the customization phase may be due to IP cores or EDA tools or outsourced foundries or untrusted intruder in the design team. Most of the literature of HT attacks explained in pre-customization and customization phases of PLDs (refer Section 2.1.1.1) applies to ASICs too. Especially, HT attacks by IP codes, EDA tools and intruder are common. In case of ASICs, as the chip functionality is known during fabrication, they are highly vulnerable than PLDs for insertion of malicious components, keeping backdoor, etc. For example, Muehlberghuber et al. [2.39]

demonstrated the addition of small and effective HT at the mask-layout level with very less or no knowledge about the inner functionality of the victim circuit.

During system integration in post-customization stage, HT attacks may be initiated by interconnected components such as processing units, memory, I/O, power grid and clock grid. It may cause various malicious effects such as change the execution order of instructions, alter the value stored in the memory, block read or write access to certain memory locations, alter the voltage and current supplied to the chip, change the frequency of the clock and insert glitches in the clock supplied to the chip. Various kinds of physical attacks explained in Section 2.1.1.1 such as invasive, semi-invasive and non-invasive attacks can also be performed on these custom tailored ICs. Lin et al. [2.40] explored an HT design with fewer than 50 gates that were embedded into an AES cryptographic circuit to perform the side-channel attack. As a result, this HT was capable of leaking multi-bit information below the noise power level of the host device to avoid its detection.

2.1.2.2 Potential HT Defense Solutions for ASIC Life Cycle

The set of defense solutions explained in Section 2.1.1.2 for HT attacks by IP cores, EDA tools and the untrusted intruder is also valid to ASIC designs. Subramanyan et al. [2.41] proposed a structural and functional analyses based algorithmic reverse engineering technique to analyze the unstructured netlist to detect any malicious logic. This method also verifies the integrity of ICs and identifies IP violations. Later, Amin et al. [2.42] developed a methodology to detect or prevent HT in third-party IP cores by comparing the outputs of different untrusted implementations of the same IP core. Specification traceability using post synthesis netlist will ensure the design integrity. Furthermore, tests such as functional and timing simulations, formal

verification and external hardware tester module [2.43] can identify any design alteration. With respect to potential HT insertions at the foundry, ASICs are highly vulnerable than PLDs as chip functionality is well defined during fabrication. Hence, trust must be considered as an important design criterion in the design flow of modern ICs instead of being an afterthought. Indeed, ASICs shall be designed with possible DFS strategies in mind to facilitate HT countermeasures. As part of DFS features, which facilitates HT prevention, detection and diagnosis, various techniques are proposed as follows: circuit obfuscation, layout filling or modification, ring oscillator (RO) networks, PUF blocks, redundancy techniques, duality and gate-keeper logic.

Hardware obfuscation techniques obscure the functional and structural properties of the design, so that it is very difficult to read and understand during reverse engineering. They are in general realized by adding extra combinational gates such as XOR/ XNOR [2.44, 2.45], redundant states in FSM [2.46], or by inserting memory elements [2.47]. Most of them are key-based techniques, which hides the correct functionality behind a sequence of input vectors or secret keys. Without these key values, it is tough to read and understand during reverse engineering. Hence, it is extremely challenging for an attacker to perform successful insertion of potential HTs. During layout generation, the physical verification process shall be performed to confirm that the desired modifications have been made and no malicious modifications (i.e. HTs) have been introduced either by the untrusted intruder or EDA tools. For example, XOR check based physical verification [2.48] involves comparing the original and modified layout databases or GDS-II by XOR operation of the layout geometries. This check results a database which has all the mismatching geometries in both the layouts.

Also, it is important to develop a systematic practice to analyze the susceptibility of the circuit layout to HT attacks by a foundry. Salmani et al. [2.49] derived a successful vulnerability

analysis approach of the circuit layout to HT insertion. Here, to minimize the impact of HTs on chip layout, a few layout parameters such as availability of white space, unused routing channels for HT placement were studied and the existence of nets with low transition probability on non-critical paths were investigated. Xiao et al. [2.50] proposed to keep the functional filler in unused spaces of layout and they are connected to form a functional circuit for the post-manufacturing test. As a result, this approach makes it challenging for an adversary to find any real estate on the device to insert HT. By comparing digital signatures generated during self-testing, a designer would know whether a chip has been tampered for HT attacks or not. In addition, the signature of layout designer can be embedded in the form of watermark [2.51]. During verification, if signatures of ICs are closely matched with the desired one with very few bits flipped, then it strongly indicates the inclusion of HTs in the foundry. Similarly, by measuring the frequency of RO network [2.52–2.54] which is already inserted in circuit layout and verification of its values in post-silicon stage aids to detect if the circuit is modified during fabrication. The physical characteristics of PUFs are unique; hence, any deviation in this value due to malicious modifications will not perform the intended functionality [2.55]. HTs are nothing but faults that are intentionally inserted in the circuit to cause malfunction.

In this scenario, redundancy-based fault mitigation techniques such as triple modular redundancy and duplication with the comparison shall be applied either at design or component level to bypass the HT effect. The authors in [2.56, 2.57] discussed few methods to detect HTs at rare trigger nodes using circuit duality concept and prevent data leakage with gate-keeper logic. Since low controllable and low observable nodes are suitable HT trigger and payload candidates, improving design testability either by increasing the transition probability of low controllable nodes [2.58, 2.59] or by tapping low observable nodes to the output ports can be a part of design

enhancement steps. To evade the control of chip fabrication over single untrusted foundry, split-manufacturing technique is proposed [2.60, 2.61] that keeps interlock between designer and various fabrication services and increases device integrity.

In post-customization phase (or post-silicon), HT detection tests are widely performed using destructive and non-destructive methods as shown in Figure 2.3.

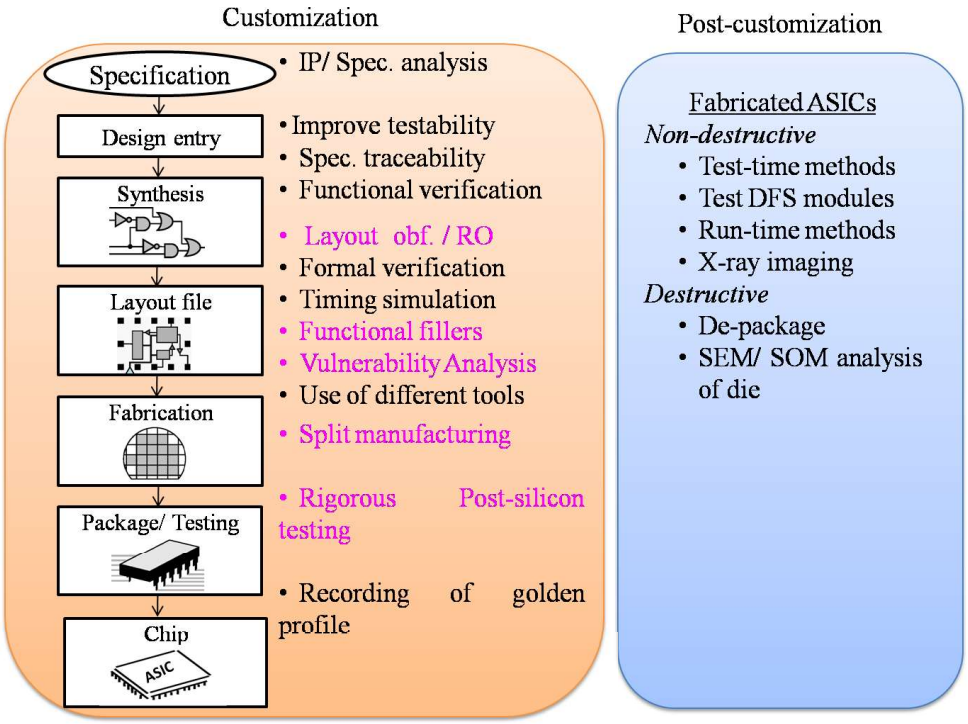


Figure 2.3: Potential HT defense solutions for ASIC life cycle

Destructive methods are already covered in Section 2.1.1.2. Though HT modifications might only be placed in distinct IC instances, destructive reverse engineering using chip de-cap and SEM or SOM analysis, as used by Agrawal et al. [2.62], can be used to find golden or reference ICs. Furthermore, these golden ICs can be utilized to determine tampered IC. Bhasin et al. [2.63] proposed a method to detect if HTs are inserted at foundry stage by comparing the optical microscopic pictures of the silicon product and the original view from a GDS-II layout

database. With respect to non-destructive HT detection approaches, logic testing and side-channel analysis are extensively used. Logic testing is basically a functional test. To perform effective logic testing to detect HTs, complete set of effective test vectors are required to activate hidden HTs and to propagate their effects to output ports.

An adversary can insert a number of different HT instances with multiple activation mechanisms. As a result, generation of deterministic test patterns to activate all of them is impractical. Chakraborty et al. [2.64] proposed a statistical approach for test vector generation by exciting rare logic conditions at internal nodes for multiple times. This, in turn, maximizes HT activation probability and detection by logic testing. In another logic testing method [2.65], additional key inputs are used to trigger rare occurring events and corresponding signatures (i.e. outputs) of the circuit are generated in a special mode of operation called as transparent mode. However, this technique works well for small circuits, as generating multiple signatures for a complex design would be difficult. Waksman et al. [2.21] have used the stealthiness property of HT affected nets to isolate them from a given design. They also developed a tool, called FANCI, which uses a scalable, Boolean functional analysis to detect these infected nets.

Side-channel analysis has been widely applied to HT detection due to the fact that HT attack by malicious insertion might reflect its presence in any of the following side-channel parameters such as leakage or transient current [2.66–2.69], signal delay [2.70–2.72] and electromagnetic radiation due to switching activity [2.73]. Instead of individual measurement of side-channel parameters, various techniques to perform multi-side-channel analysis [2.74–2.76] are proposed to enhance the sensitivity of these methods. However, in modern nano-meter technologies, HT detection efficiency of side-channel analysis is limited by large intrinsic device parameter variations also called as process variations. Hence, various characterization or

calibration mechanisms [2.77–2.81] are integrated with side-channel methods to increase detection efficiency. Also, these HT detection approaches typically require golden ICs to compare the measured values to identify HT infected ICs. The general methods to obtain golden chips are as follows:

1. Destructive reverse engineering to ensure that they are trusted [2.82] and

2. By exhaustive testing to verify their trustworthiness.

However, both approaches are highly expensive and time-consuming. As a result, few techniques to perform side-channel analysis without golden ICs [2.83, 2.84] are proposed. Overall, to achieve effective test time HT detection approaches, it is suggested to combine side-channel analysis with logic test approaches that focus on generating appropriate test patterns to activate HTs [2.85, 2.86].

While HT detection at pre and post silicon level is desirable, the existing HT protection techniques cannot guarantee 100% coverage of all types and sizes of HTs. Hence, continuous runtime monitoring of system functionality or parameters [2.87–2.91] can significantly reduce the potential effect of HT attacks. With some performance over-head, these runtime measurements can be used either to disable the chip upon HT detection or to bypass it in the presence of unreliable components. It is also possible that HT could be inserted in such run-time monitors [2.92] and cause malicious actions. As a result, it is at the most necessary to ensure the trustworthiness of DFS monitors. Therefore, a combination of post-silicon HT defense solutions can be used to find the HTs inserted at the design stage and the manufacturing stage. However, these techniques can also be used to combat the HTs introduced in other stages. Though the comparison of various HT prevention and detection techniques has already been published, the

HT defensive complexities at different stages of PLD and ASIC development cycle have not been compared yet. Table 2.1 provides the quantitative comparison of the HT defensive complexities based on the following parameters such as the cost, time, dependency on golden sample and test vector generation, original design change and HT defensive guarantee.

Table 2.1: Comparison of HT defensive complexities at different stages of PLD and ASIC development

Development stages	**Cost**	**Time**	**GS & TVG**	**ODC**	**HDG**
Pre-Customization (PLD)					
BD (Non-Des)	medium	medium	high	NA	medium
BD (Des)	high	high	high	NA	high
Customization (PLD)					
Specification	low	low	ND	NC	low
Design entry	low	low	high	high	medium
Syn. & Map & PAR	high	high	medium	high	high
Programming	low	low	ND	NC	low
Post-Customization (PLD)					
PD (Test-time)	medium	low	high	NC	low
PD (Run-time)	medium	low	high	NC	medium
Customization (ASIC)					
Specification	low	low	ND	NC	low
Design entry	low	low	high	high	medium
Syn. & layout	high	high	medium	high	high
Pac.& testing	medium	medium	high	NC	medium
Post-Customization (ASIC)					
IC (Non-Des)	medium	medium	high	NC	medium
IC (Des)	high	high	high	NC	high

GS – Golden Sample; TVG– Test Vector Generation; ODC – Original Design Change; HDG – HT Defensive Guarantee; BD – Blank Device; Des –Destructive; NA – Not Applicable; NC – No Change; ND – Not Dependent; Syn – Synthesis; PD – Programmed Device, Pac – Package.

The total budget and manpower allotted to accomplish the successful HT defensive action using sophisticated tools and software is quantified by the parameter "cost." The overall period required to perform HT diagnosis or detection is termed as "time." The "golden sample and test vector generation" highlights the design stages which rely highly on the golden IP & IC samples and test vector generation in HT detection. In few HT defensive techniques, it is required to introduce design modification without affecting the original functionality and that is obtained by "original design change" and finally, "HT defensive guarantee" defines the reliability of the HT diagnosis or detection techniques.

Though there are several HT defensive solutions, the technique with the highest accuracy is more recommended for evaluation. Through this comparison, the practicability of HT diagnosis or detection at various stages of PLD and ASIC development has to be determined. With respect to the pre-customization phase of PLDs, the un-programmed blank devices are subjected to either non-destructive or destructive testing. In general, destructive testing such as SEM and SOM analyses [2.16, 2.17] gives better results than non-destructive testing [2.15]. However, it requires high cost and time and also relies on the golden chip for HT detection.

In customization phase, HT defense at the specification and design entry (includes IP core) stages can be carried out at a relatively minimal cost and time with low to medium accuracy. For example, a thorough review of the specification and architecture provides a minimum accuracy for HT detection whereas simulation of the HDL has medium accuracy. During the physical design processes such as synthesis, mapping and PAR, an evaluator would require higher resources and more time to perform HT diagnosis or detection [2.18–2.35, 2.37]. Here, the functional and structural verification [2.18–2.27] of netlists and technology mapped design requires golden sample and the inclusion of DFS features [2.33–2.35] might change the

original design without affecting the functionality. Using the design security features [2.36] such as encryption, read-back and flash lock during bitstream generation or programming stages, it is possible to achieve a minimum level of HT defensive with low cost and time.

In post-customization phase, logic testing [2.64, 2.65], side-channel analysis [2.38, 2.66–2.76] and run-time monitoring schemes [2.87–2.91] are predominantly used to test the programmed devices whose efficiency relies highly on the golden IC and test vector generation. The side-channel measurements are significantly affected by process variations; hence, test-time methods achieve minimum accuracy. As run-time monitoring can also detect sequential HTs that triggers during operation, they guarantee a medium HT detection with medium cost. HT defensive complexities involved in most of the customization and post-customization stages of ASICs are comparable to PLDs. During the physical design process, techniques like formal analysis and reverse engineering of layout [2.41–2.43] provide a high accuracy, but they are expensive and time-consuming. Besides, DFS schemes such as layout obfuscation [2.44–2.47], layout filling [2.50], RO [2.52–2.54] and PUF [2.55] require additional hardware resources. HT defensive at the package and testing stage has a medium cost and time because of the complexity of the design. Also, it requires the generation of a complete set of test vectors, but it cannot detect HTs that triggers during operation. Therefore, medium accuracy level shall be possible at this stage. The complexity of non-destructive methods such as X-ray imaging, logic testing and side-channel analysis is already discussed. While the destructive or reverse engineer technique is the most accurate HT detection method in post-customization phase, but it does not guarantee the integrity of all the other fabricated ICs.

Ultimately, the security aspect of PLD and ASIC designs in pre-customization, customization and post-customization phases is analyzed for potential HT attacks with possible

HT defense solutions. It is also clearly evident that there are a few pitfalls associated with some of HT defensive approaches such as the requirement of golden ICs, generation of a complete set of test vectors to trigger HT circuitry, design over-head due to extra circuitry in invasive techniques and process variation effects in side-channel measurements. However, implementation of possible HT defense solutions at different stages of IC life cycle increases the design immunity against HT attacks and ensures the security of systems.

2.2 Hardware Obfuscation based Anti-Tamper Solutions

In general, the design of any digital systems includes outsourced IP cores, commercial EDA tools and offshore fabrication services. IP cores are the reusable unit of logic, cell, or chip layout that can be in any of following three forms such as soft IP, firm IP and hard IP. Due to this high chance of third-party involvement, the IP cores or HDL codes or chip designs in digital circuits are highly vulnerable to various design security threats such as IP piracy, duplication or cloning, IC overbuilding, reverse engineering and HTs. For example, RE of any safety critical designs may cause leakage of critical parameters or encrypted keys and insertion of HTs. As a result, it may deny or destroy the system during critical operations and cause serious consequences. As per [2.93, 2.94], the semiconductor industry loses $4 billion annually due to such attacks. Also, it was estimated that the cost of counterfeiting and piracy for G20 nations was U.S. $450–650 billion in 2008 and U.S. $1.2–1.7 trillion in 2015 [2.95]. As these attacks will not only have a negative impact on brand reputation and research and development efforts but also might have the serious impact on systems and operations. As a result, it is at the most required to incorporate the necessary defensive solutions against such IP or IC attacks [2.96]. Especially, it is very important to ensure the design and data security of safety critical systems such as space,

defense and nuclear as the compromise of which will lead to the disastrous event [2.97]. Logic obfuscation approach is one such defensive solution that is used to increase the RE complexity.

2.2.1 Logic Obfuscation and Classification

In general, obfuscation is a technique that transforms an application or a design into one that is functionally equivalent to the original but is significantly more difficult to reverse engineer [2.98]. Here, logic obfuscation technique modifies either HDL code or structure (netlist) [2.99] such that it is very difficult to read and understand during RE and hence increase the cost and complexity of RE attacks. This, in turn, ensures a certain level of design security by preventing against stealing of original design by analyzing and rebuilding during RE. Also, it avoids the insertion of successful and hard-to-detect HTs. To achieve better RE complexity, Desai [2.100] discussed the requirements to perform effective hardware obfuscation as follows:

1. It shall be hard to differentiate the obfuscated hardware from the core logic,

2. It shall change the behavior of the circuit dynamically in the obfuscated mode,

3. Original specifications of the chip should not be modified and

4. It shall preserve the same number of inputs and outputs as the original design with minimum design overhead.

Besides, as timing is one of the critical design parameters in safety critical applications, obfuscated design should accomplish the system timing requirement. With respect to the changes in original functionality, hardware obfuscation techniques are widely classified as passive and active obfuscation as shown in Figure 2.4.

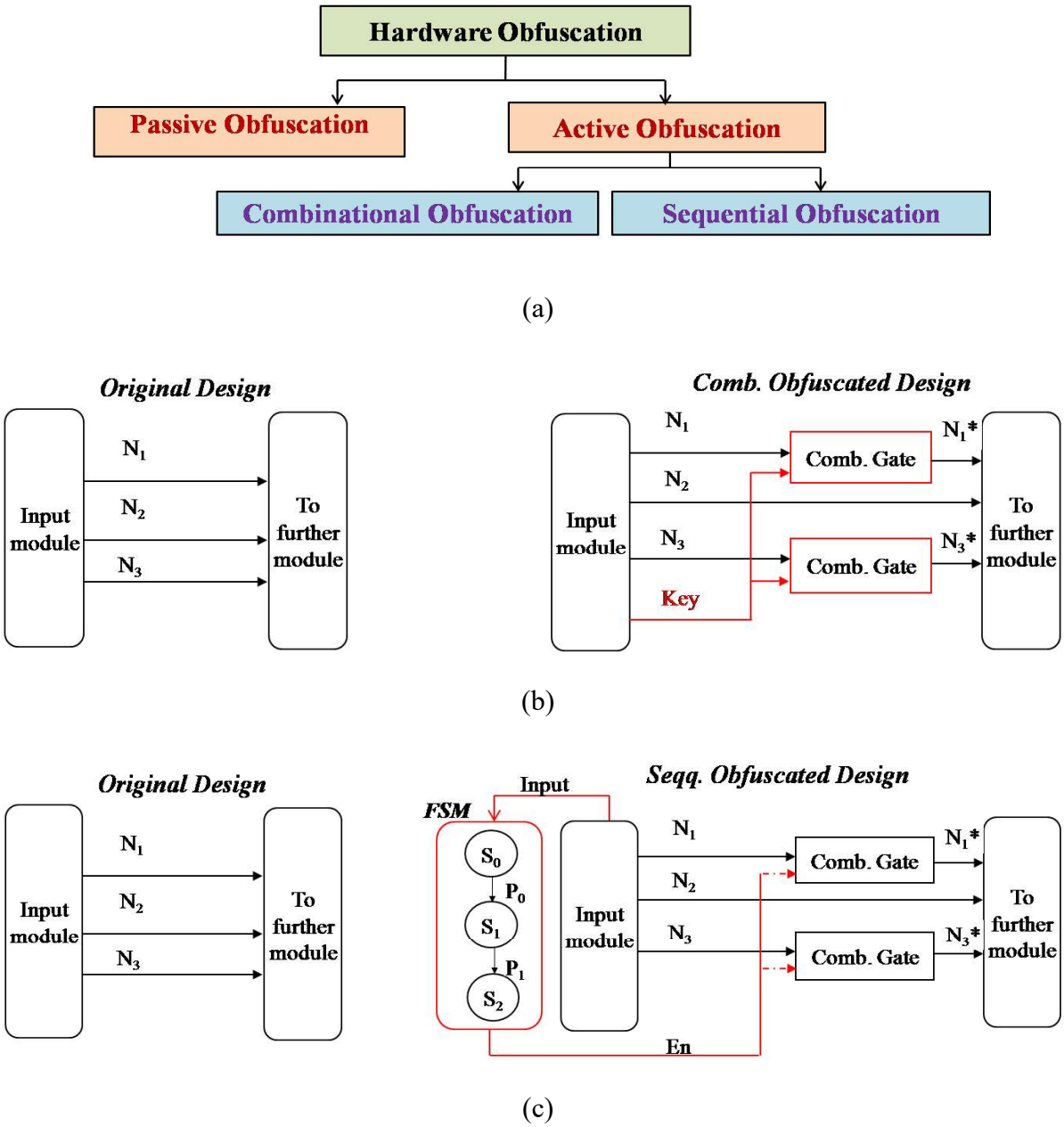


Figure 2.4: (a) Taxonomy of hardware obfuscation techniques (b) Combinational hardware obfuscation and (c) Sequential hardware obfuscation

By definition, passive obfuscation techniques modify the circuit description, but it does not affect the functionality. For example, it employs either string substitution by variable renaming or comment removal or structural change by loop unrolling or register renaming of HDL codes [2.101-2.103] or obscuring branch functions (e.g. for, while) [2.104]. In contrast,

active obfuscation schemes directly alter the circuit functionality without perturbing the original specifications by inserting additional logics in it. The active obfuscation techniques can, in turn, be broadly categorized as combinational and sequential obfuscations [2.105]. In combinational logic obfuscation, the selected internal wires are modified using different obfuscation cell structures such as XOR, XNOR gates and MUX as shown in Figure 2.4a.

The general criterion to select internal wires is that it shall be the non-critical path and one of the inputs is connected to a 1-bit key input. Upon applying the valid key, the obfuscated design will exhibit the correct functionality. However, to build a secure key storage is highly challenging since attackers may control them in a hostile environment and thus, carry out physical attacks [2.106]. To solve this issue, sequential obfuscation methods are developed. In sequential logic obfuscation, additional obfuscated states are introduced in addition with the obfuscation cells. More often, it is implemented as key-based techniques and the obfuscation cell are driven by FSM outputs, rather than externally stored key values. It enables circuit operation in two distinct modes such as obfuscated mode (S_0, S_1 in Figure 2.4c) and functional mode (S_2 in Figure 2.4c). Normal functionality is enabled by successful application of the secret key and the mode switching or state transition function (STF) is defined by Eqn. 2.1.

$$\text{STF} = \text{combination (present state, present inputs)} \tag{2.1}$$

Hence, a particular sequence of input vectors (P_0, P_1 in Figure 2.4c) on initialization is required to enter into functional mode. Otherwise, the circuit remains in obfuscated mode and generates incorrect control value to the obfuscation cells; As a result, the circuit does not perform the actual operation. Since active obfuscation modifies the functionality of the design, it keeps the

obfuscated code or circuit as a black block in a design that is not possible by passive methods. This thesis concentrates mainly on active obfuscation schemes.

2.2.2 Need of Obfuscation Techniques for PLDs

In general, logic obfuscation techniques are extensively used to achieve the IP cores or IC protection against piracy, cloning, RE, chip over-building, etc. However, these techniques can also be applied to PLD-based digital designs for most of simulation or structural RE based security threats. Most of the PLDs design starts with HDL coding as the frontend process. Following by a few back-end processes such as synthesis, mapping, fitting or PAR and bitstream generation are performed using the vendor specific EDA tools. To ensure design correctness, the HDL code undergoes simulation at pre-synthesis, post synthesis and post fitting or PAR. The netlist is generated in the synthesis stage whereas bitstream is generated after fitting or PAR. Finally, the bitstream is used to program the target device and deployed in the field.

With the assumption that front-end processes are completely trusted, major PLD threats are induced by third-party EDA tools [2.107, 2.108], manipulations at the field [2.109] and by natural or human-made phenomena such as radiation effects [2.110]. Most of the static random access memory (SRAM)-based FPGAs are volatile. Hence, bitstreams are stored in a separate programmable read only memory chip connected to the FPGA. During every power up, the bitstream configures the FPGA. An adversary may utilize this opportunity to execute the side-channel attacks [2.111], further extract the bitstream to perform cloning or RE or tampering [2.112, 2.113]. In the case of CPLDs or non-volatile memory based FPGAs, theft of programmed boards can be performed to RE it [2.114]. The basic aim of such attacks can vary from analyzing and rebuilding of competitor‘s technology to destroying of critical systems by inserting HTs

[2.115, 2.116]. To protect the IP cores and to prevent fraud, e.g., by cloning an FPGA or manipulating its content, modern FPGAs offer a bitstream encryption feature. However, a successful attack against the bitstream encryption engine was demonstrated in [2.111]. To successfully carry out such attacks, an adversary first performs RE of the complete design. Hence, it is necessary to embed a technique that either thwarts RE or increases RE complexity. Obfuscation is one such solution that increases RE complexity and acts as anti-tamper or anti-Trojan techniques to improve hardware security.

2.2.3 Existing Work on Active Obfuscation Methods

The logic obfuscation technique is one of the most popular IP or IC protection techniques. In 2008, Roy et al. [2.117, 2.118] explained the classical combinational logic obfuscation method that conceals the IC designs by inserting the XOR/ XNOR gates on selected non-critical wires. One of the inputs to the key gates is the functional input and the other control input is connected to the common key register. Upon applying the correct key, the obfuscated design will exhibit the intended functionality. In order to avoid the key extraction by image processing-based RE [2.119], the authors proposed to replace the XOR gate with the XNOR gate and the inverter and, similarly, replace XNOR gates with XOR gates and inverters. However, this approach incurs high area and power overheads due to the logic redesign. To increase RE complexity of obfuscated gates, Rajendran et al. [2.120] developed an algorithm to insert XOR, XNOR gates at non-resolvable and corruptible gates for a stronger obfuscation; so that, the encrypted circuit is not vulnerable to the fault-analysis attack. Later, Colombier et al. [2.121] presented the IP or IC protection mechanisms such as logic encryption, logic obfuscation, logic masking and logic locking using few combinational circuits. Besides, graph analysis-based novel

technique was proposed to select the optimal nodes to be modified to achieve effective logic locking of the combinational netlist.

In addition to the usage of XOR, XNOR gates as the combinational key gates, MUX are extensively used. For example, Rajendran et al. [2.122] proposed two algorithms that insert XOR, XNOR and MUX gates at locations which maximize the hamming distance between correct and incorrect outputs. In 2015, Zhang [2.123] explained the circuit obfuscation technique using two inputs MUX and PUF as the lock and key mechanisms. Here, the obfuscated net and its complement are connected to MUX inputs and PUF key is given to the selection line of MUX. The functionality of obfuscated cells cannot be identified unless correct obfuscation keys are given. The chips that are authorized by the designer can only guarantee the correct functionalities. Hence, this obfuscation framework can prevent IC from RE, piracy and overbuilding. Afterwards, Wang et al. [2.124] recommended a scheme to replace the selected logic gates with specially designed MUXs. The authors are not the first to obfuscate circuits with MUXs, but they are the first to use programmable camouflage connectors to configure MUXs.

In 2009, Chakraborty et al. [2.125] explained the structural modification based obfuscation technique and it is the first work to put forth the active sequential obfuscation scheme by means of hardware protection and authentication. The obfuscated design has two modes of operation such as obfuscated and normal mode. The mode switching is performed using FSM. The control outputs from FSM need to be “zero” for the proper functioning of the circuit. They are stitched with high fan-out (HF) internal nets using appropriate obfuscation structures e.g. XOR gates or combination logics. During power up, a sequence of input vectors as authentication sequence must be applied to drive the circuit to functional mode. Otherwise, the circuit stays at obfuscated mode and does not perform actual operations. The authors extended

the work to realize obfuscation at register transfer level [2.126] by changing the control or data flow of original circuit. Finally, the design procedure goes through synthesis and optimization and hence protecting hardware blends well into the rest of the logic.

In most of the sequential obfuscation techniques, FSM enters into functional mode only when a correct initialization sequence is applied. In 2013, Desai [2.100] proposed an obfuscation technique where the functional mode is always entered. However, the critical operations (or states) are derived using a variable called "code word" i.e. the code word is integrated into the STF of FSM. Therefore, interlocked control word generation during state transition ensures the correct functionality of the circuit. Except the FSM-based unlocking schemes, also termed as the sequential obfuscation, random number or signature generator circuits such as PUF can be used as the control circuitry [2.127, 2.128]. The physical characteristics of PUFs are unique. Therefore, it generates a device-specific unique challenge-response pairs that are physically unclonable. Wendt et al. [2.127] proposed PUF and PLD based obfuscation method where PUF and FPGA modules replace the critical portion of original logic. The PUF module implements the original functionality of the replaced circuit and FPGA device generates the corresponding challenges to implement its original functionality. Therefore, the designer holds the control on obfuscated PUF and FPGA logics and hides its functionality.

Apart from the application of key-based obfuscation techniques in IP protection, they can also be employed to achieve security against HTs [2.129-2.131]. As the circuit modification introduced by obfuscation hides the rareness of the internal circuit nodes, it is very challenging for an adversary to insert hard-to-detect HTs. Chakraborty et al. [2.129] exploited this strategy to evade the HT action on the original functionality by making them activate only in the obfuscated mode. In [2.130], the authors employed the state obfuscation as a HT countermeasure as it tightly

couples the obfuscation states with true states, providing more paths from the functional states to the obfuscation states. That is, without the correct key, the adversary cannot successfully tamper the critical control unit without being detected. Finally, the illegal states (i.e. obfuscation states) and illegal state transitions induced by a wrong key are examined to detect the occurrence of HTs.

2.3 Objective of the Thesis

Based on the inputs gained from the literature study on “hardware security threats and defense solutions in VLSI designs” and the motivation generated, studies are planned to meet the following objectives:

1. An investigation into the protection mechanisms for hardware Trojans and
2. Hardware obfuscation based design security solution for PLDs

For clear understanding of the scope of the research work, the PLD design flow highlighting the potential attacks associated at different stages with the proposed hardware security defense solutions is depicted in Figure 2.5.

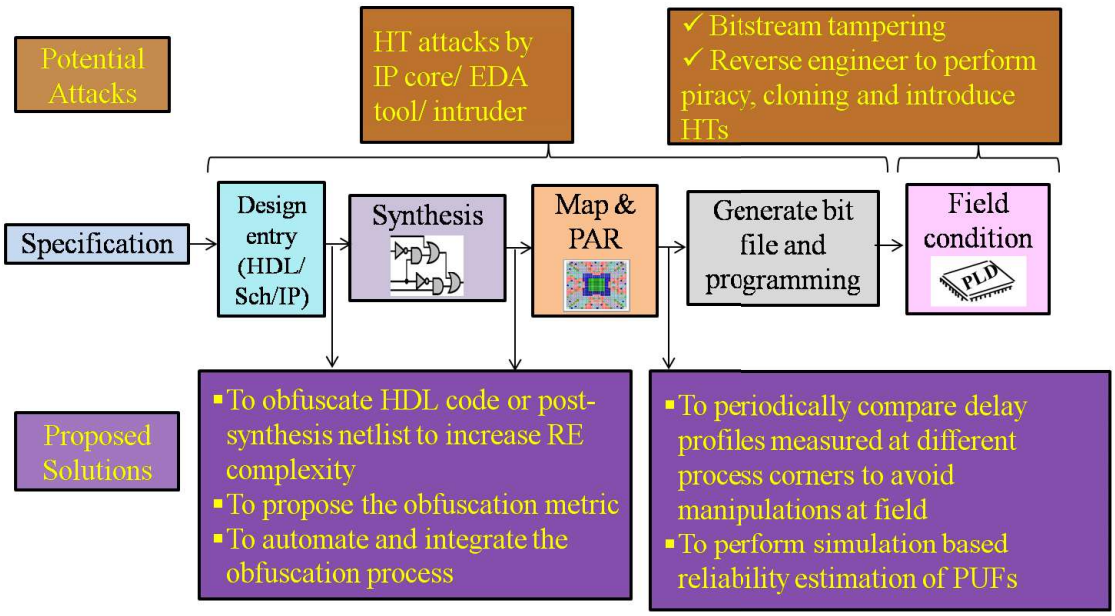


Figure 2.5: PLD design flow highlighting potential attacks with proposed defense solutions

The thesis titled "Secure and Reliable VLSI Designs" is structured into six chapters with three major working chapters addressing the above objectives as given below:

Chapter-1 Introduction

Chapter-2 Literature study and motivation

Chapter-3 Detection mechanism for HTs using delay signatures

Chapter-4 Hardware obfuscation based design security solution for PLDs

Chapter-5 Multi-corner timing analysis based reliability calculation of PUFs

Chapter-6 Summary and scope for future work of the thesis

The first working chapter 3 discusses the development of delay profile based HT detection technique with improved HT detection efficiency. Chapter 4 discusses the experiment

carried on hardware obfuscation based design security solution for PLDs, obfuscation metric calculations and the automation and integration of hardware obfuscation technique using scripting languages. The final working chapter 5 proposes the multi-corner timing analysis based reliability calculation of PUFs.

2.4 Summary

a. In recent years, most of the digital designs are surfaced with hardware security threats like HTs, with effects ranging from a subtle degradation of service to a complete and permanent shut-down of a system.
b. Especially, in applications such as nuclear power plant, space and defense where safety critical systems play an important role, it is mandatory to increase system immunity for the data and design security against HT attacks. Therefore, it is utmost importance to ensure that the chip being in use performs only the intended function.
c. As PLDs and ASICs are two dominant digital devices being used in these applications, they are analyzed for the feasible HT attacks in pre-customization, customization and post-customization stages. Also, a set of possible HT defense solutions applicable to each phase is summarized in this chapter. Therefore, this work will definitely help the digital system designers or users to understand the severity of HTs associated with PLD and ASIC life cycles and to incorporate a combination of HT preventive, detective and diagnosis methods at respective stages.
d. In the future, there will be an enormous usage of outsourced IP cores as well as much scope to preserve indigenous critical designs as IPs to manage the fast obsolescence of PLDs. Eventually, it is very important to secure the IPs as well as the critical digital designs against HT attacks. This, in turn, triggers the active research on anti-tamper

mechanisms of digital design, which holds a lot of research avenues, is undertaken in academia and industry.

e. The detailed study on the definition and classification of logic obfuscation techniques with the need of hardware obfuscation technique to PLDs is discussed in this chapter. Also, the complete literature on the active logic obfuscation technique is explored.

f. Based on the inputs gained from the literature study on "hardware security threats and defense solutions in VLSI designs", the objective of the thesis is derived. Finally, the overall structure of the thesis which has six chapters with three major working chapters addressing the above objectives is clearly framed out.

References:

[2.1] S. Skorobogatov, and C. Woods, "Breakthrough silicon scanning discovers backdoor in military chip," Proc. 14th Int. Conf. Cryptographic Hardware and Embedded Systems, pp. 23–40, 2012.

[2.2] S. Trimberger, "Trusted design in FPGAs," Proc. ACM/ IEEE 44th Des. Auto. Conf., pp. 5–8, 2007.

[2.3] Y. Shiyanovskii, F. Wolff, A. Rajendran, C. Papachristou, D. Weyer, and W. Clay, "Process reliability based Trojans through NBTI and HCI effects," Proc. Conf. Adaptive Hardware & Syst., pp. 215–22, 2010.

[2.4] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson, "Stealthy dopant level hardware Trojans," Lect. Notes Comput. Sci., Vol. 4, no. 1, pp. 197–214, 2013.

[2.5] M. Tehranipoor, R. Karri, F. Koushanfar, and M. Potkonjak, "TrustHub," [Online]. Available: https://www. trust-hub.org/, accessed Sep. 23, 2016.

[2.6] Z. Chen et al. "Hardware Trojan designs on basys fpga board," CSAW Embedded System Challenge, 2008.

[2.7] X. Wang, S. Narasimhan, A. Krishna, T. Mal-Sarkar, and S. Bhunia, "Sequential hardware Trojan: side-channel aware design and placement," Proc. IEEE Int. Conf. Comput. Design, pp. 297–300, 2011.

[2.8] A. Roy, F. Koushanfar, and I. L. Markov, “Extended abstract: circuit CAD tools as a security threat,” Proc. IEEE Workshop Hardware Oriented Security Trust, pp. 65–6, 2008.

[2.9] G. Qu, and Y. Yuan, “Design things for the internet of things—an EDA perspective,” Proc. IEEE/ACM Int. Conf. Comput. Aided Des., pp. 411–16, 2014.

[2.10] C. Marchand and J. Francq, “Low level implementation and side-channel detection of stealthy hardware Trojans on field programmable gate arrays,” IET Comput. Digit. Tech., Vol. 8, no. 6, pp. 246–55, Nov. 2014.

[2.11] R. Kastner and T. Huffmire, “Threats and challenges in reconfigurable hardware security,” Proc. Int. Conf. Engineering of Reconfigurable Systems & Algorithms, 2008.

[2.12] A. Moradi, D. Oswald, C. Paar, and P. Swierczynski, “Side-channel attacks on the bitstream encryption mechanism of Altera Stratix II,” Proc. ACM/SIGDA Int. Symp. FPGAs, pp. 91–100, 2013.

[2.13] J. B. Note, and E. Rannaud, “From the bitstream to the netlist,” Proc. ACM/SIGDA Int. Symp. FPGAs, pp. 264– 64, 2008.

[2.14] F. Benz, A. Seffrin, and S. A. Huss, “Bil: a tool chain for bitstream reverse engineering,” Proc. 22nd Int. Conf. Field Program. Logic Appl., pp. 735–38, 2012.

[2.15] “X-ray Nanotomography Imaging for Circuit Integrity,” [Online]. Available: http://www-ssrl.slac.stanford.edu/content/science/highlight/2011-09-26/x-ray-nanotomography-imaging-circuit-integrity, accessed Sep. 23, 2016.

[2.16] F. Courbon, P. Loubet Moundi, J. J. A. Fournier, and A. Tria, “A high efficiency hardware Trojan detection technique based on fast SEM imaging,” Proc. Design Auto. Test Europe, pp. 788–93, 2015.

[2.17] B. Zhou et al. “Detecting hardware Trojans using back side optical imaging of embedded watermarks,” Proc. 52nd ACM/EDAC/IEEE Design Automation Conf., pp. 8– 12, 2015.

[2.18] M. Banga and M. Hsiao, “Trusted RTL: Trojan detection methodology in pre silicon designs,” Proc. IEEE Int. Symp. Hardware Oriented Security and Trust, pp. 56–9, 2010.

[2.19] X. Zhang and M. Tehranipoor, “Case study: detecting hardware Trojans in third-party digital IP cores,” Proc. IEEE Int. Symp. Hardware Oriented Security and Trust, pp. 67–70, 2011.

[2.20] H. Salmani and M. Tehranipoor, “Analyzing circuit vulnerability to hardware Trojan insertion at the behavioural level,” IEEE Int. Symp. Defect and Fault Tolerance in VLSI and Nanotechnology Systems, pp. 190–95, 2013.

[2.21] A. Waksman, M. Suozzo, and S. Sethumadhavan, “FANCI: Identification of stealthy malicious logic using Boolean functional analysis,” Proc. ACM/SIGSAC Conf. Comput. Comm. Secur., pp. 697–08, 2013.

[2.22] J. Zhang, F. Yuan, L. Wei, Y. Liu, and Q. Xu, “VeriTrust: verification for hardware trust,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., Vol. 34, no. 7, pp. 1148–61, 2015.

[2.23] M. Rathmair, F. Schupfer, and C. Krieg, “Applied formal methods for hardware Trojan detection,” Proc. IEEE Int. Symp. Circuits and Systems, pp. 169–72, 2014.

[2.24] E. Love, Y. Jin, and Y. Makris, “Proof-carrying hardware intellectual property: a pathway to trusted module acquisition,” IEEE Trans. Inf. Forensics Security, Vol. 7, no. 1, pp. 25–40, Feb. 2012.

[2.25] J. Rajendran, V. Vedula, and R. Karri, “Detecting malicious modifications of data in third-party intellectual property cores,” Proc. 52nd ACM/EDAC/IEEE Des. Auto. Conf., pp. 8–12, 2015.

[2.26] J. Rajendran, A. Dhandayuthapany, V. Vedula, and R. Karri, “Formal security verification of third-party intellectual property cores for information leakage,” Proc. 29 th Int. Conf. VLSI design, pp. 547–52, 2016.

[2.27] T. Reece and W. H. Robinson, “Detection of hardware Trojans in third-party intellectual property using untrusted modules,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., Vol. 35, no. 3, pp. 357–66, 2015.

[2.28] M. Potkonjak, “Synthesis of trustable ICs using untrusted CAD tools,” Proc. Design Auto. Conf., pp. 633–34, 2010.

[2.29] B. Khaleghi, A. Ahari, H. Asadi, and S. Bayat Sarmadi, “FPGA based protection scheme against hardware Trojan Horse insertion using dummy logic,” IEEE Embedded Syst. Letters, Vol. 7, no. 2, pp. 46–50, Feb. 2015.

[2.30] R. S. Chakraborty and S. Bhunia, “Security against hardware trojan through a novel application of design obfuscation,” Proc. Int. Conf. Comput. Aided Des., pp. 113–16, 2009.

[2.31] Y. Jin, “EDA tools trust evaluation through security property proofs,” Proc. Des. Auto. Test Europe, pp. 24–8, 2014.

[2.32] G. Sumathi, L. Srivani, D. Thirugnana Murthy, N. Murali, S. A. V. Satya Murty, and T. Jayakumar, “DSDPC: delay signatures at different process corners based hardware Trojan detection technique for FPGAs,” Proc. IEEE Int. Conf. Robotics, Automation, Control Embedded Systems, pp. 1–7, 2015.

[2.33] P. Yu and P. Schaumont, “Secure fpga circuits using controlled placement and routing,” Proc. Int. Conf. Hardware/Software Co-design Syst. Synthesis, pp. 45–50, 2007.

[2.34] A. Bogdanov, A. Moradi, and T. Yalcin, “Efficient and side-channel resistant authenticated encryption of FPGA bitstreams,” Proc. Int. Conf. Reconfigurable Computing and FPGAs, pp. 1–6, 2012.

[2.35] J. Zhang, Y. Lin, Y. Lyu, and G. Qu, “A PUF FSM binding scheme for FPGA IP protection and pay-per-device licensing,” IEEE Trans. Information Forensics and Security, Vol. 10, no. 6, pp. 1137–50, 2015.

[2.36] “Overview of Design Security Using Microsemi FPGAs and SoC FPGAs,” [Online]. Available:http://www.microsemi.com/document-portal/doc_view/132862-overview-of-design-security-using-microsemi-fpgas-and-soc-fpgas, accessed Sep. 23, 2016.

[2.37] S. MalSarkar, A. Krishna, A. Ghosh, and S. Bhunia, “Hardware Trojan attacks in FPGA devices: threat analysis and effective countermeasures,” Proc. 24th Great lakes Symp. VLSI, pp. 287–92, 2014.

[2.38] A. Waksman, and S. Sethumadhavan, “Silencing hardware backdoors,” Proc. 32nd IEEE Symp. Security Privacy, pp. 49–63, 2011.

[2.39] M. Muehlberghuber, F. K. Gurkaynak, T. Korak, P. Dunst, and M. Hutter, “Red team vs. blue team hardware Trojan analysis: detection of a hardware Trojan on an actual ASIC,” Proc. Workshop Hardware and Architectural Support for Security and Privacy, pp. 1–8, 2013.

[2.40] L. Lin, W. Burleson, and C. Paar, “MOLES: malicious off-chip leakage enabled by side-channels,” Proc. Int. Conf. Comput. Aided Des., pp. 117–22, 2009.

[2.41] P. Subramanyan et al. “Reverse engineering digital circuits using structural and functional analyses,” IEEE Trans. Emerg. Topics Comput., Vol. 2, no. 1, pp. 63–80, Dec. 2013.

[2.42] H. A. M. Amin, Y. Alkabani, G. M. I. Selim, “System level protection and hardware Trojan detection using weighted voting,” J. Adv. Res., Vol. 5, no. 4, pp. 499–505, Jul. 2014.

[2.43] N. Kishore, and S. K. Ghoshal, “Design and implementation of a PC based FPGA tester,” IETE Tech. Rev., Vol. 12, no. 2, pp. 119–32,Mar. 1995.

[2.44] J. A. Roy, F. Koushanfar, and I. L. Markov, “EPIC: ending piracy of integrated circuits,” Proc. Des. Auto. Test Europe, pp. 1069–74, 2008.

[2.45] J. Rajendran et al., “Securing processors against insider attacks: a circuit micro-architecture co-design approach,” IEEE Trans. Des. Test, Vol. 30, no. 2, pp. 35–44, Apr. 2013.

[2.46] R. S. Chakraborty, and S. Bhunia, “HARPOON: an obfuscation based SoC design methodology for hardware protection,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., Vol.28, no.10, pp.1493–1502, Oct. 2009.

[2.47] A. Baumgarten, A. Tyagi, and J. Zambreno, “Preventing ic piracy using reconfigurable logic barriers,” IEEE Trans. Des. Test Comput., Vol. 27, no. 1, pp. 66–75, Feb. 2010.

[2.48] “Physical verification,” [Online]. Available: https://en. wikipedia.org/wiki/Physical_verification, accessed Sep. 23, 2016.

[2.49] H. Salmani and M. Tehranipoor, “Vulnerability analysis of a circuit layout to hardware Trojan insertion,” IEEE Trans. Information Forensics Security, Vol. 11, no. 6, pp. 1214–25, Jan. 2016.

[2.50] K. Xiao, D.Forte, and M.Tehranipoor, “A novel built-in self authentication technique to prevent inserting hardware Trojans,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., Vol. 33, no. 12, pp. 1778–91, Dec. 2014.

[2.51] D. Saha, “A resilient authentication and Trojan detection technique for hard IP,” Proc. 2nd Int. Conf. Communication, Computing & amp Security, Vol. 6, pp. 24–30, 2012.

[2.52] X. Zhang and M. Tehranipoor, “RON: an on-chip ring oscillator network for hardware Trojan detection,” Proc. Des. Auto. Test Europe, pp. 1–6, 2011.

[2.53] J. Rajendran, V. Jyothi, O. Sinanoglu, and R. Karri, “Design and analysis of ring oscillator based design-for-trust technique,” Proc. IEEE 29th VLSI Test Symp., pp. 105–10, 2011.

[2.54] N. Karimian, F. Tehranipoor, M. Rahman, S. Kelly, and D. Forte, “Genetic algorithm for hardware Trojan detection with Ring Oscillator Network (RON),” Proc. IEEE Symp. Tech. Homeland Security, pp. 1–6, 2015.

[2.55] J. Li and J. Lach, “At-Speed delay characterization for IC authentication and Trojan Horse detection,” Proc. IEEE Int. Workshop Hardware Oriented Security Trust, pp. 8–14, 2008.

[2.56] Y. Alkabani, “Trojan immune circuits using duality,” Proc. 15th Euro micro Conf. Digital System Design, pp. 177–84, 2012.

[2.57] A. Das, G. Memik, J. Zambreno, and A. Choudhary, “Detecting/ preventing information leakage on the memory bus due to malicious hardware,” Proc. Des. Auto. Test Europe, pp. 861–66, 2010.

[2.58] B. Zhou et al. “Cost efficient acceleration of hardware Trojan detection through fan-out cone analysis and weighted random pattern technique,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., Vol. 35, no. 5, pp. 792–805, Jul. 2015.

[2.59] H. Salmani, M. Tehranipoor, and J. Plusquellic, “A novel technique for improving hardware Trojan detection and reducing Trojan activation time,” IEEE Trans. Very Large Scale Integr. Syst., Vol. 20, no. 1, pp. 112–25, Jan. 2011.

[2.60] F. Imeson, A. Emtenan, S. Garg, and M. V. Tripunitara, “Securing computer hardware using 3D integrated circuit (IC) technology and split manufacturing for obfuscation,” Proc. 22nd USENIX Conf. Security, pp. 495–510, 2013.

[2.61] J. Rajendran, O. Sinanoglu, and R. Karri, “Is split manufacturing secure?” Proc. Des. Auto. Test Europe, pp. 1259–64, 2013.

[2.62] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, “Trojan detection using IC fingerprinting,” Proc. IEEE Symp. Security Privacy, pp. 296–310, 2007.

[2.63] S. Bhasin, J. Danger, S. Guilley, X. T. Ngo, and L. Sauvge, “Hardware Trojan Horses in cryptographic IP cores,” Proc. 10th Workshop Fault Diagnosis Tolerance in Cryp., pp. 15–29, 2013.

[2.64] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia, “MERO: a statistical approach for hardware Trojan detection,” Proc. Workshop Cryp. Hardware Embedded Syst., pp. 396–410, 2009.

[2.65] R. S. Chakraborty, S. Paul, and S. Bhunia, “On Demand transparency for improving hardware Trojan detectability,” Proc. IEEE Int. Workshop Hardware Oriented Security Trust, pp. 48–50, 2008.

[2.66] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, “Trojan detection using IC fingerprinting,” Proc. IEEE Symp. Security Privacy, pp. 296–310, 2007.

[2.67] J. Aarestad, D. Acharyya, and J. Plusquellic, “Detecting Trojans through leakage current analysis using multiple supply pad IDDQs,” IEEE Trans. Inf. Forensics Security, Vol. 5, no. 4, pp. 893–904, Jul. 2010.

[2.68] S. Wei and M. Potkonjak, “Scalable hardware Trojan diagnosis,” IEEE Trans. Very Large Scale Integ. Syst., Vol. 20, no. 6, pp. 1049–57, Jun. 2012.

[2.69] Y. Cao, C. H. Chang, and S. Chen, “A cluster based distributed active current sensing circuit for hardware trojan detection,” IEEE Trans. Inf. Forensics Security, Vol. 9, no. 12, pp. 2220–31, Sep. 2014.

[2.70] Y. Jin and Y. Makris, “Hardware Trojan detection using path delay fingerprint,” Proc. IEEE Int. Workshop Hardware Oriented Security Trust, pp. 51–7, 2008.

[2.71] K. Xiao, X. Zhang, and M. Tehranipoor, “A clock sweeping technique for detecting hardware Trojans impacting circuits delay,” IEEE Trans. Des. Test Comput., Vol. 30, no. 2, pp. 26–34, Apr. 2013.

[2.72] S. Shekarian and M. Zamani, “Improving hardware Trojan detection by retiming,” Microprocessors Microsystems, Vol. 39, no. 3, pp. 145–56, May. 2015.

[2.73] O. Soll, T. Korak, M. Muehlberghuber, and M. Hutter, “EM based detection of hardware Trojans on FPGAs,” Proc. IEEE Int. Symp. Hardware Oriented Security Trust, pp. 84–7, 2014.

[2.74] F. Koushanfar and A. Mirhoseini, “A unified framework for multimodal submodular integrated circuits Trojan detection,” IEEE Trans. Information Forensics Security, Vol. 6, no. 1, pp. 162–74, Dec. 2010.

[2.75] S. Narasimhan et al., “Hardware Trojan detection by multiple parameter side-channel analysis,” IEEE Trans. Comput., Vol. 62, no. 11, pp. 2183–95, Nov. 2013.

[2.76] N. Nowroz, K. Hu, F. Koushanfar, and S. Reda, “Novel techniques for high sensitivity hardware Trojan detection using thermal and power maps,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., Vol. 33, no. 12, pp. 1792–1805, Dec. 2014.

[2.77] R. Rad, J. Plusquellic, and M. Tehranipoor, “A sensitivity analysis of power signal methods for detecting hardware Trojans under real process and environmental conditions,” IEEE Trans. Very Large Scale Integ. Syst., Vol. 18, no. 12, pp. 1735–44, Oct. 2009.

[2.78] Y. Alkabani, and F. Koushanfar, “Consistency based characterization for IC Trojan detection,” Proc. Int. Conf. Comput. Aided Design, pp. 123–27, 2009.

[2.79] H. Salmani, and M. Tehranipoor, “Layout Aware switching activity localization to enhance hardware Trojan detection,” IEEE Trans. Inf. Forensics Security, Vol. 7, no. 1, pp. 76–87, Aug. 2011.

[2.80] S. Wei and M. Potkonjak, “Self consistency and consistency based detection and diagnosis of malicious circuitry,” IEEE Trans. Very Large Scale Integr. Syst., Vol. 22, no. 9, pp. 1845–53, Sep. 2013.

[2.81] A. Nejat, S. M. H. Shekarian, and M. S. Zamani, “A study on the efficiency of hardware Trojan detection based on path delay fingerprinting,” Microprocessors Microsystems, Vol. 38, no. 3, pp. 246–52,May. 2014.

[2.82] C. Bao, D. Forte, and A. Srivastava, “On reverse engineering based hardware Trojan detection” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., Vol. 35, no. 1, pp. 49–57, Oct. 2015.

[2.83] S. Narasimhan, X. Wang, D. Du, R. S. Chakraborty, and S. Bhunia, “TeSR: a robust temporal self referencing approach for hardware Trojan detection,” Proc. IEEE Int. Symp. Hardware Oriented Security Trust, pp. 71–4, 2011.

[2.84] A. Davoodi, L. Min, and M. Tehranipoor, “A Sensor Assisted Self Authentication framework for hardware Trojan detection,” IEEE Trans. Design & Test, Vol. 30, no. 5, pp. 74–82, Mar. 2013.

[2.85] M. Banga and M. S. Hsiao, “A novel sustained vector technique for the detection of hardware Trojan,” Proc. 22nd Int. Conf. VLSI Design, pp. 327–32, 2009.

[2.86] H. Li and Q. Liu, “Hardware Trojan detection acceleration based on word level statistical properties management,” Proc. Int. Conf. Field Programmable Technology, pp. 153–60, 2014.

[2.87] L. Kim and J. Villasenor, “A system-on-chip bus architecture for thwarting integrated circuit Trojan horses,” IEEE Trans. Very Large Scale Integr. Syst., Vol. 19, no. 10, pp. 1921–26, Aug. 2011.

[2.88] S. Narasimhan, W. Yueh, X. Wang, S. Mukhopadhyay, and S. Bhunia, “Improving IC security against Trojan attacks through integration of security monitors,” IEEE Trans. Des. Test Comput., Vol. 29, no. 5, pp. 37–46, Oct. 2012.

[2.89] L. W. Kim, and J. Villasenor, “Dynamic function replacement for system-on-chip security in the presence of hardware based attacks,” IEEE Trans. Reliability, Vol. 63, no. 2, pp. 661–75, Apr. 2014.

[2.90] C. Bao, D. Forte, and A. Srivastava, “Temperature tracking: toward robust run time detection of hardware Trojans,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., Vol. 34, no. 10, pp. 1577–85, Oct. 2015.

[2.91] X. T. Ngo, J. Danger, S. Guilley, Z. Najm, and O. Emery, “Hardware property checker for Run Time hardware Trojan detection,” Proc. European Conf. Circuit Theory and Design, 2015, pp. 1–4. DOI: 10.1109/ ECCTD.2015.7300085.

[2.92] Xuan Thuy Ng et al., “Integrated sensor: a backdoor for hardware Trojan insertions?,” Proc. Digital System Design, 2015, pp. 415–22. DOI: 10.1109/DSD.2015.119.

[2.93] “Managing the risks of counterfeiting in the information technology” [Online]. Available: http://www.agmaglobal.org/press_events/press_docs/Counterfeit_WhitePaper_Final.pdf, accessed Dec. 14, 2016.

[2.94] “Innovation is at risk as semiconductor equipment and materials industry loses up to $4 billion annually due to IP infringement” [Online]. Available: http://semi.org/en/innovation-risk-losses-4-billion-annually-due-ip-infringement, accessed Dec. 14, 2016.

[2.95] “Estimating the global economic and social impacts of counterfeiting and piracy,” [Online]. Available: http://www.illicittrademonitor.com/reports/article/estimating-the-global-economic-and-social-impacts-of-counterfeiting-and-piracy/, accessed Dec. 14, 2016.

[2.96] B. Colombier, and L. Bossuet, “Survey of hardware protection of design data for integrated circuits and intellectual properties,” IET Comput. Digital Techn. Vol. 8, pp. 274-287, 2014.

[2.97] Department of Defense, Defense Science Board (DSB) “Study on High Performance Microchip Supply” [Online]. Available: ttp://www.aoq.osd.mil/dsb/reports/ADA435563.pdf, accessed Dec.14, 2016.

[2.98] X. Zhuang, T. Z. Hsien-Hsin, S. Lee, and S. Pande, “Hardware assisted control flow obfuscation for embedded processors,” in Proc. Int. Conf. Compilers, Archit., Synth. Embedded Syst., pp. 292-302, 2004.

[2.99] G. Sumathi, L. Srivani, D. Thirugnana Murthy, Anish Kumar, K. Madhusoodanan, and S.A.V. Satya Murty, “Structural Modification based Netlist Obfuscation Technique for PLDs,” in Proc. IEEE Int. Conf. WiSPNET, Chennai, pp. 1418-1423, Mar. 2016.

[2.100] A.R. Desai, et al. “Interlocking Obfuscation for Anti-Tamper Hardware,” in Proc. 8th Annual Workshop Cyb. Sec. & Inf. Intelligence Research, Jan. 2013.

[2.101] M. Brzozowski, and V. N. Yarmolik, “Obfuscation as intellectual rights protection in VHDL language,” in Proc. 6th Int. Conf. Comput. Inf. Syst., pp. 337-340, Jun. 2007.

[2.102] U. Meyer-Base, E. Castillo, and G. Botello, “Intellectual property protection (IPP) using obfuscation in C, VHDL, and Verilog coding,” in Proc. SPIE Independent Component Analyses, Wavelets, Neural Networks, Biosystems, and Nanoengineering IX, vo1. 8058, Apr. 2011.

[2.103] V. V. Sergeichik, A. A. Ivaniuk, and Chang, “Obfuscation and watermarking of FPGA designs based on constant value generators,” in Proc. 14th Int. Symp. Integrated Circuits, pp. 608-611, Dec. 2014.

[2.104] M. Kainth, L. Krishnan, C. Narayana, S. G. Virupaksha, and R. Tessier, “Hardware assisted code obfuscation for FPGA soft microprocessors,” in Proc. Design Auto. Test Eur., pp. no. 127-132, Mar. 2015.

[2.105] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, “Security analysis of integrated circuit camouflaging,” in Proc. ACM SIGSAC conf. Computer & communications security, pp. no. 709-720, Nov. 2013.

[2.106] V. Immler, M. Hennig, L. Kürzinger, and G. Sigl, “Practical Aspects of Quantization and Tamper Sensitivity for Physically Obfuscated Keys,” in Proc. Third Workshop on Cryptography and Security in Computing Systems, Pages 13-18, 2016.

[2.107] A. Roy, F. Koushanfar, and I. L. Markov, “Extended abstract: Circuit CAD tools as a security threat,” in Proc. IEEE Workshop on Hardware Oriented Security Trust, pp. 65-66, 2008.

[2.108] G. Qu, and Y. Yuan, “Design things for the internet of things an EDA perspective,” in Proc. IEEE/ACM Int. Conf. Computer Aided Design, pp. 41-416, 2014.

[2.109] “N.S.A. Devises Radio Pathway Into Computers” [Online]. Available: http://www.nytimes.com/2014/01/15/us/nsa-effort-pries-open-computers-not-connected-to-internet.html?_r=0, accessed Jul. 08, 2016.

[2.110] “Time To Refocus on The EMP Threat” [Online]. Available: http://www.defensenews.com/story/defense/commentary/2015/08/18/tim e-refocus-emp-threat/31915021/, accessed Jul. 08, 2016.

[2.111] A. Moradi, D. Oswald, C. Paar, and P. Swierczynski, “Side-channel attacks on the bitstream encryption mechanism of Altera Stratix II,” in Proc. ACM/SIGDA Int. Symp. FPGAs, pp. 91-100, 2013.

[2.112] J. B. Note, and E. Rannaud, “From the bitstream to the netlist,” in Proc. ACM/SIGDA Int. Symp. FPGAs, pp. 264-264, 2008.

[2.113] F. Benz, A. Seffrin, and S. A. Huss, “Bil: A tool chain for bitstream reverse engineering,” in Proc. 22nd Int. Conf. Field Program. Logic Appl., pp. 735-738, 2012.

[2.114] “Design Security in Nonvolatile Flash and Antifuse FPGAs” [Online]. Available: http://www.microsemi.com/document-portal/doc_view/131564-designsecurity-wp, accessed Jul. 08, 2016.

[2.115] “Stopping Hardware Trojans in Their Tracks” [Online]. Available: http://spectrum.ieee.org/semiconductors/design/stopping-hardware-trojans-in-their-tracks, accessed Jul. 08, 2016.

[2.116] G. Sumathi, L. Srivani, D. Thirugnana Murthy, K. Madhusoodanan, and S.A.V. Satya Murty, “A Review on HT attacks in PLD and ASIC designs with Potential Defense Solutions,” in IETE Technical Review. (DOI: 10.1080/02564602.2016.1246385)

[2.117] J. A. Roy, F. Koushanfar, and I. L. Markov, “EPIC: Ending piracy of integrated circuits,” in Proc. Des. Autom. Test Eur., 2008, pp. 1069-1074.

[2.118] J. A. Roy, F. Koushanfar, and I. L. Markov, “Ending piracy of integrated circuits,” Computer, vol. 43, no. 10, pp. 30-38, 2010.

[2.119] R. Torrance and D. James, “The state-of-the-art in semiconductor reverse engineering,” in Proc. 48th AC M/EDAC/IEEE Design Autom. Conf., Jun. 2011, pp. 333–338.

[2.120] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, “Security Analysis of Logic Obfuscation,” in Proc. IEEE/ ACM Design Automation Conf., pp. 83-89, May. 2012.

[2.121] B. Colombier, L. Bossuet, and D. Hély, “From secured logic to IP protection,” Microprocessors and Microsystems, In Press, Corrected Proof Note to users, Available online 26 February 2016.

[2.122] J. Rajendran, et al. “Fault analysis based logic encryption,” IEEE Trans. Comput., vol. 64, no. 2, pp. 410-424, 2015.

[2.123] J. Zhang, “A Practical Logic Obfuscation Technique for Hardware Security,” IEEE Trans. VLSI Syst., vol. 24, no. 3, pp. 1193-1197, 2015.

[2.124] X. Wang et al., “Secure and Low-Overhead Circuit Obfuscation Technique with Multiplexers,” in Proc. 26th Great Lakes Symposium on VLSI, pp. 133-136, 2016.

[2.125] R. S. Chakraborty and S. Bhunia, “HARPOON: An obfuscation based SoC design methodology for hardware protection,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., vol. 28, no. 10, pp. 1493-1502, 2009.

[2.126] R. S. Chakraborty and S. Bhunia, “RTL hardware IP protection using key based control and data flow obfuscation,” in Proc. Int. Conf. on VLSI Design, pp. 405-410, Jan. 2010.

[2.127] J. B. Wendt, and M. Potkonjak, “Hardware obfuscation using PUF based logic,” in Proc. IEEE/ACM Int. Conf. Computer Aided Design, pp. 270-277, 2014.

[2.128] D. Li et al., “Hardware IP Protection through Gate Level Obfuscation,” in Proc. 14th Int. Conf. Computer Aided Design and Computer Graphics, pp. 186-193, 2015.

[2.129] R. Chakraborty, and S. Bhunia, “Security against Hardware Trojan through a Novel Application of Design Obfuscation,” in Proc. IEEE/ ACM Int. Conf. on Computer Aided Design, pp. 113-116, Nov. 2009.

[2.130] J. Frey, and Q. Yu, “Exploiting state obfuscation to detect hardware trojans in NoC network interfaces,” in Proc. IEEE 58th Int. Midwest Symp. Circuits and Systems, pp. 1-4, 2015.

[2.131] A. Nejat, D. Hely, and V. Beroulle, “Facilitating side channel analysis by obfuscation for Hardware Trojan detection,” in Proc. 10th Int. Design & Test Symp., pp. 129-134, 2015.

3

Detection Mechanism for HTs using Delay Signatures

The present chapter discusses on delay signatures at different process corners based HT detection technique for SRAM based FPGAs.

3.1 Introduction

A typical PLD design cycle includes programming using HDL, synthesis (netlist generation), simulation, mapping to technology, PAR, generation of configuration bitstream and finally programming the target device. In order to prevent reverse engineering/ manipulation of FPGAs, a few vendors offer bitstream encryption feature. However, Moradi et al. [3.1] demonstrated a successful attack against the bitstream encryption engine i.e. they tampered the bitstream, but it is very difficult to perform such attacks on one time programmable FPGAs. Hence, to detect HTs inserted by reverse engineering SRAM based FPGAs, the DSDPC: delay signatures at different process corners based HT detection technique is proposed. For this, the delay profile of original netlist has to be stored. At regular intervals, the field extracted netlist profile is compared with the stored profile to identify tampering. Since the simulation results of EDA tools are repetitive, mismatches among delay signatures clearly reveal the HT presence. The proposed method improves detection efficiency by using process corners.

3.2 Methodology

The growing use of FPGAs in critical applications has urged designers to indulge security right from concept level. In this context, security refers to protecting HDL code or IP cores that involve a high development cost and contain sensitive information, mapped to a FPGA device. The following subsections explain how delay signatures and process corners are related to each other and detail the proposed method with its flow diagram.

3.2.1 Path Delay and Process Corners

This section presents the mathematical relationship between three important IC parameters such as supply voltage (V_{dd}), operating temperature (T) and circuit path delay (D) as summarized [3.2]. As per alpha-power law, the delay of a cell is expressed as

$$\text{Delay} \propto [(C_{out} . V_{dd}) / I_d] \tag{3.1}$$

where C_{out} is output load capacitance, V_{dd} is supply voltage and I_d is drain current. From Eqn. 3.1, it is clear that the delay is inversely proportional to the drain current I_d and is expressed as

$$I_d \propto [\mu(T) (V_{dd} — V_{th}(T))^{\alpha}] \tag{3.2}$$

where μ is the mobility, V_{th} is the threshold voltage and α is a carrier velocity saturation index. The temperature (in Kelvin) dependence of I_d is due to those of the mobility μ and the threshold voltage V_{th}. The temperature dependence of these parameters is as

$$\mu(T) = [\mu(T_R) (T_R / T)^m] \tag{3.3}$$

$$V_{th}(T) = V_{th}(T_R) — \kappa(T — T_R) \tag{3.4}$$

where, T_R is the room temperature in Kelvin, m and κ are small positive constants. From Eqn. 3.3 and Eqn. 3.4, it is clearly evident that both mobility and threshold voltage decrease with increasing temperature. However, as Eqn. 3.2 shows, they affect the drain current in opposite ways: lower mobility decreases the drain current, but lower threshold voltage increases the drain current. The final drain current is determined by the trend that dominates the drain current at a given voltage and temperature pair. Hence, at high voltages, the mobility determines the drain current but at low voltages, the threshold voltage determines the drain current. In other words, at high voltages, the delay value increases as temperature increases whereas at low voltages the delay value decreases with increase in temperature. Thus, the delay value increases or decreases with increasing temperature depending on the magnitude of V_{dd}. This is the inverted temperature dependence (ITD) phenomenon. The voltage where temperature dependence reverses (or inverts) is called the crossover voltage, the zero-temperature coefficient voltage, or the inversion voltage. However, delay values of devices fabricated using 90 nm technology and below decrease as the supply voltage increases and operating temperature decreases which reveals no ITD effects. In this work, we have performed experiments using 90nm FPGA device. So, there is no need to include the ITD effect. In order to understand the effect of process corners on delay values, the following paragraph defines the parameters of process corners with respect to their minimum, typical and maximum values.

Process corners represent the extremes of the device parameter variations within which a circuit that has been etched onto the wafer must function correctly. For example, process corner (P, V, T) denotes the parameters P- process, V-voltage and T-temperature. For each PVT parameter (X), X_{max} and X_{min} denote respective maximum and minimum values. The values of parameters at different process corners such as slow and fast corners are (P_{max}, V_{min}, T_{max}) and

(P_{min},V_{max},T_{min}) respectively. Using the temperature and voltage dependency of circuit path delay, at fast process corners i.e. at maximum voltage and minimum temperature, circuit takes less path delay whereas at slow process corners i.e. at minimum voltage and maximum temperature condition, the delay value is high. Since the simulation results of EDA tools are repetitive, mismatches among delay signatures clearly reveal the HT presence. The proposed method improves detection efficiency by using process corners.

Aging impacts on FPGA delays are considered to be an important degree-of-freedom, other than P-V-T, i.e. transistor aging, in particular, is one of the most important reliability challenges at nano-scale [3.3]. It happens on a relatively long time period, where the circuit delay degrades (i.e. increases) continuously over the operational lifetime leading to timing failures. The degradation caused by aging mechanisms is related to several different parameters such as temperature, supply voltage and usage [3.4]. To detect if any field tampering of FPGAs, here it is not proposed to measure the delay signatures directly from FPGAs, i.e. silicon. This method suggest to read-back the bitstream from FPGA and to regenerate the netlist using bitstream. Finally, the delay signatures of extracted netlist are captured by the software tool. Hence, aging impact on the delay measurement of field device is not considered.

3.2.2 Delay Signature at Different Process Corners based HT Detection for FPGA

Any FPGA design starts with HDL code (may include IP) and undergoes synthesis, mapping, PAR and bit file generation using the vendor specific EDA tools. To prove design correctness, the HDL code undergoes simulation at pre-synthesis, post synthesis and post PAR. The netlist is generated in the synthesis stage whereas bitstream is generated after PAR. Finally,

the bitstream is used to program the target device and deployed in the field. The read-back option is more common for designs that are in orbit. Single event upset (SEU) events can change the logic on the device. Hence, many FPGA vendors allow read-back feature that can directly read out the configuration of the device either through JTAG, ICAP or another similar bitstream programming interface. Hence, a successful read-back attack directly acquires the bitstream from the functioning device. Being able to detect the changes to FPGA allows a portion or the entire reconfigurable logics of the device to be reprogrammed in a timely manner. This can be achieved using “read-back operation” in iMPACT tool for Xilinx devices or “EXAMINE operation” in Quartus tool for Altera devices or “FlashLock feature” in Libero SoC tool for Actel/ Microsemi devices. For example, by launching iMPACT tool, Xilinx FPGAs support the read-back & verify operation and it generates the read-back files such as iMPACT.bin (binary) and iMPACT.rbd (ASCII 1's and 0's). So far, the ideal design phase is elaborated.

Figure 3.1 shows the complete flow diagram representing the ideal design phase, tampering phase and the detection phase. If an adversary intends to manipulate the design, can extract the bit file from the field device, in turn generate the netlist file. Once netlist of any design is extracted, insertion of HTs is easier. This modified design can undergo further PAR and HT inserted bitstream is generated to reprogram the field device.

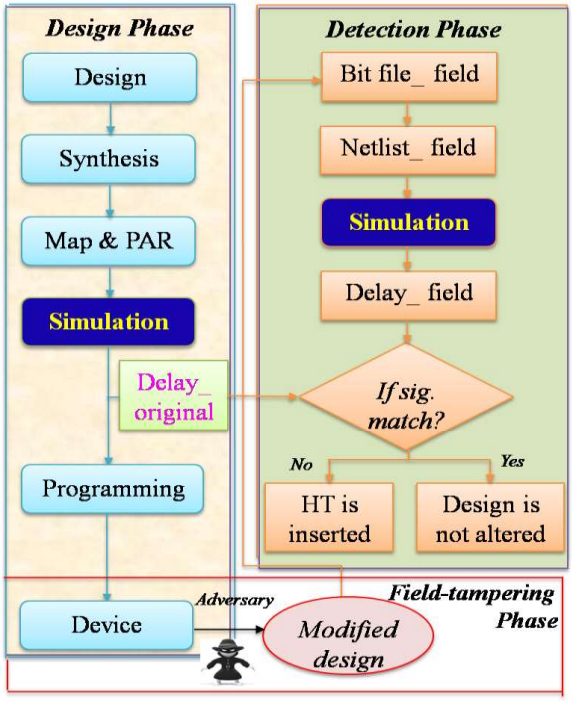


Figure 3.1: Flow diagram of HT detection using delay profile for FPGAs

Most of the Xilinx applications cover two families of FPGAs such as high-volume Spartan series and high-performance Virtex series. Xilinx FPGAs are SRAM technology based and volatile. The bitstreams are stored in a separate programmable read only memory (PROM) chip connected to the FPGA. During every power up, the bitstream configures the FPGA. HDL synthesis by universal EDA tools generates vendor independent netlist file (e.g. edn) which is then converted to Xilinx netlist ngd file. The technology mapped and PAR file is ncd file using which programmable bitstream file is generated. In turn, it has a tool called "ncd2xdl" which generates xdl file from ncd file. The xdl file format is a clear-text representation of the ncd file without much detail about the static, non-configurable internal architecture of the FPGA to become a fully-fledged netlist. Note et al. [3.5] proposed an algorithm called "Debit" which to an extent able to convert bitstream into xdl file and in turn generate the netlist file. If the design is altered during the tampering phase, the HT detection approach using delay profiles is proposed.

In this method, initially the delay signatures of original HDL code are collected. During periodic testing, the delay signatures of field extracted netlist are collected and compared with original signatures. Any mismatch among delay signatures reveals the tampering at field. As delay signatures need not be stored at field operating condition, which might be considered to be unsecure if attacker has control over it, the practical adoption of the proposed method is feasible. In general, an adversary inserts HTs that are likely to cause little or no changes, thereby making it difficult to discover their presence, effectively resulting in "false negatives." To effectively reduce false negatives and improve the efficiency of delay signatures based HT detection method, the delay signatures at different process corners technique is incorporated as shown in Figure 3.2.

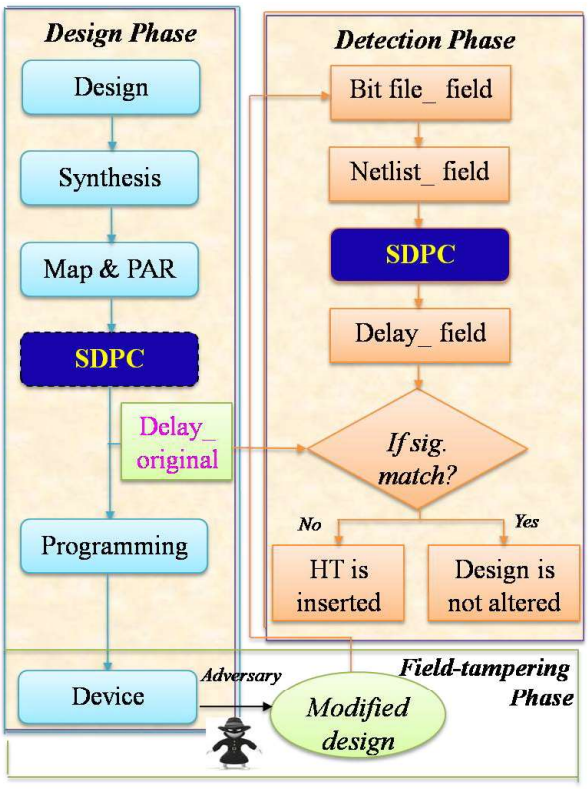


Figure 3.2: Flow diagram of DSDPC method for Xilinx device.

SDPC*- Simulation at different process corners

In DSDPC, the simulations are performed at three different process corners defined in Section 3.2.1 namely slow, typical and fast process corners. Without considering the ITD effect, slow process corner is performed at (low voltage, high temperature) condition, whereas fast corner is at (high voltage, low temperature) condition. Since the delay value at slow process corner is higher than fast process corner, there is a relative increase in the delay difference between with and without HT files from slow to fast corners, which in turn increases the HT detection efficiency. The simulation results of delay signatures with and without HT are discussed in the following section.

3.3 Results and Discussion

In order to verify the effectiveness of delay signatures based HT detection method, timing analysis was performed using ISCAS'89 sequential benchmark circuit "s9234" implemented in 90nm Xilinx Spartan-3s100evq100-4 device using Xilinx ISE 12.2 (free version). The HDL code of "s9234" benchmark circuit has 36 inputs, 39 outputs, 211 D flip-flops, 3570 inverters and 2027 total gates. It is clearly evident that with 36 numbers of inputs, 2^{36} possible combinations of test vectors have to be generated for complete test suit. Also, the entire 39 outputs have to be monitored during experimentation. To ease the process of simulation and to generate complete set of test vectors, combinational circuitry is introduced at both input and output ports to reduce them to 9 inputs and 8 outputs circuitry. Thereby, we only have to generate 512 numbers of test vectors and to monitor 8 outputs. As shown in Figure 3.3, the number of inputs and outputs are reduced to 9 and 8 using a simple combinational logic. After synthesis, the top module consumed 328 slices, 204 flip flops and 598 look up tables of Xilinx resources and the device utilization is 34% (refer Table 3.1).

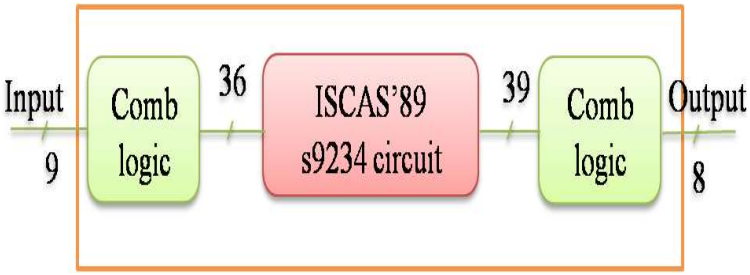


Figure 3.3: Modified "s9234" circuit

Table 3.1: Resource utilization percentage of "s9234" circuit

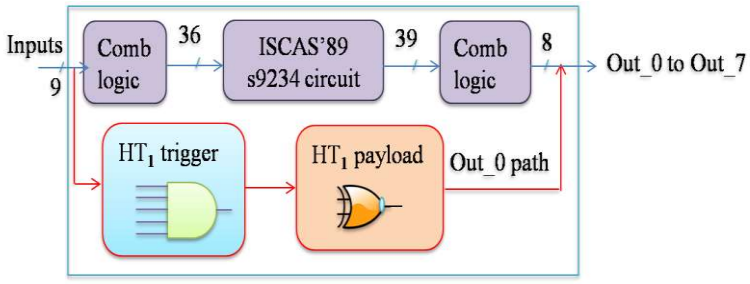
Resources	Without HT
Slices	328/960 – 34%
Flip-flops	204/1920 – 10%
4 input LUTs	589/1920 – 30%
IOs	17
Bonded IOBs	17/66 – 25%
GCLKs	1/24 – 4%

In general, to check the design functionality at extreme operating conditions, option to simulate at different process corners is available with EDA tools. As elaborated in Section 3.2.1, circuit path delay and process corners are internally related. As per the software manual, most of the EDA tools perform worst case delay analysis at slow process corner with maximum voltage and low temperature settings. The operating voltage and temperature ranges of the selected device are 1.14 V to 1.32 V and -40 °C to 100 °C respectively. Using ISim from Xilinx, timing simulations were performed at the following process corners: slow corner at (1.14 V and 100 °C), typical or nominal corner at (1.2 V and 25 °C) and fast corner at (1.32 V and -40 °C). The measured delays of two randomly selected paths i.e. out_0 and out_1 are listed in Table 3.2.

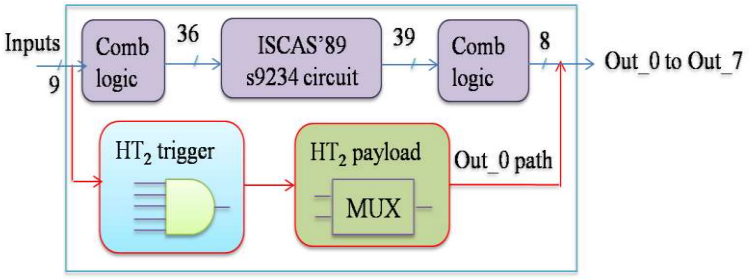
Table 3.2: Path delays at slow, typical and fast process corners

Data path	Out_0	Out_1
Delay at Slow process corner	5169 ps	2759 ps
Delay at Typical process corner	4833 ps	2203 ps
Delay at Fast process corner	4007 ps	1974 ps

The process of extracting netlist from bitstream is highly depending on third-party propriety tool called as “debit tool”. As we don’t have access to this tool and to demonstrate the scenario of bitstream tampering at field condition, two different HT circuitries are manually inserted at RTL level. However, the ideal methodology explains as per the flowchart shown in Figure 3.2.



(a)



(b)

Figure 3.4: Insertion of HTs in out_0 path (a) Insertion of HT_1 circuitry and (b) Insertion of HT_2 circuitry

To demonstrate the effect of HT insertion either at bitstream or netlist level in field conditions, two different HT circuitries, namely HT_1 and HT_2, are manually inserted as shown in Figure 3.4, at RTL level in the same top module. HT_1 is designed using 5 inputs AND as its trigger mechanism and a combination of XOR gate with inverter as buffer payload whereas HT_2 has the same trigger circuit with 2x1 multiplexer as its payload circuit. For experiments, the HT circuits HT_1 and HT_2 are inserted in out_0 path separately. Table 3.3 lists the device utilization percentage of the top module with HT circuits. HT1 uses 3 slices of Xilinx resources whereas HT2 uses 5 slices and their respective resource utilizations are 0.3% and 0.5%.

Table 3.3: Device utilization percentage with and without HTs

Resources	**Without HT**	**With HT_1**	**With HT_2**
Slices	328/960 – 34%	331/960 –34%	333/960 –34%
Flip-flops	204/1920 – 10%	204/1920 – 10%	204/1920 – 10%
4 input LUT	589/1920 – 30%	600/1920 – 31%	600/1920 – 31%
IOs	17	17	17
Bonded IOB	17/66 – 25%	17/66– 25%	17/66– 25%
GCLKs	1/24 – 4%	1/24 – 4%	1/24 – 4%

In this research, the concept of "delay signatures at different process corners" is used to increase the delay difference between with and without HT circuits. Upon receiving the trigger i.e. output of trigger logic becomes high, the HT_1 payload buffers the value at out_0 while HT_2 asserted logic high at out_0. The delay signatures of HT inserted circuits are also measured using ISim at slow, typical and fast process corners. For convenience, the affected path (i.e. out_0) and non-affected path (i.e. out_1) are listed here. Increase in the path delay reflects the HT affected paths.

Table 3.4-3.6 show the path delays at slow process corner (voltage = 1.14 V, temperature = 100 °C), typical process corner (voltage = 1.2 V, temperature = 25 °C) and fast process corner (voltage = 1.32 V, temperature = -40 °C), respectively.

Table 3.4: Path delay with and without HT at slow process corner

Data path		**Out_0**	**Out_1**
Path delay without HT		5169 ps	2759 ps
HT_1	Path Delay	5667 ps	2801 ps
	Increased delay	*498 ps*	*42 ps*
HT_2	Path Delay	5686 ps	2803 ps
	Increased delay	*517 ps*	*44 ps*

Table 3.5: Path delay with and without HT at typical process corner

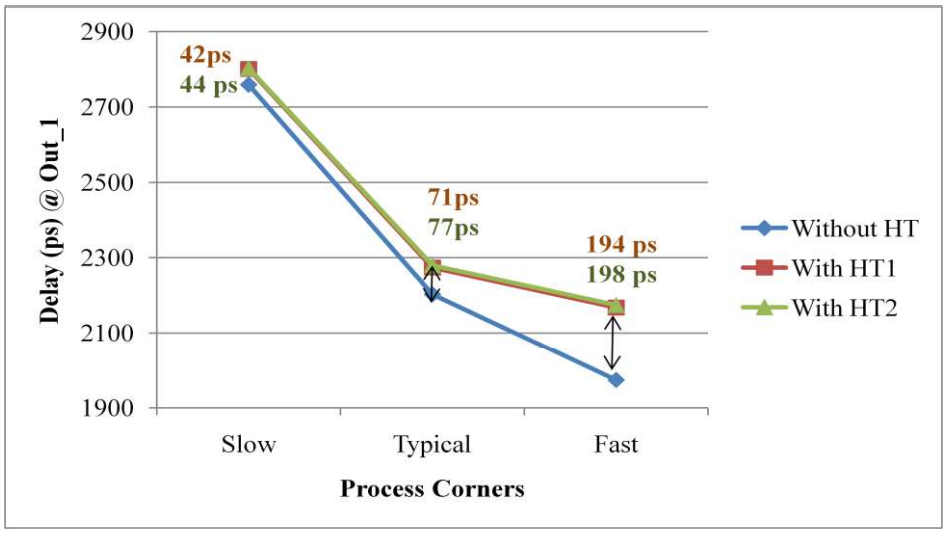
Data path		**Out_0**	**Out_1**
Path delay without HT		4833 ps	2203 ps
HT_1	Path Delay	5387 ps	2274 ps
	Increased delay	*554 ps*	*71 ps*
HT_2	Path Delay	5414 ps	2280 ps
	Increased delay	*581 ps*	*77 ps*

Table 3.6: Path delay with and without HT at fast process corner

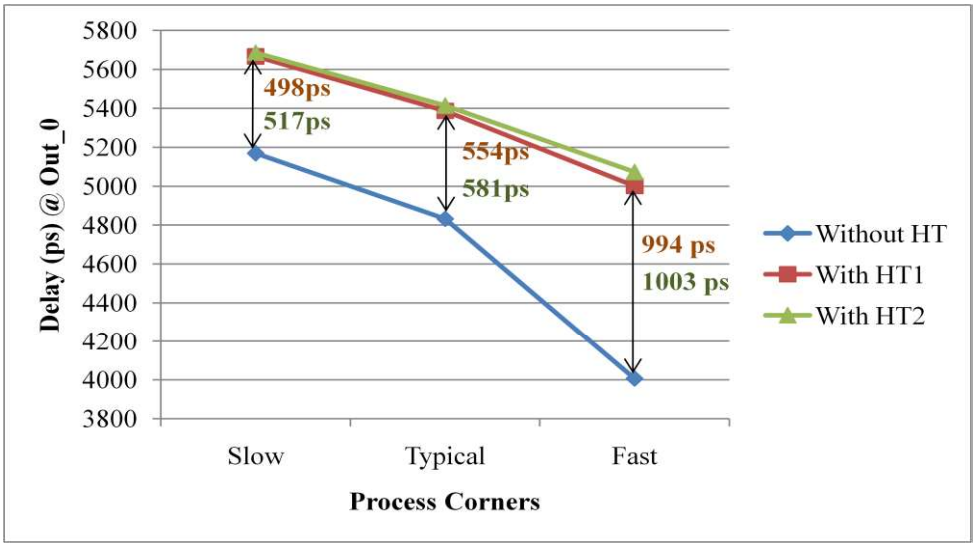
Data path		**Out_0**	**Out_1**
Path delay without HT		4007 ps	1974 ps
HT_1	Path Delay	5001 ps	2168 ps
	Increased delay	*994 ps*	*194 ps*
HT_2	Path Delay	5073 ps	2175 ps
	Increased delay	*1003 ps*	*201 ps*

As expected, the HT free circuit delay values decreased from slow (5169 ps) to fast process corner (4007 ps) at all the output paths. Once the HT is inserted, the modified design

undergoes synthesis, map and PAR. So there will be new alignment, eventually it is reflected in all the signal path delays. It is observed that the difference in delay values between HT inserted and HT free circuit increased from slow to fast process corner (refer Figure 3.5a and Figure 3.5b).



(a)



(b)

Figure 3.5: Delay signatures at different process corners. Delay difference at (a) Out_1 path and (b) Out_0 path

However, the delay difference of HT affected path i.e. out_0 is higher than non-affected path i.e. out_1 as shown in Figure 3.5a and Figure 3.5b, respectively. Using this property, the proposed DSDPC approach supports in detecting the HT and to increase the delay difference between with and without HT circuitries from slow to fast process corners which, in turn, enhances the HT detection efficiency.

3.4 Advantages

To highlight the advantages of the proposed approach, this method is

1. Non-destructive,
2. Does not require to trigger HT circuitry and
3. Non-invasive i.e. requires no extra circuitry for HT detection.

The present chapter details the delay profile based HT detection mechanism for PLDs. In addition to the proposed HT detection mechanism, implementation of the device dependent secret key generation module in the PLDs is one of the best solutions to avoid the side-channel attacks and various invasive attacks. The PUFs are widely used to address them and they are discussed in chapter 4.

3.5 Summary

Most of the digital designs in recent years are surfaced with FPGA devices. Thus, it is necessary to ensure the security of those devices i.e. to ensure the security of IP codes of the design, EDA tools used in design phase and finally to ensure tamper-free FPGAs from fabrication foundry. Among all, the thread called “bitstream tampering” that enables the adversary to reverse engineer the bitstream and to alter the original design file is considered for

this study. Though many anti-tamper techniques are available to prevent such attacks, they are invasive and consume extra logic and power. In this chapter, the DSDPC technique: delay signatures at different process corners based HT detection technique is successfully explored for SRAM based FPGAs, where delay signature of original netlist is compared with netlist extracted from field. Ideally the results of EDA tools are consistent; hence any deviation in delay signatures clearly reveals the tampering of design. Since the proposed method requires no extra logic to detect HTs, it considerably reduces the HT detection cost compared to other existing techniques. With advent of technology, advanced families of FPGAs are introduced in digital market by different vendors. To extent the proposed technique on advanced families of Xilinx FPGAs, it is expected that similar type of netlist extraction software/ tool might be developed for reverse engineering purpose in near future.

References:

[3.1] A. Moradi, A. Barenghi, T. Kasper, and C. Paar, "On the vulnerability of FPGA bitstream encryption against power analysis attacks: extracting keys from Xilinx Virtex-II FPGAs," in Proc. ACM Conf. Computer and Communications Security (CCS), pp. 111-124, 2011.

[3.2] A. Dasdan, and I. Hom, "Handling inverted temperature dependence in static timing analysis," in Proc. ACM Tran. Design Automation of Electronic Syst., vol. 11, no. 2, pp. 306-324, Apr. 2006.

[3.3] T. Nigam, K.-Y. Yiang, and A. Marathe, "Moores Law: Technology Scaling and Reliability Challenges," Microelectronics to Nanoelectronics: Materials, Devices & Manufacturability, p. 1, 2012.

[3.4] A. Amouri, et al., "Aging effects in FPGAs: an experimental analysis," in Proc. Int. Conf. Field Programmable Logic and Applications (FPL), pp. 1-4, 2014.

[3.5] B. Note, and E. Rannaud, "From the bitstream to the netlist," in Proc. ACM/SIGDA Int. Symp. FPGAs, pp. 264, 2008.

4

Hardware Obfuscation based Design Security Solution for PLDs

This chapter discusses the experiments carried out on hardware obfuscation based design security solution for PLDs, obfuscation metric calculations and the automation and integration of hardware obfuscation technique using scripting languages.

4.1 Introduction

Chapter-3 discussed the delay profile based HT detection mechanism. However, it is very clear that deriving single detection mechanism for all HT attacks is practically infeasible. Hence, this chapter discusses on hardware obfuscation techniques which ensure a certain level of design security by preventing against stealing of original design by analyzing and rebuilding during RE. That is, it increases RE complexity of HDL code or IP core or IC design. This, in turn, avoids the insertion of successful and hard-to-detect HTs. The security analysis of various obfuscation techniques and their application to the ASIC technology has been published over a decade [4.1-4.7]. With the expansion of the use of PLDs beyond commercial markets to internet of things, avionics, defense and nuclear applications, designs in PLDs take on the additional aspects of safety and national security. Besides, most of PLD-based critical applications attempt to preserve their indigenous designs as IPs to handle the fast obsolescence of PLDs and to upgrade with technology. However, the importance of logic obfuscation technique to PLD-based safety critical designs is not discussed yet. In this work, the experiments are carried out on the structural

modification based hardware obfuscation technique to PLDs using benchmark circuits. In particular, the following major contributions are made in this chapter such as

1. Most of the obfuscation techniques are implemented using single obfuscation cell structures throughout the design. This, in turn, aids the adversary to find-out the glue logic during image processing-based RE. Hence, it is proposed to use different obfuscation cell structures [4.8].

2. To avoid repeatability in netlist obfuscation, it is demonstrated to select obfuscation cells in random for each iteration of logic obfuscation process.

3. It is proposed to randomize the control value generation using true random number generator. So that, the output values in obfuscation mode will not be identical in each power on condition.

4. Insertion of obfuscation cells at HF nets may leave a hint to adversary about logic obfuscation. To avoid this scenario, the inputs of HF-driver modules are used instead of HF nets to insert obfuscation cells.

5. To validate the proposed claims, the complete automation and integration of the logic obfuscation approach with the regular FPGA design flow is performed using “perl” and “tcl” scripting languages.

6. To evaluate the improvement in RE complexity, the novel obfuscation metric is proposed to quantify the percentage of modification introduced by obfuscation techniques. The results are presented from validation of the modeled obfuscation tool.

4.2 Implementation of Structural Modification based Netlist Obfuscation Technique to PLDs

4.2.1 Methodology

The structural modification based netlist obfuscation technique is applied for PLD-based digital designs. This method aims to achieve a high percentage of simulation or structural mismatch during RE. A combination of lock and key structure accomplish the requirement. The obfuscation cell structures and FSM-based initialization keys act as the lock and key mechanisms as shown in Figure 4.1. There are two modes of operation in FSM such as obfuscated and functional modes. The control signal "En" derived from FSM is required to be "one" for the obfuscated mode and "zero" for the functional mode. It is stitched with selective internal nets (N_1, N_2,..N_n) using the obfuscation cell structures (M_1, M_2,..M_n) and the obfuscated nets are N^*_1, N^*_2,..N^*_n. The block diagram of a obfuscation cell structure "M" is shown in Figure 4.2.

The circuit stays in obfuscated mode upon global reset (i.e. initial state). From initial state, a set of inputs or initialization key sequence (e.g. P_0, P_1, P_2, P_5, P_7 in Figure 4.1) must be applied to drive the circuit to functional mode. This enabling key sequence acts as authentication sequence and allows the circuit to enter into functional mode; otherwise, circuit stays at obfuscated mode and does not perform the required functionality. Once user authentication is performed, the circuit remains in functional mode and ensures correct functionality.

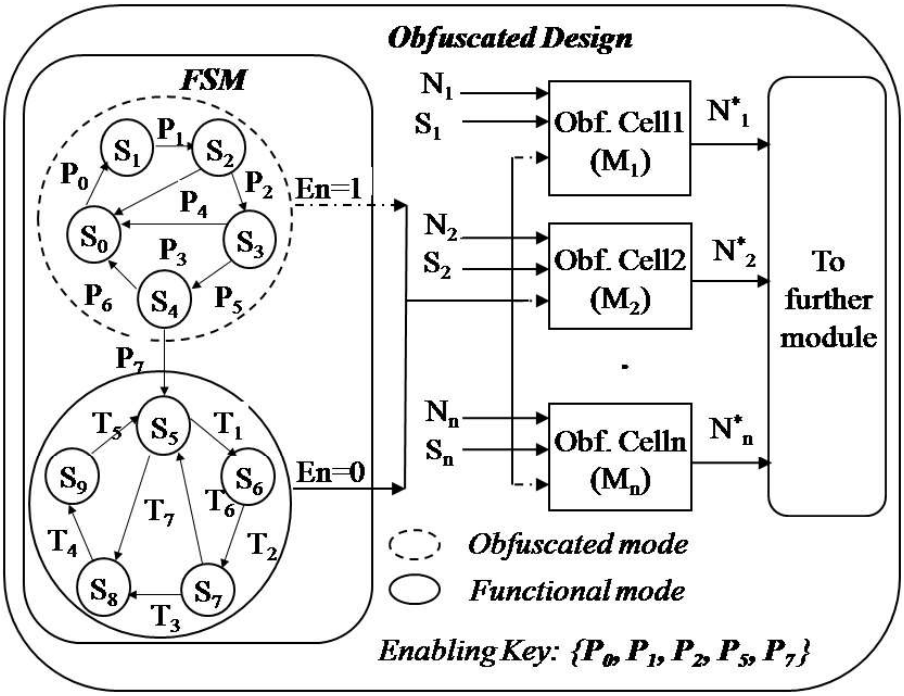


Figure 4.1: Structural modification based obfuscation technique

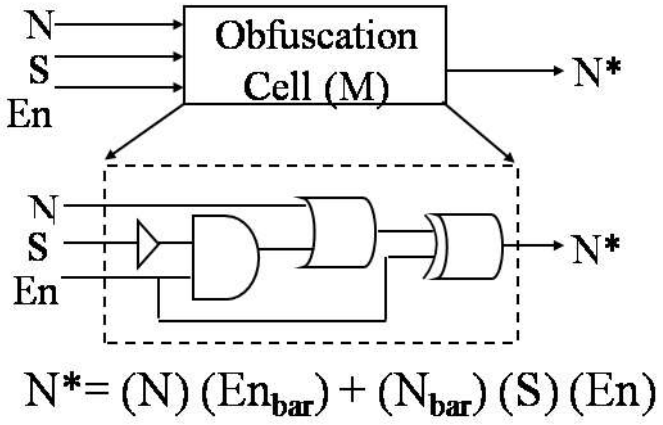


Figure 4.2: Block diagram with example of obfuscation cell structure

Most of the obfuscation techniques are implemented using single obfuscation cell structures throughout the design. This, in turn, aids the adversary to find-out the glue logic during image processing-based RE. Hence, it is proposed to use different obfuscation cell structures. As shown in Figure 4.2, the value of an obfuscated net (N^*) is a function of (N, En,

S). In the proposed method, the obfuscation logic is implemented such that N^* equal to N in functional mode (i.e. En='0') and N^* is not equal to N in obfuscated mode (i.e. En='1'). The total 2^3=8 input combinations generates 2^8=256 different output values. Among the total 256 possible output values, our requirement is such a way that for half of the combinations (2^4=16), the output value is same as input, i.e. when En='0'. Therefore, we finally have 2^4=16 different OC structures. Also, to avoid repeatability in the netlist obfuscation, it is proposed to select obfuscation cells in random for each iteration of logic obfuscation process. That is, the obfuscation tool uses "the random function" to select the appropriate OC structure from the set of 16 different OCs. This, in turn, avoids the repeatability in logic obfuscation, i.e. it generates different obfuscated designs at different time intervals. Therefore, no two obfuscated designs of the same target circuit are identical. In addition, it is proposed to randomize the control value "En" using true random number generator module in the device. So that, when an adversary performs RE, the output values of the obfuscated design will not be the same at each power on conditions.

Insertion of obfuscation cells at HF nets may leave a hint to adversary about logic obfuscation. To avoid this scenario, the inputs of HF-driver modules (N_1, N_2…N_n in Figure 4.1) are used instead of HF nets to insert obfuscation cells. Usually, the net with larger fan-out logic cone (i.e. the set of logic bounded by registers, inputs/ outputs, or black boxes) will affect the input logic of a comparatively larger number of nets which, in turn, controls more outputs. In case, the HF-driver module is either AND or OR gate with one of the inputs as logic "1" or logic "0", respectively then the effect of obfuscation is bypassed as shown in Figure 4.3. Therefore, it is required to perform the transition probability (TP) calculation of "one" and "zero" of AND or OR based HF-driver modules.

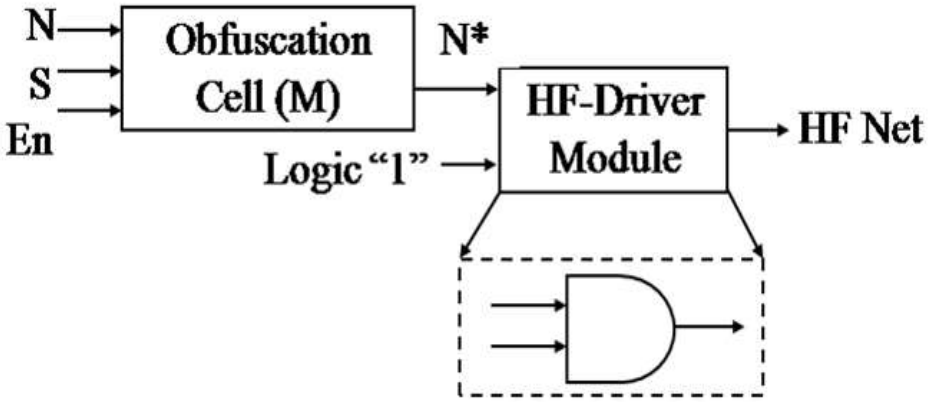


Figure 4.3: Effect of AND gate based HF-driver module on logic obfuscation

TP calculation of a circuit shown in Figure 4.4 is explained as follows [4.9]. At the input stage, it is possible to give either a zero or a one; hence, the TP of either of the inputs are 0.5 for all the gates in very first level of the circuit (i.e. G_1, G_2, G_3 and G_4). TP (P_{trans}) for both the inputs are P(1)×P(0) = 0.5×0.5 = 0.25. From the truth table of an AND gate, it is observed that "0" is obtained as an output for three different input combinations and "1" is the output just for one combination out of four possible combinations. Hence, the TP of obtaining "0" is 0.75 and that of obtaining "1" is 0.25. TP at the output of the AND gate is P(0) × P(1) = 0.75×0.25 =0.1875. Following this procedure, the TP of the primary inputs and the very first level of the circuit (G_1, G_2, G_3 and G_4) are calculated. If input probabilities of a gate are not equal such as G_5, G_6 and G_7, it is required to follow the individual probability method as explained below. In case of an AND gate, there is only one combination (1,1) which gives an output of "1". Hence, this probability is a product of all the probabilities when input is "1", P(out=1) = P(in_1=1) × P(in_2=1), where in_1 and in_2 are the two inputs. If there are more than two inputs to the AND gate then P(out=1) =P(in_1=1) × P(in_2=1)×...×P(in_n=1). Hence, TP will be P_{trans} = P(out=1) × P(out=0) =P(out=1) × [1 – P(out=1)]. For an OR gate, P(out = 0) = P(in_1= 0) × P(in_2= 0). In case of multiple inputs for

an OR gate P(out=0) =P(in_1=0) × P(in_2=0) ×...×P(in_n=0). Therefore, TP is P_{trans} = P(out=1) ×P(out=0) = [1 – P(out=0)] × P(out=0). Once TP calculations are performed, a net with high TP for "1" for AND gate and a net with high TP for "0" for OR gate is selected to stitch the OC structures. Therefore, the effect of logic obfuscation will not be completely bypassed.

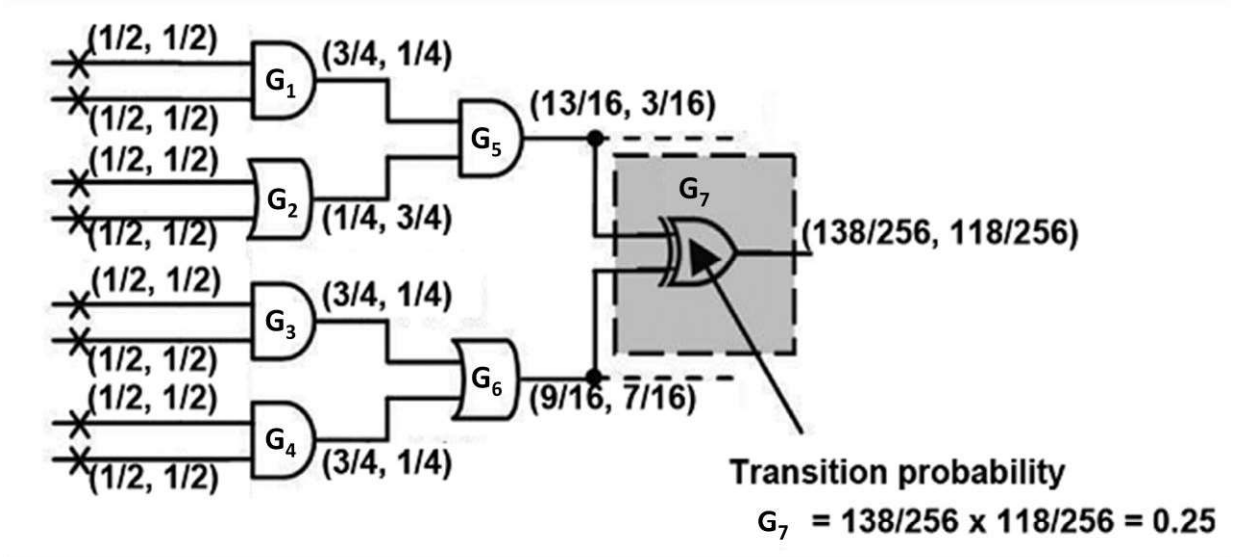


Figure 4.4: Transition probability of a circuit [4.9]

Also, large input logic cone of a net is indicative of its higher logic depth. Any change in such a net is likely to alter a large number of primary outputs. Therefore, specific internal signals S_1, S_2…S_n is included as one of the inputs to obfuscation cell structures to increase its input logic cone. The conditions to select internal nets "S" are as follows [4.1].

1. It should have a very large fan-in cone, which would substantially expand the logic cone of the obfuscated net,

2. It should not be in the fan-out cone of the obfuscated net and

3. It should not have any net in its fan-in cone that is in the fan-out cone of the obfuscated net.

The conditions (2) and (3) are essential to prevent any combinational loop in the obfuscated netlist. To increase simulation or structural RE complexity, either total number of obfuscation cells or initialization sequence length can be modified based on the user constraint. To study the effect of both parameters on RE complexity, two scenarios of obfuscation are performed such as:

1. For better structural mismatch during RE, obfuscation cells are inserted at different numbers of HF-driver nets with fixed initialization sequence length. As per the designer‘s area constraint, the total number of nets to be obfuscated is chosen.

2. For better functional simulation mismatch during RE, FSM with a different number of obfuscated states and initialization sequence is included in the design with fixed number of obfuscation cells. The initialization sequence length “L” is decided with respect to the system clock cycle (i.e. delay constraint).

The design flow of PLDs is shown in Figure 4.5 as the design phase. The obfuscation phase shows the complete flow diagram of structural modification based netlist obfuscation technique. The post-synthesis HDL netlist (.v/ .vhd file) generated through back annotation is chosen as the input file for obfuscation process. Using the detailed fitting or PAR report generated by EDA tools, the list of HF nets are extracted and they are used to find out the HF driver nets. To avoid additional delay introduced by obfuscation, HF-driver modules which fall in the critical path are not selected for obfuscation process. This, in turn, helped to get positive slack in the design. As per the designer constraints on the total number of obfuscation cells

(scenario 1) and obfuscated states (scenario 2), the required number of obfuscated nets and L values are chosen. Finally, different obfuscation cell structures are stitched at selected HF-driver nets "N" using the control signal "En" and internal net "S". The obfuscated netlist is re-synthesized and undergoes remaining back-end processes so that both original and obfuscation logics are merged to increase structural RE complexity. Finally, the entire process of structural modification based netlist obfuscation approach is automated and integrated with the EDA tool using "perl" and "tcl" scripting languages.

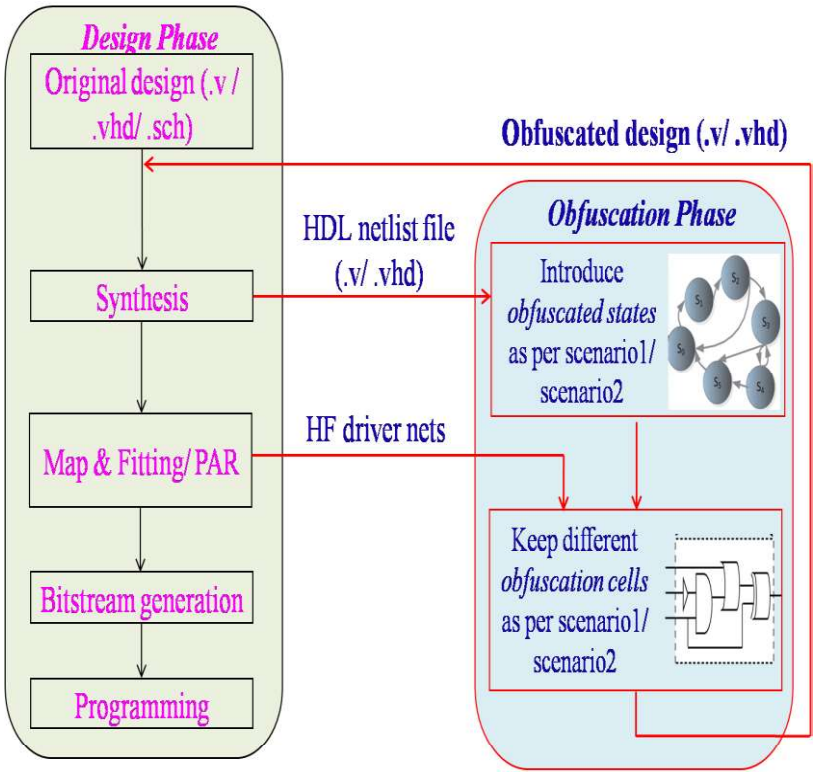


Figure 4.5: Flow diagram of structural modification based netlist obfuscation

4.2.2 Results and Discussion

In order to verify the effectiveness of logic obfuscation technique on PLD-based digital designs, the lock and key-based obfuscation technique is applied on various ISCAS'89

benchmark circuits. While structural modification based netlist obfuscation is applicable to both CPLDs and FPGAs, in this section, the simulation result of FPGA is presented. Implementation of this method is performed on the FPGA development board using Libero SoC v10.1 as shown in Figure 4.6. The Libero SoC v10.1 has multi EDA tools support for various back-end processes such as SynplifyPro for synthesis, Modelsim for simulation, Designer for PAR and FlashPro for device programming.

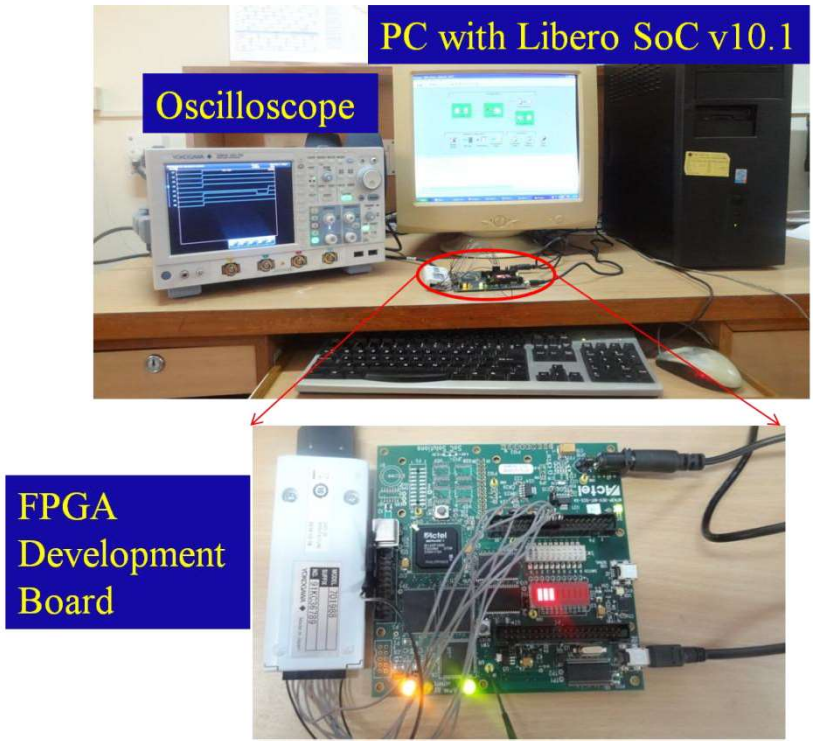


Figure 4.6: Implementation of obfuscated design using FPGA development board

The verilog codes of ISCAS benchmark circuits are converted to post-synthesis HDL netlist such as VHDL or verilog files using SynplifyPro as described in the flow diagram in Figure 4.5. The true random number generator is available as hard IP in Actel devices; hence, implementation of this module would not consume additional core cells. The implementation details of the benchmark circuits with respect to I/O and core cells are listed in Table 4.1. The

structural modification based netlist obfuscation method consists of two major implementations such as (1) FSM based key mechanism and (2) Obfuscation cell based lock mechanism. In general, the key mechanism is implemented as separate FSM logic with varying obfuscation states, i.e. varying L values. The lock mechanism is implemented at the inputs of HF-driver modules of target circuit using different obfuscation cell structures. To achieve better structural RE complexity, the nets that are directly connected to I/Os are not considered for circuit obfuscation as they are easily decodable nets during RE.

Table 4.1: Resource utilization of ISCAS'89 circuits

Circuits	**Resource Utilization (cells)**	
	I/O	**Core**
s382	11	90
s444	11	94
s510	28	146
s526	11	101
s641	60	102
s820	39	165
s832	39	173
s838	37	177
s1196	30	348
s1238	30	347
s1423	24	519
s1488	29	384
s5378	86	823
s9234	69	567

Two scenarios of logic obfuscation are performed such as (1) For better structural mismatch during RE, insertion of obfuscation cells at different numbers of HF-driver nets with fixed L value. As per the designer's area constraint, the total number of nets to be obfuscated is chosen. For example, the obfuscation cells are inserted for 10%, 15%, 20% and 25% area

constraints with L equal to 2. (2) For better functional simulation mismatch during RE, FSM with different L values is included in the design with a fixed number of obfuscation cells. The initialization sequence length is decided with respect to the system clock cycle (i.e. delay constraint). For example, FSM with L varying from 2 to 8 with 10% area constraint is included. However, as the obfuscated net percentage and L value increase, the resource utilization and system clock cycle increase, respectively. Based on the designer‘s requirement either on simulation or structural RE complexity, obfuscation methods described in scenario 1 or scenario 2 can be applied. The generalized procedure of the structural modification based netlist obfuscation approach is completely automated using the “perl” scripting language. Later, the obfuscation process is integrated with Libero SoC v10.1 using “tcl” scripting language i.e. to perform the pre-obfuscation and the post-obfuscation steps. Finally, batch file is created to call all three scripts in order, i.e. perl script for obfuscation, tcl scripts for pre and post obfuscation processes.

As logic obfuscation is a key technique incorporated for the hardware security using additional logics, the percentage modification introduced in following three important design parameters are measured:

1. Area- total number of technology mapped resources utilized to implement obfuscation, measured in terms of core and I/O cells. As the system I/Os are not modified during obfuscation, the percentage modifications introduced only in core cells are considered,

2. Delay- critical path delay of the circuit, measured in nanoseconds and

3. Power- total power consumption of logic and it is a sum of static and dynamic power consumptions, measured in milliwatts.

The area, delay and power overheads of the re-synthesized benchmark circuits, following the application of the proposed obfuscation scheme for scenario 1 and scenario 2 are listed in Table 4.2 and Table 4.3.

Table 4.2: Overhead calculation with varying area constraints (Scenario 1)

Ckts	**10%**			**15%**			**20%**			**25%**		
	Area %	***Delay %***	***Power %***	***Area %***	***Delay %***	***Power %***	***Area %***	***Delay %***	***Power %***	***Area %***	***Delay %***	***Power %***
s382	10.00	-0.45	6.92	14.44	0.45	10.8	20.00	0.85	12.98	24.44	0.05	15.05
s444	9.57	0.09	5.48	14.89	-1.26	8.97	19.15	0.57	12.82	24.47	-0.67	16.36
s510	9.59	-0.38	6.08	14.38	-0.14	12.37	19.86	0.17	14.85	25.34	-0.05	17.19
s526	9.90	0.1	8.23	14.85	-0.68	11.99	19.80	0.18	14.54	24.75	0.27	16.95
s641	9.80	-0.02	7.24	14.71	0.43	11.77	19.61	-0.9	15.12	24.51	-1.32	18.22
s820	9.70	0.14	5.37	13.94	-0.6	11.67	20.00	-0.66	15.44	24.85	0.04	18.9
s832	9.83	0.52	7.67	14.45	0.37	10.54	20.23	-1.36	15	25.43	-1.06	19.03
s838	9.60	0.65	8.59	14.12	-0.07	11.71	19.77	0.04	17.74	24.86	-0.9	23.01
s1196	9.84	0.17	6.68	14.84	-1.25	13.74	20.00	-0.89	18.24	24.84	0.12	22.3
s1238	9.91	-0.21	6.12	14.96	0.27	13.87	19.24	-1.02	19.18	24.99	0.54	23.88
s1423	9.85	-0.58	7.57	14.24	-0.21	10.71	19.87	0.8	16.73	24.70	0.45	21.99
s1488	9.72	0.24	8.18	14.75	-0.14	13.8	19.72	-0.41	17.25	24.93	-0.87	20.44
s5378	9.90	0.07	6.43	14.82	0.87	14.57	19.99	-1.02	20.19	24.90	-1.54	25.12
s9234	9.99	-0.15	10.03	14.92	0.17	15.7	19.63	0.27	20.2	24.79	-0.08	24.24

Table 4.3: Overhead calculation with varying initialization sequence length (Scenario 2)

Ckts	L=2			L=4			L=6			L=8		
	Area %	*Delay %*	*Power %*	*Area %*	*Delay %*	*Power %*	*Area %*	*Delay %*	*Power %*	*Area %*	*Delay %*	*Power %*
s382	10.00	-0.45	6.92	18.18	-0.52	9.69	25.00	-0.72	13.63	28.57	-0.27	17.24
s444	9.57	0.09	5.48	20.34	-0.34	16.69	27.13	-0.53	20.57	31.88	0.13	24.10
s510	9.59	-0.38	6.08	16.09	-0.90	20.65	21.08	-0.60	25.55	27.72	-0.67	29.89
s526	9.90	0.1	8.23	19.84	0.27	16.61	26.28	-0.42	20.83	30.82	0.04	24.64
s641	9.80	-0.02	7.24	19.05	0.48	18.94	25.55	0.18	24.02	35.03	0.23	28.50
s820	9.70	0.14	5.37	15.38	-1.20	17.89	19.90	-1.17	22.73	25.34	-1.20	27.04
s832	9.83	0.52	7.67	15.20	-0.87	19.61	19.53	-0.97	24.52	24.78	-0.87	28.87
s838	9.60	0.65	8.59	14.90	0.01	14.16	19.18	-0.32	18.82	24.68	0.20	23.00
s1196	9.84	0.17	6.68	12.93	-0.72	15.68	16.09	0.13	18.99	19.25	0.41	22.05
s1238	9.91	-0.21	6.12	12.97	0.08	13.42	16.14	0.14	16.91	19.31	-0.48	20.13
s1423	9.85	-0.58	7.57	11.95	0.42	11.59	14.07	0.33	14.11	16.18	-0.64	16.49
s1488	9.72	0.24	8.18	12.50	-0.11	14.34	15.36	-0.41	17.50	18.23	-0.47	20.43
s5378	9.90	0.07	6.43	11.18	-2.18	11.54	12.52	-1.01	13.68	13.85	-1.29	15.73
s9234	9.99	-0.15	10.03	11.99	-0.60	13.62	13.93	-0.80	16.08	15.87	0.22	18.40

To study the effect of total number of obfuscation cells and “L” values on the delay and power overheads, the data are plotted as shown in Figure 4.7 and Figure 4.8. From results, it is clearly evident that the overhead calculations i.e. the area and power measurements are smaller than the imposed constraints in most of the scenarios, while the timing overhead was negative, i.e., the timing constraint was met with positive slack in most cases. The design overhead is ultimately caused both by the addition of combinational (in the form of the obfuscation cells) and few sequential elements to implement the inserted FSM. Here, the overhead is gradually increasing as the percentage of obfuscated nets and L values are increased. As future direction of this research, performing experiments that demonstrate the effectiveness of the proposed

technique to advanced attacks [4.10-4.13] such as reverse engineering finite state machine (REFSM), fault injection attacks and SAT-based attacks on logic obfuscation is suggested.

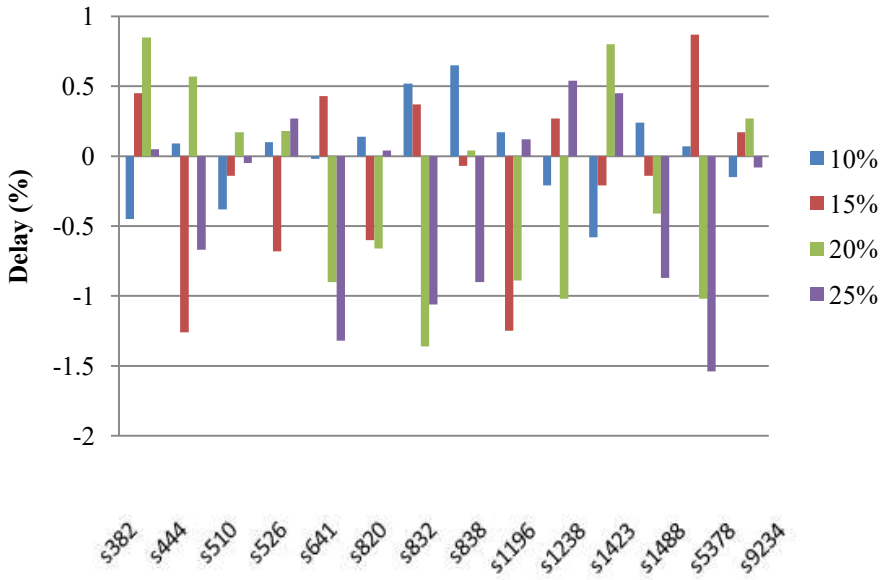


Figure 4.7: Delay overhead with varying area constraints (Scenario 1)

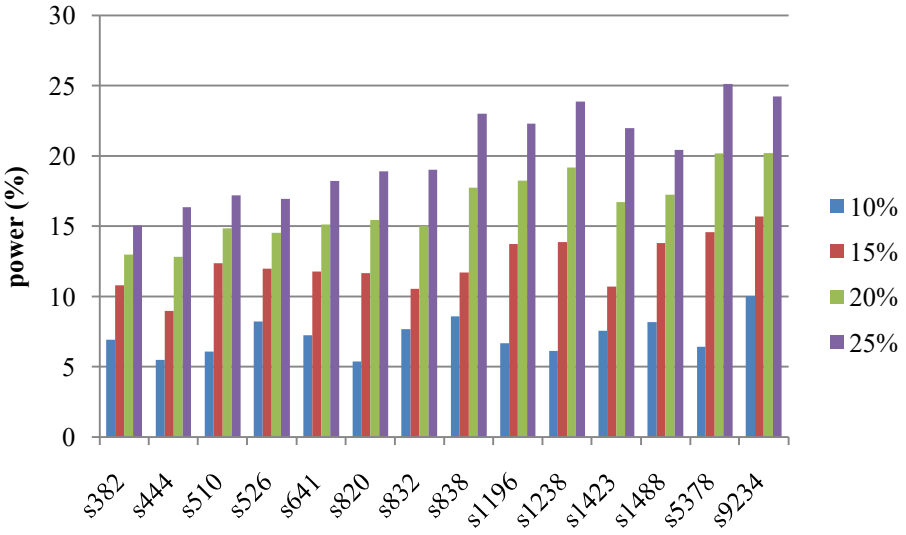


Figure 4.8: Power overhead with varying area constraints (Scenario 1)

4.3 Obfuscation Metric Calculation

In addition to the implementation of logic obfuscation techniques for circuit anti-tampering, it is also required to quantify the percentage of obfuscation or circuit hiding achieved. Ultimately, the strength of obfuscation defines the difficulty for an attacker to find out the correct functionality. In fact, hardware obfuscation and RE process are complementing to each other. That is, higher the obfuscation efficiency higher the RE complexity. The following four different obfuscation metrics are recommended such as:

1. Code modification metric (% M_{cm}),

2. Area modification metric (% M_{am}),

3. Simulation metric (% M_{sim}) and

4. Failing verification point metric (% M_{fvp}).

4.3.1 Code Modification Metric (% M_{cm})

To measure the difficulty of manual attack by visual inspection, percentage of modification introduced in the original code called as "code modification metric (% M_{cm})" is introduced. As the original and obfuscated codes are in the netlist format whose syntax consists of component/ signal declaration and port mapping statements, total modification in these parameters are measured to calculate the code modification metric. Let N_c is total number of component declarations; N_s is total number of signal declarations and N_{pm} is total number of port mapping statements. Hence, the code modification metric (% M_{cm}) is given in Eqn. 4.1.

$$\% M_{cm} = \frac{((N_{cspm})_{obf} - (N_{cspm})_{org})}{(N_{cspm})_{obf}} \text{x}100 \quad (4.1)$$

where N_{cspm} is addition of N_c, N_s and N_{pm}. Here, 'org'and 'obf' denotes the original and obfuscated codes.

4.3.2 Area Modification Metric (% M_{am})

It is well understood that higher the design complexity higher the RE difficulty. Hence, to highlight the RE difficulty with respect to the complexity of obfuscated design, percentage of area modification, also called as "area modification metric (% M_{am})" shown in Eqn. 4.2 is chosen as one of the obfuscation parameters.

$$\% M_{am} = \frac{((Area)_{obf} - (Area)_{org})}{(Area)_{obf}} \text{ x100} \quad (4.2)$$

where, $(Area)_{org}$ and $(Area)_{obf}$ denote total core cell utilization of the original and obfuscated codes, respectively.

4.3.3 Simulation Metric (% M_{sim})

The functional simulation is one of the major brute force attacks to get in to functional mode. Therefore, the difficulty to apply correct initialization sequence is measured as "simulation metric (% M_{sim})." Let "K" is total number of inputs of original design, then probability (P_1) to apply correct input to enter into functional mode is shown in Eqn. 4.3. Similarly, let "K" is total number of inputs of obfuscated design and "L" is initialization sequence length, then probability (P_2) to apply correct initialization sequence to enter into functional mode is shown in Eqn. 4.4. The simulation metric (% M_{sim}), i.e. the difficulty in applying correct initialization sequence is calculated using P_1 and P_2 as shown in Eqn. 4.5.

$$P_1 = \frac{1}{2^K} \quad (4.3)$$

$$P_2 = \frac{1}{2^{K.L}} \tag{4.4}$$

$$\% M_{sim} = \frac{P_1 - P_2}{P_1} \text{ x100} \tag{4.5}$$

4.3.4 Failing Verification Point Metric (% M_{fvp})

As simulation and structural analysis are the conventional methods to perform RE, it is clearly evident that higher the structural mismatch, higher the code is obfuscated from its original structure. This, in turn, increases the RE complexity. The structural mismatch between the original and obfuscated design is measured using the formal verification tools also called as "failing verification point metric % M_{fvp}" and it is included in the obfuscation metric calculation. Applying the original and obfuscated designs to the formal verification tool, it reports the total number of comparison points (T_c) and number of removed comparison points (R_c). The ratio between R_c and T_c gives the failing verification point metric % M_{fvp} as shown in Eqn. 4.6.

$$\% M_{fvp} = \frac{R_c}{T_c} \text{x100} \tag{4.6}$$

4.3.5 Obfuscation Metric (% M_{obf})

Using the four obfuscation parameters such as % M_{cm}, % M_{am}, % M_{fvp} and % M_{sim}, the total percentage of the RE complexity or obfuscation achieved can be deduced as "obfuscation metric % M_{obf}" parameter.

$$\% M_{obf} = (w_1 . M_{cm} + w_2 . M_{am} + w_3 . M_{sim} + w_4 . M_{fvp}) \text{ x } S \tag{4.7}$$

where w_1, w_2, w_3 and w_4 are weights given to each parameter and it is assumed that $w_1 = w_2 = w_3 = w_4 = 0.25$ due to the important contribution of each parameters to measure the RE complexity. Here, "S" is the size parameter which is used to incorporate the effect of target design size on

% M_{obf} calculation. It is the ratio between target design size to target device size. The simplified flow diagram of obfuscation metric calculation i.e. to measure % M_{cm}, % M_{am}, % M_{fvp} and % M_{sim} parameters and to calculate % M_{obf} from the regular PLD design flow is depicted in Figure 4.9.

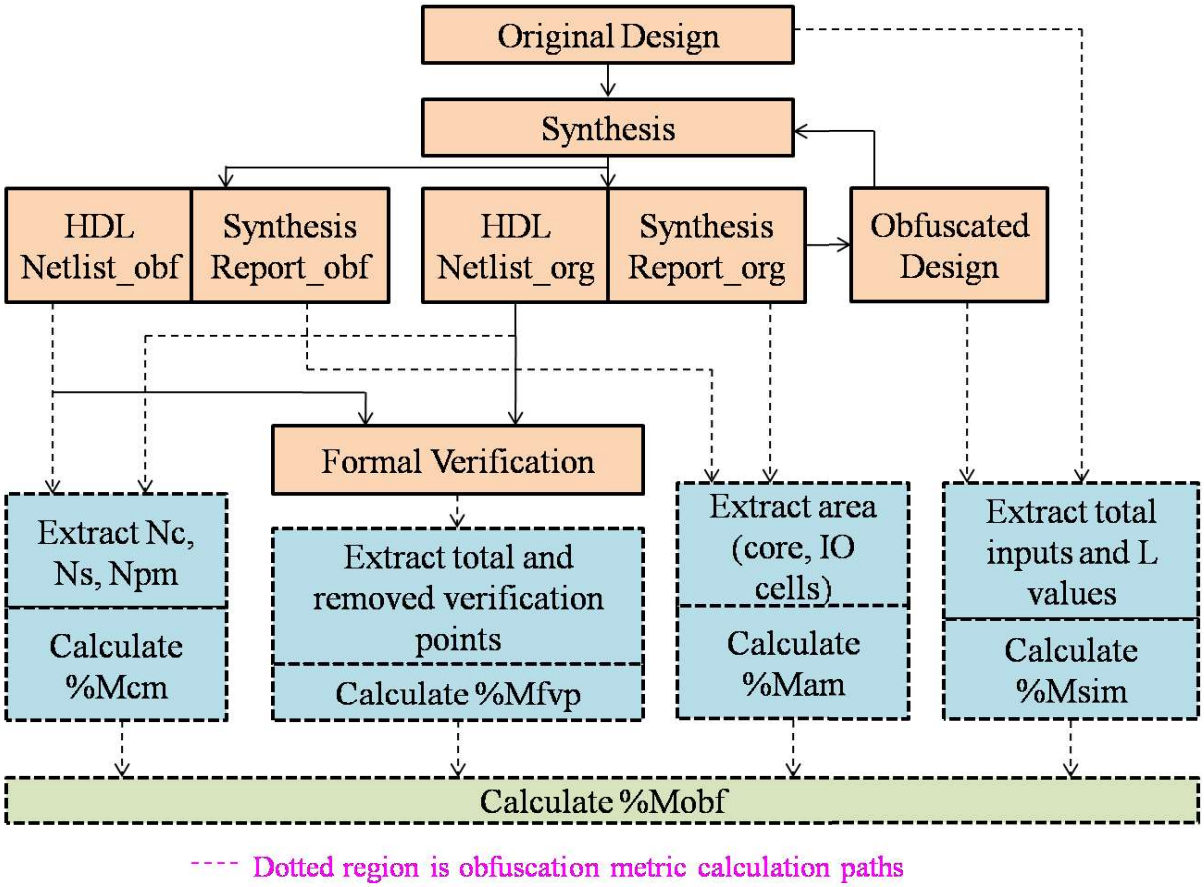


Figure 4.9: Flow diagram of obfuscation metric calculation

4.3.6 Results and Discussion

The obfuscation metric calculation to quantify the percentage of obfuscation or RE complexity achieved is performed using the following obfuscation parameters such as code modification metric (% M_{cm}), area modification metric (% M_{am}), simulation metric (% M_{sim}) and

failing verification point metric (% M_{fvp}). The results are listed in Table 4.4 and Table 4.5 and shown in Figure 4.10 and Figure 4.11, respectively, for varying area constraints and initialization sequence length. From results, it is well understood that the obfuscation metric is gradually increasing with increase in obfuscated nets and L values. The result also shows that for small target circuits such as s382 and s444, the % M_{obf} value is less i.e. reverse engineering of smaller circuit is comparatively easier than larger target circuits. However, these values shall be balanced with reference to the permitted design overhead limit i.e. area and clock constraints; thus, the implementations must realize a trade-off between them.

Table 4.4: % M_{obf} calculations with varying area constraints (Scenario 1)

Ckt	% Msim	10%				15%				20%				25%			
		% Mcm	% Mam	% Mfvp	% Mobf	% Mcm	% Mam	% Mfvp	% Mobf	% Mcm	% Mam	% Mfvp	% Mobf	% Mcm	% Mam	% Mfvp	% Mobf
s382	3.12	0.93	0.33	0.30	4.68	1.40	0.47	0.31	5.3	1.96	0.65	0.55	6.28	2.75	0.8	0.59	7.26
s444	3.25	1.24	0.32	0.37	5.18	1.86	0.5	0.54	6.15	2.60	0.72	0.71	7.28	3.65	0.89	0.88	8.67
s510	5.21	4.09	0.5	0.50	10.3	6.14	0.76	0.87	12.98	8.59	1.04	1.29	16.13	12.02	1.33	1.73	20.29
s526	3.50	2.54	0.36	0.53	6.93	3.81	0.54	0.56	8.41	5.33	0.72	1.16	10.71	7.47	0.9	1.45	13.32
s641	3.65	2.61	0.4	0.35	7.01	3.92	0.6	0.56	8.73	5.48	0.86	0.78	10.77	7.67	1.02	1.06	13.4
s820	5.89	4.97	0.58	0.41	11.85	7.46	0.83	0.78	14.96	10.44	1.19	1.07	18.59	14.61	1.48	1.26	23.24
s832	6.18	5.78	0.61	0.99	13.56	8.67	0.9	1.69	17.44	12.14	1.26	2.12	21.7	16.99	1.59	2.55	27.31
s838	6.32	5.92	0.68	0.61	13.53	8.88	1.01	1.08	17.29	12.43	1.39	1.60	21.74	17.40	2.02	2.05	27.79
s1196	12.43	11.89	1.24	1.72	27.28	13.08	1.86	2.91	30.28	14.39	2.51	3.81	33.14	15.83	3.12	4.66	36.04
s1238	12.40	11.93	1.29	1.33	26.95	13.12	1.89	2.25	29.66	14.44	2.41	3.26	32.51	15.88	3.08	4.16	35.52
s1423	18.55	19.87	1.85	3.91	44.18	21.86	2.67	7.62	50.7	24.04	3.72	9.39	55.7	26.45	4.63	10.71	60.34
s1488	13.69	12.07	1.35	1.59	28.7	13.28	2.04	2.90	31.91	14.60	2.73	3.77	34.79	16.07	3.45	4.66	37.87
s9234	29.43	25.87	2.94	5.54	63.78	28.46	4.41	10.30	72.6	31.30	5.94	12.98	79.65	34.43	7.4	15.88	87.14
s5378	20.31	20.94	2.05	3.96	47.26	23.03	3.06	7.95	54.35	25.34	4.03	10.36	60.04	27.87	5.08	11.86	65.12

Table 4.5: % M_{obf} calculations with varying initialization sequence length (Scenario 2)

Ckt	% Mcm	% Mfvp	L=2			L=4			L=6			L=8		
			% Mam	% Msim	% Mobf	% Mam	% Msim	% Mobf	% Mam	% Msim	% Mobf	% Mam	% Msim	% Mobf
s382	0.93	0.30	0.33	3.12	4.68	0.62	3.58	5.43	1.18	3.91	6.31	2.23	4.10	7.56
s444	1.24	0.37	0.32	3.25	5.18	0.60	3.84	6.05	1.17	4.20	6.98	2.22	4.49	8.32
s510	4.09	0.50	0.5	5.21	10.3	0.96	5.66	11.21	1.82	6.02	12.43	3.46	6.58	14.62
s526	2.54	0.53	0.36	3.50	6.93	0.69	4.10	7.86	1.30	4.46	8.84	2.48	4.75	10.30
s641	2.61	0.35	0.4	3.65	7.01	0.71	4.10	7.77	1.34	4.46	8.76	2.56	5.11	10.63
s820	4.97	0.41	0.58	5.89	11.85	1.10	6.35	12.82	2.08	6.71	14.16	3.96	7.19	16.53
s832	5.78	0.99	0.61	6.18	13.56	1.17	6.64	14.58	2.22	7.00	15.98	4.21	7.49	18.47
s838	5.92	0.61	0.68	6.32	13.53	1.16	6.77	14.47	2.27	7.13	15.93	4.38	7.65	18.56
s1196	11.89	1.72	1.24	12.43	27.28	2.35	13.15	29.11	4.46	13.67	31.74	8.48	14.26	36.34
s1238	11.93	1.33	1.29	12.40	26.95	2.71	13.12	29.09	4.93	13.64	31.83	9.10	14.23	36.59
s1423	19.87	3.91	1.85	18.55	44.18	3.51	19.27	46.56	6.67	19.76	50.21	12.67	20.31	56.76
s1488	12.07	1.59	1.35	13.69	28.7	2.56	14.42	30.64	4.86	14.91	33.43	9.23	15.49	38.39
s9234	20.94	3.96	2.05	20.31	47.26	3.89	21.26	50.05	7.39	21.78	54.07	14.05	22.36	61.31
s5378	25.87	5.54	2.94	29.43	63.78	5.59	30.50	67.51	10.63	31.02	73.06	20.19	31.54	83.15

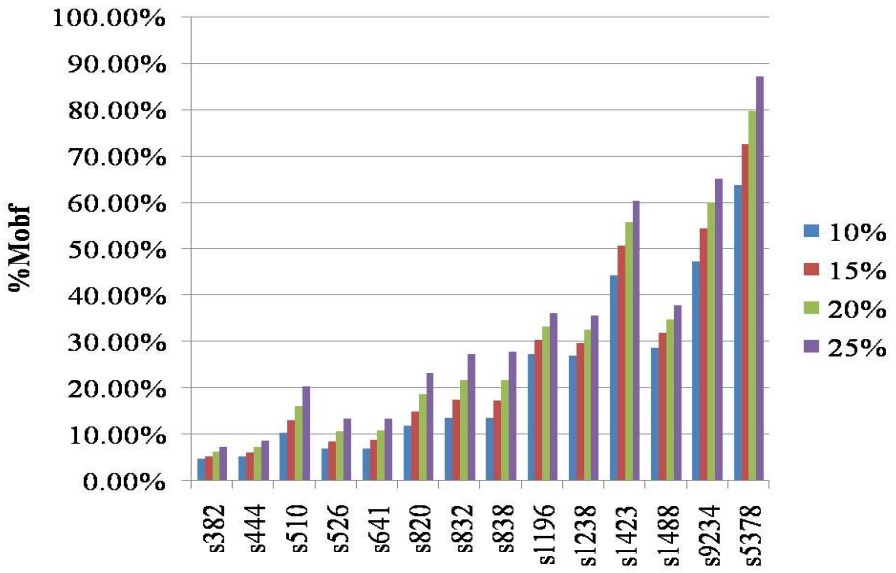


Figure 4.10: % M_{obf} calculation with varying area constraints

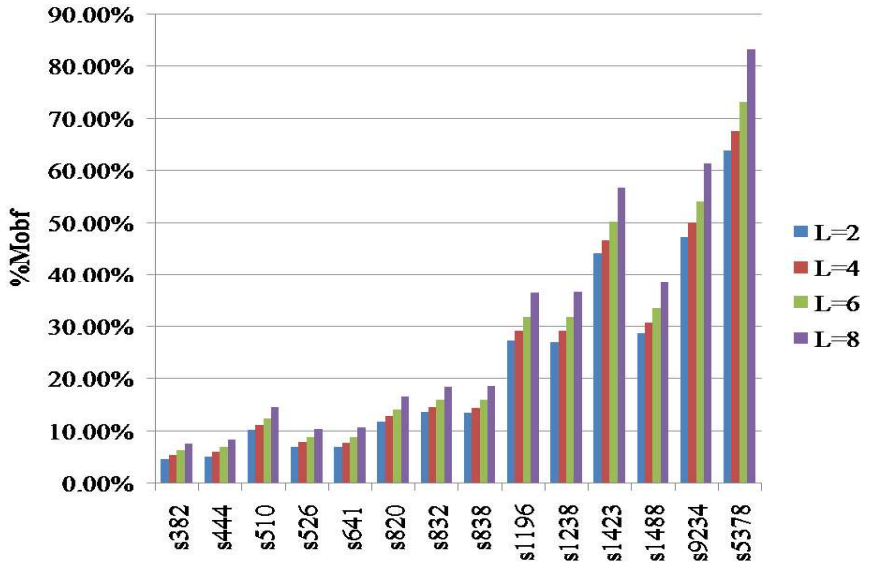


Figure 4.11: % M_{obf} calculation with varying initialization sequence lengths (L)

4.4 Automation and Integration of Obfuscation Methods

The generalized procedure of the structural modification based netlist obfuscation approach is completely automated using the "perl" scripting language. Later, the obfuscation process is integrated with Libero SoC v10.1 using "tcl" scripting language i.e. to perform the pre-obfuscation and the post-obfuscation steps. Finally, batch file is created to call all three scripts in order, i.e. perl script for obfuscation and tcl scripts for pre and post obfuscation processes as shown in Figure 4.12. The screenshot of the batch file running through command prompt to perform logic obfuscation and obfuscation metric calculation in automation is shown in Figure 4.13 for "s641" circuit.

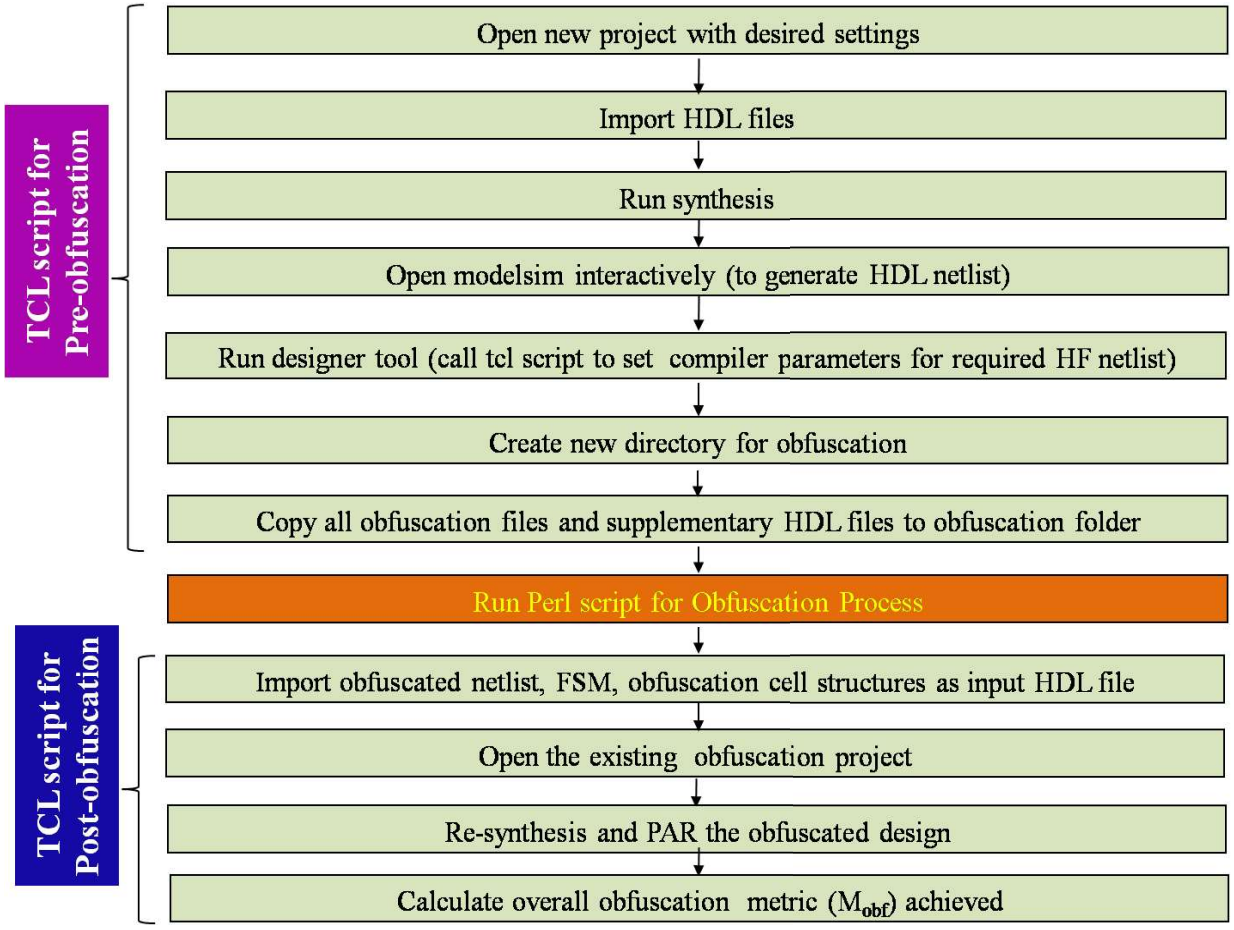


Figure 4.12: Steps in automation and integration of obfuscation methods

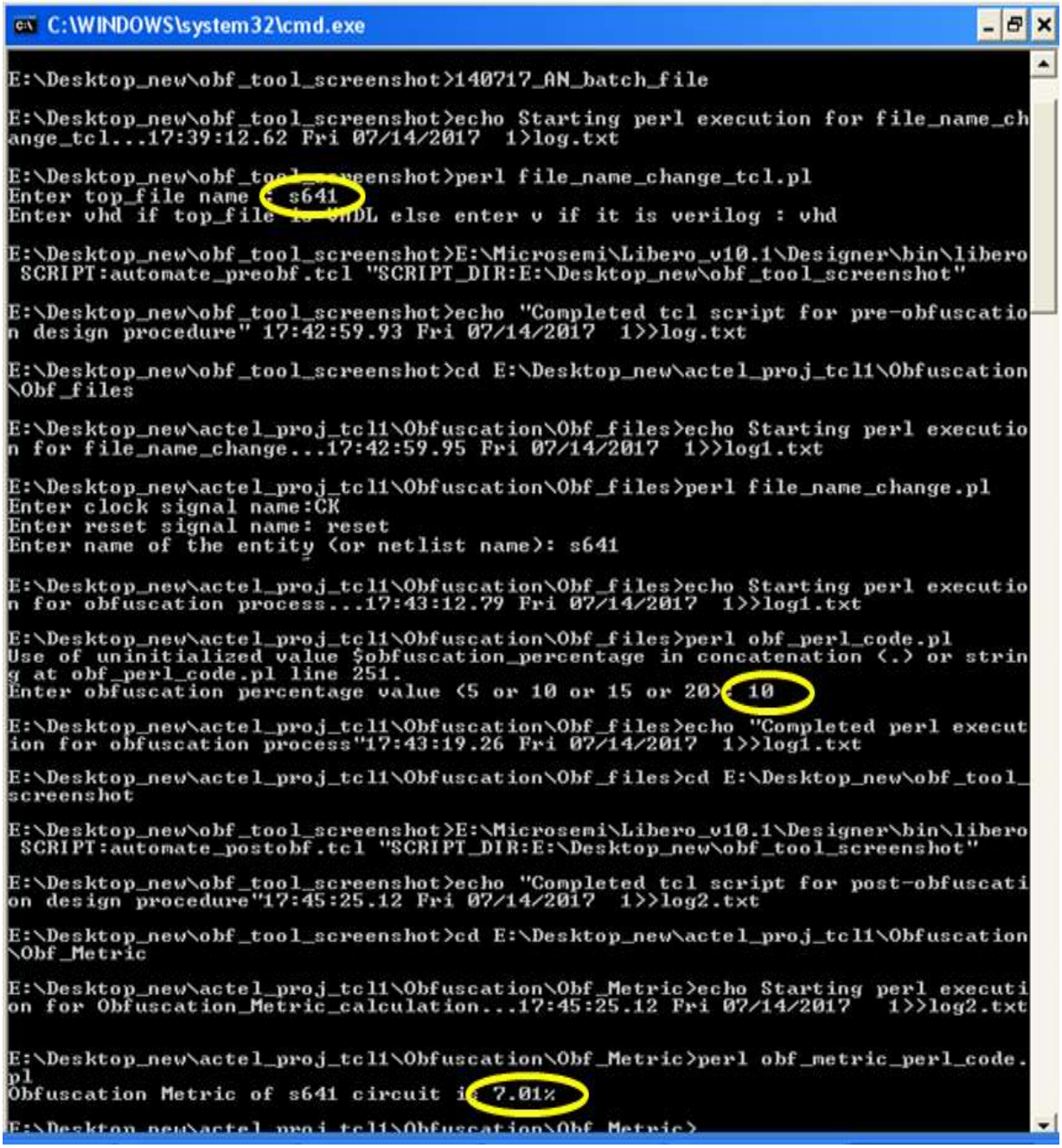


Figure 4.13: Obfuscation tool automation by command window

4.5 Summary

Reverse engineering of any safety critical designs may cause leakage of critical design parameters or encrypted keys and insertion of hardware Trojans to deny or destroy the critical

systems. To ensure the design and data security of such systems, logic obfuscation techniques are widely adopted. The structural modification based netlist obfuscation technique to PLD based digital designs is implemented. As per the need of simulation or structural RE complexity as well as the area and delay constraints, obfuscation with varying obfuscated nets and initialization sequence length is proposed. Results reveal that the proposed approach appears to be a quite useful technique for PLD designs and design overhead is well below the design constraints. In addition, the obfuscation metric is introduced based on the percentage of modifications introduced in code or structure, resource utilization, failing verification points and initialization sequence detection probability. This attempt to develop the obfuscation metrics will be highly useful to the designers to quantify the overall percentage of modification introduced by obfuscation techniques. Finally, the entire process of structural modification based netlist obfuscation approach is automated and integrated with the EDA tool using "perl" and "tcl" scripting languages. As a future scope, hardware obfuscation techniques shall be applied on the target design that includes the PUF and error correction circuits to increase the complexity of reverse engineering.

References:

[4.1] R. S. Chakraborty and S. Bhunia, "HARPOON: An obfuscation-based SoC design methodology for hardware protection," IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., vol. 28, no. 10, pp. 1493-1502, 2009.

[4.2] J. A. Roy, F. Koushanfar, and I. L. Markov, "Ending piracy of integrated circuits," Computer, vol. 43, no. 10, pp. 30-38, 2010.

[4.3] A. Baumgarten, A. Tyagi, and J. Zambreno, "Preventing IC piracy using reconfigurable logic barriers," IEEE Des. Test Comput., vol. 27, no. 1, pp. 66-75, 2010.

[4.4] W. P. Griffin, A. Raghunathan, and K. Roy, “CLIP: Circuit level IC protection through direct injection of process variations,” IEEE Trans. Very Large Scale Integr. Syst., vol. 20, no. 5, pp. 791-803, 2012.

[4.5] J. Rajendran, et al. “Fault analysis-based logic encryption,” IEEE Trans. Comput., vol. 64, no. 2, pp. 410-424, 2015.

[4.6] J. Zhang, “A Practical Logic Obfuscation Technique for Hardware Security,” IEEE Trans. VLSI Syst., vol. 24, no. 3, pp. 1193-1197, 2015.

[4.7] B. Colombier, L. Bossuet, and D. Hély, “From secured logic to IP protection,” Microprocessors and Microsystems, In Press, Corrected Proof- Note to users, Available online 26th February, 2016.

[4.8] G. Sumathi, L. Srivani, D. Thirugnana Murthy, Anish Kumar, and K. Madhusoodanan, “Hardware Obfuscation using Different Obfuscation Cell Structures for PLDs,” Cryptology and Information Security Series, vol. 15, pp. 143-157, 2017.

[4.9] H. Salmani, M. Tehranipoor, and J. Plusquellic, “A novel technique for improving hardware Trojan detection and reducing Trojan activation time,” IEEE Trans. Very Large Scale Integr. Syst., Vol. 20, no. 1, pp. 112–25, Jan. 2011.

[4.10] M. El Massad, S. Garg, and M. V. Tripunitara, “Integrated circuit (ic) decamouflaging: Reverse engineering camouflaged ics within minutes.” in NDSS, 2015.

[4.11] P. Subramanyan, S. Ray, and S. Malik, “Evaluating the security of logic encryption algorithms,” in IEEE Int. Symp. Hardware Oriented Security and Trust (HOST), pp. 137-143, 2015.

[4.12] T. Meade, Y. Jin, M. Tehranipoor, and S. Zhang, “Gate-level netlist reverse engineering for trojan detection and hardware security,” in IEEE Int. Symp. Circuits and Systems (ISCAS), pp. 1334-1337, 2016.

[4.13] T. Meade, S. Zhang, and Y. Jin, “Netlist reverse engineering for highlevel functionality reconstruction,” in 21st Asia and South Pacific Design Automation Conf., pp. 655-660, 2016.

5

Multi-Corner Timing Analysis based Reliability Calculation of PUFs

The present chapter discusses about the multi-corner timing analysis based reliability calculation of PUFs.

5.1 Introduction

The usage of outsourced IP cores, EDA tools and offshore fabrication facilities are highly encouraged in the design of digital electronic systems. Due to high third-party involvement, the digital systems are highly vulnerable to various hardware security threats such as IP piracy, hardware Trojans and counterfeiting. The cryptography algorithms and logic obfuscation based authentication schemes may mitigate these security issues to some extent. However, the storage of encryption keys, decryption keys and authentication signatures in on-chip or external non-volatile memories is of more concern from physical attacks such as laser cutting, micro-probing and power analysis. One of the strongest solutions to solve this problem is a generation of unique device-dependent binary signature, which avoids the burden of external storage. In this angle, the research enabled designers to come with the idea of implementation of PUF circuits [5.1]. A PUF is a challenge-response module which generates unique, reliable and tamper-proof signatures for a given IC based on the process variations inherent in the chip manufacturing process, i.e. the output response is totally random and unpredictable. PUFs are an emerging

technology, which makes it possible to base secure identification, authentication or generation of keys on unclonable hardware tokens. Since PUFs can be used to generate cryptographic keys, they can also be used instead of complex and expensive secure storage systems for storage of private keys. Instead of having to keep a private key stored in external secure memory, it can be generated by a PUF on demand and discarded after use. Hence, the key only exists during the cryptographic operations, which restricts the attack surface for an adversary immensely, i.e. PUF avoids side-channel attacks. Though performing side-channel attacks on PUFs are considered to be practically challenging process, it is possible with additional time, money and sophisticated lab facilities, i.e. with the advent of technology and sophisticated laboratory, any adversary can very well accomplish these attacks.

This section concentrates more on reliability estimation of these PUF modules implemented in FPGA devices. Though similar kind of research works have already been published in recent years [5.2-5.6], they are all calculated during testing phase by admitting devices at different temperature and voltage scenarios. There were very few simulation-based works on reliability-estimation and high-reliability PUF designs have been previously proposed in the literature [5.7-5.10]. A systematic statistical approach is presented in [5.7] to quantitatively evaluate various types of MUX-based PUFs. A failure-based PUF that uses failures induced by controlling the duration of power gating is proposed in [5.8] and demonstrated through simulation using D flip-flop circuits. DFT based technique is discussed in [5.9] to quantify improvement in overall reliability and efficiency of SRAM based PUFs. Also, the proposed method is analyzed in greater detail using random walk models.

Most of EDA tools allow performing multi-corner timing analysis, which enables the designer to check whether their design meets its intended functionality and performance at

extreme operating conditions i.e. allowable temperature and voltage ranges mentioned in the specification sheet. Multi-corner timing analysis allows the designer to set appropriate temperature, voltage values and standard delay format file includes the appropriate delay values introduced at the modified operating condition. Instead of allowing the device to undergo variable temperature and voltage testing to calculate reliability percentage of PUF module, a method to simulate those operating conditions using multi -corner timing analysis is proposed. The hamming distance between PUF responses measured at the typical operating condition and elevated operating conditions are calculated. Using reliability calculation model proposed in [5.4], the reliability estimation is performed.

5.1.1 Physically Unclonable Function Circuits

In applications such as cryptography, device authentication and digital signature generation, the secret key is stored in non-volatile memory (e.g. fuses) which is highly vulnerable to physical attacks. For example, side-channel attacks such as simple/ dynamic power analysis are possible to extract the secret keys. Instead of storing the secret key in a memory, it can be intrinsically generated by the PUF [5.11]. A PUF is a challenge-response primitive which generates unique, reliable and tamper-proof signatures for a given IC based on the process variations inherent in the IC manufacturing process, i.e. the output response is totally random and unpredictable. Based on its randomness, it generates unique and different response bits for each of its challenge values as shown in Figure 4.1. Since PUF response depends solely on the unique and random characteristics of physical devices, it is also unique for different instances, even if the two instances are exactly the same and use the exact same components. The very important security feature of a PUF circuit is its unclonability. It is because, an attacker may access to the circuit through reverse engineering and build an exact same copy of it using the

same components, but the response of the new device to a given challenge would be different from that of the original device to the same challenge.

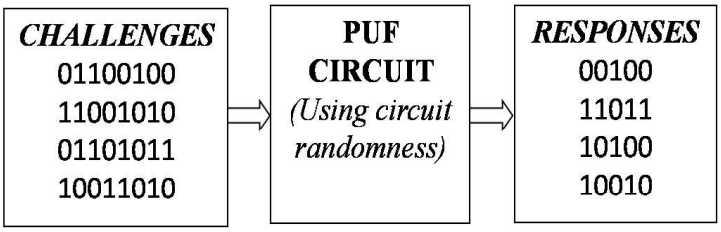


Figure 5.1: Block diagram of simple PUF block with challenge-response pairs

5.1.2 Taxonomy of PUFs

Over a decade, a variety of PUF constructions has been introduced. As shown in Figure 4.2, PUF constructions are broadly classified as non-electrical and electrical PUFs [5.12, 5.13].

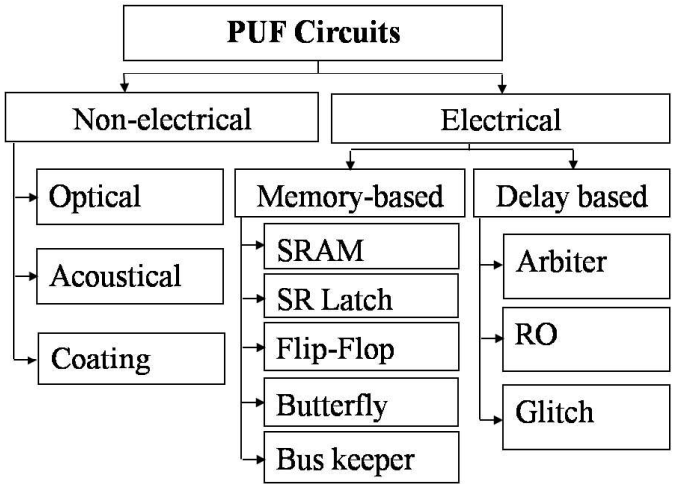


Figure 5.2: Taxonomy of PUF circuits

In general, electrical PUFs can be easily integrated into semiconductors and come in different flavors such as memory-based PUFs and delay-based PUFs. Memory-based PUFs include SRAM, flip-flop, latch, butterfly, bus-keeper PUFs [5.14-5.19], etc. Memory-based PUFs use randomness associated with the settling state of the bi-stable memory element (which

can contain only 1 bit of information) into its meta-stable state. However, the amount of unique responses of a memory-based PUF is limited by the number of its memory cells, i.e., the size of the underlying memory block.

Furthermore, delay-based PUF uses inherent gate delay associated with the manufacturing variations as its random parameter to produce response bits. In general, it consists of arbiter PUFs [5.20], RO-PUFs [5.21], glitch or Anderson PUFs [5.22], etc. The general approach is to design two identical signal paths consisting of wires and transistors that in theory should generate the same signal delay. However, due to manufacturing process variations, the physical characteristics of both signal paths will be slightly different and thus, the actual signal delay of each path will deviate from the ideal delay. These delay differences will be used to generate the PUF response bits. For example, an arbiter PUF [5.13] shown in Figure 4.3 consist of two identically designed signal paths consisting of wires, N switching components and an arbiter (i.e. a circuit used to determine which of several signals arrive first) at the end of both paths. The N switching components allow the signal paths to be configured according to an external input x = (x_0, …, x_N), i.e., the PUF challenge. To evaluate the arbiter PUF, both paths are simultaneously excited with the same signal. Depending on which of the two paths is faster, the arbiter generates output bit y which is used as PUF response.

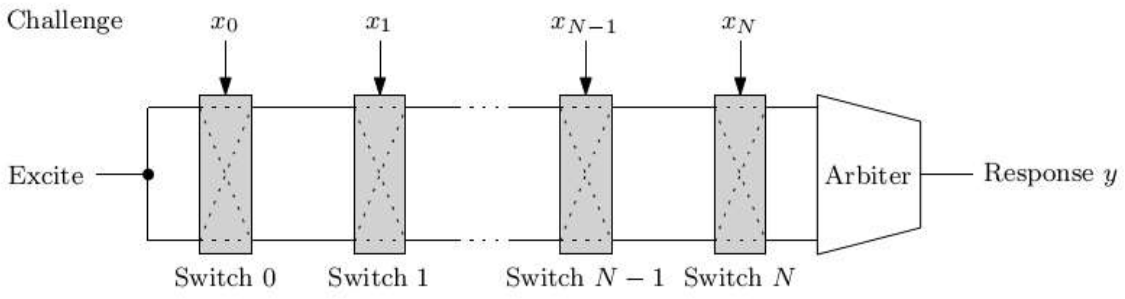


Figure 5.3: Basic arbiter PUF design [5.13]

Besides electric PUFs, there are many other PUF types that are not based on electric effects, including optical [5.23-5.24], acoustic [5.25], coating effects [5.26], etc. and they are categorized as non-electrical PUFs. In optical PUFs, whenever a laser beam is applied to the optical microstructure, it generates a random pattern that can be further processed to produce the PUF response. Acoustical PUFs are built upon the acoustical delay lines. Here, the mechanical vibration generated from electrical signal propagates through a solid medium (acoustical line) which includes random scatters. The reflected electrical signal from the end of the acoustical line has unique properties which depend on its random physical characteristics and generates unique PUF response. Unlike intrinsic random element based PUFs, a protective coating material is inserted in coating PUFs using random dielectric particles which have random properties in size, shape and location to generate response bits. However, non-electrical PUFs cannot be easily integrated into ICs as they often require non-standard manufacturing processes or special hardware components. Specifically, most non-electric PUFs require an external evaluation setup to measure the PUF response.

5.1.3 PUF Metrics

The most important and required performance metrics of any PUF circuits include uniqueness, reliability, uniformity and bit-aliasing [5.4]. The main idea behind PUF usage in security perspective is based on its uniqueness measure which represents the randomness of the PUF response bits. It is a measure of inter-distance variations (also called as hamming distance) of the response bits among different PUF instances i.e. specific challenge applied to two identical PUF instances at the same time and under the same conditions generates two different response bits and ideally this value should be 50%. As PUFs are used to generate secret keys, it is necessary to ensure that the PUF generates the same response (i.e. 100% reliability) to a given

challenge at different instances of time and under different environmental conditions such as voltage and temperature. The measure of a ratio of number of 1's and the total number of response bits is called as uniformity of a PUF circuit. Ideal PUF should have 50% uniformity i.e. 50% of the response bits are 1 and 50% are 0, which avoids the biased behavior towards a specific bit value. The randomness of a PUF is also measured using another measure known as bit-aliasing.

5.1.4 Hamming Distance and Reliability Estimation

For binary strings, a hamming distance (HD) between any two strings of equal length is defined as the number of bits that are different in the two strings. An example of HD calculation is shown below. Consider two strings of equal lengths "1110000100" and "1010011100" and HD is 3 bits i.e. 30% because only three bits are different when both strings are compared. Intra Chip HD is the HD calculated by comparing the responses generated at varying environmental conditions on the same chip [5.4]. In this work, Intra HD is the hamming distance between the responses generated at different temperatures or voltages and the operating temperature or voltage. Each response is compared with the response obtained at operating temperature or voltage and HDs are calculated. The intra HD is formulated as follows:

$$\text{Intra HD} = \frac{HD(R_x, R_{x'})}{n} \text{ X } 100\% \tag{5.1}$$

where R_x is the response of chip x at operating temperature or voltage, $R_{x'}$ is the response at varied temperature or voltage conditions and n is the total length of output response. This value represents the instability of the given PUF instance. Since this work concentrates on reliability estimation, this section elaborates reliability of a PUF instance. It is a measure of the stability of

the PUF response bits to a given challenge at different times and operating conditions. Ideally, the reliability of a PUF is 100%, means that the PUF under study generates the exact same response to a given challenge under different conditions, such as different temperatures or supply voltage values. The value of reliability can be derived by reducing the intra HD value from 100% as shown in Eqn. 5.2.

$$\text{Reliability} = 100\% - \text{Intra HD} \tag{5.2}$$

5.2 Reliability Estimation using Multi-Corner Timing Analysis

The basic design flow of a FPGA-based digital design starts with HDL coding (may include IP) and undergoes synthesis, mapping, PAR and bit file generation process using the vendor specific EDA tools. To prove design correctness, most of EDA tools support the HDL simulation at pre-synthesis, post synthesis and post PAR or layout. The netlist is generated in the synthesis stage whereas bitstream is generated after PAR. Finally, the bitstream is used to program the target device and deployed in the field. So far, the ideal design phase is elaborated. The complete flow diagram representing the ideal design phase to implement any PUF circuits in FPGA and to collect its response bits is shown in Figure 5.4.

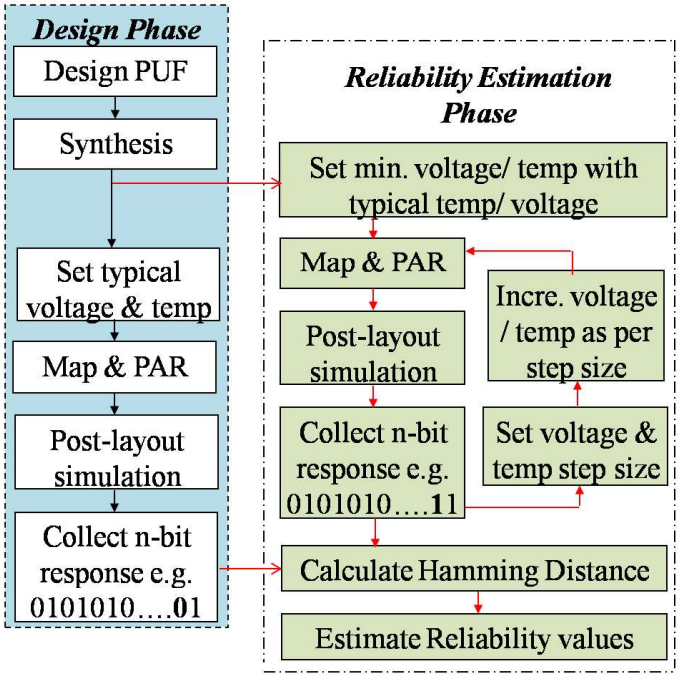


Figure 5.4: Flow diagram of reliability estimation of PUFs in FPGA devices using multi-corner timing analysis

During the design and reliability estimation phases, it is very important to make sure that PUF circuits are not optimized and also PUF hard-macros are widely used to avoid such scenarios. The designer knows that the multi-corner timing analysis option in EDA tool enables us to perform functionality and performance check at different voltage and temperature ranges as specified in datasheet during the design phase. As explained in Section 5.1.4, reliability analyses of PUFs are carried out by practically varying the device operating conditions such as supply voltage and junction temperatures. Instead of practically exposing the device to extreme operating conditions during testing or reliability analysis phase, the multi-corner timing analysis [5.27] based reliability estimation technique is proposed as shown in reliability estimation phase in Figure 5.4. During the design phase, required samples of original PUF response bits are collected using post-layout simulation at a typical operating voltage and temperature conditions.

Once sample responses are collected, the voltage and temperature step values with required intervals are set during reliability estimation phase using multi-corner option. Post-layout simulation for the modified operating condition is performed and sample response bits are collected. At each modified voltage or temperature, the hamming distance between original and modified response bits are calculated. Using reliability model explained in Section 5.1.4, the estimation of reliability values are calculated. The simulation results of reliability estimation using multi-corner timing analysis are discussed in the following section.

5.3 Results and Discussion

In order to perform reliability estimation of PUF circuits using multi-corner timing analysis, the RO or Anderson hybrid PUF circuit is implemented as shown in Figure 5.5 in Actel ProAsic3 device using Libero IDE v9.1 (free version). The randomly generated RO frequencies are used to produce the response bits in RO-PUF and the Anderson PUF uses the shift register multiplexer delay as its random parameter to generate the response bits. The hybrid PUF circuit combines these two ideas to increase the PUF randomness [5.5]. Note that, selection lines of multiplexers are the complement of each other and clock to Anderson PUF is generated using RO-PUF which introduces two kinds of randomness in the scheme. Since flip-flop output is AND'ed with the RO clock, the output of this scheme is either a 0 or a clock. After synthesis, the top module consumed 13 core cells and 2 IO cells of Actel resources and the device utilization is 5% as shown in Figure 5.6.

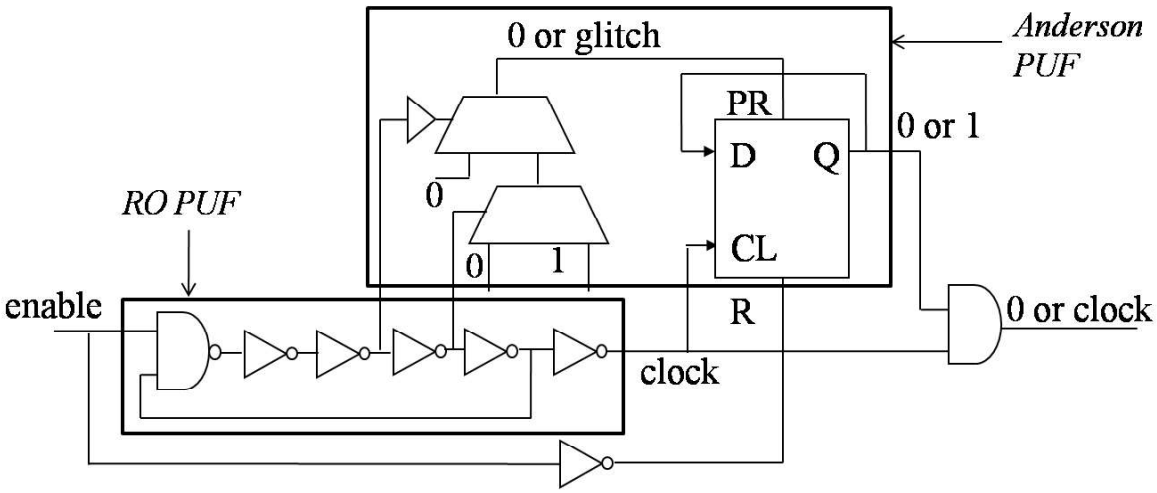


Figure 5.5: RO-Anderson hybrid PUF circuit

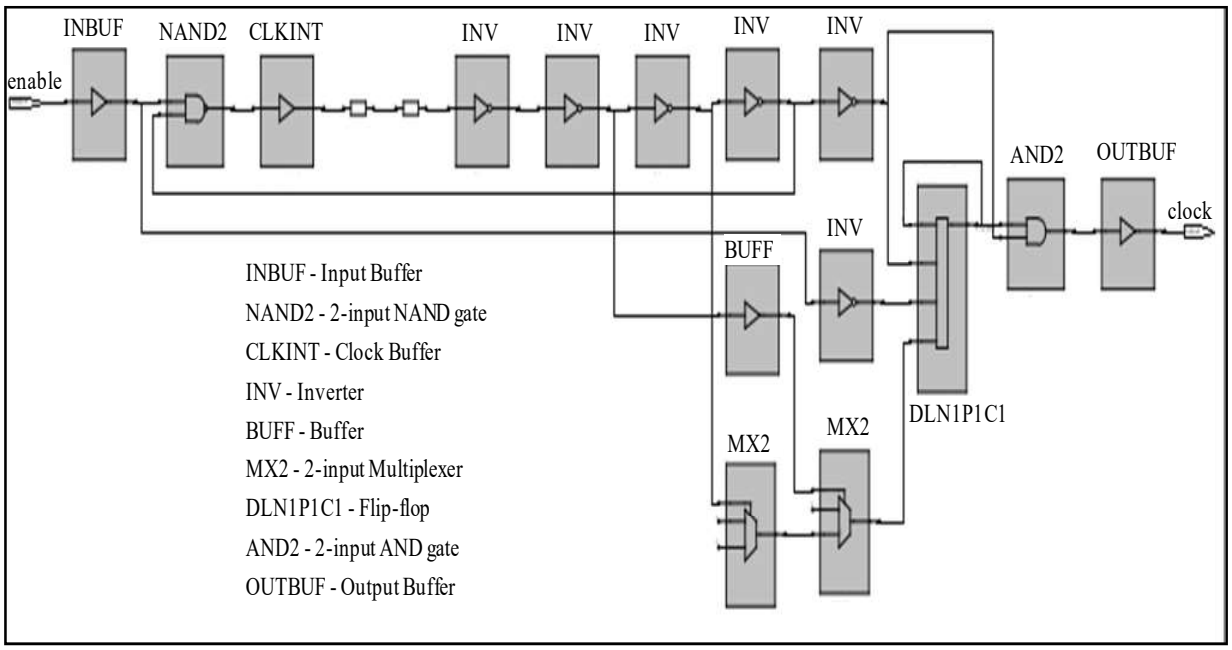


Figure 5.6: Technology mapped RO-Anderson hybrid PUF circuit

In general, an option to simulate at different operating conditions or process corners (i.e. at different temperature and voltage ranges), also called as multi-corner timing analysis, is available with EDA tools to enable the designer to check the design functionality at extreme operating conditions. Ideally, every device works at a typical voltage and temperature values as

per the data sheet. The operating voltage and temperature ranges of the selected device are (1.425 V to 1.575 V) and (-40 °C to 100 °C) respectively. Because PUF circuits are greatly used in high secure and authentication applications, the reliability of PUFs is a significant concern. As elaborated in Section 3.2.1, delay values in CMOS circuits are highly sensitive to voltage and temperature variations; hence, the reliability value of response bit or binary signature decreases with operating voltage and temperature variation. Initially, the response value (100 bit serial output) during design phase is measured, whose value is either 0 or clock, at a typical operating voltage and temperature value i.e. at (1.5 V and 25 °C). Using multi-corner timing analysis property, the design is simulated over (-40 °C to 100 °C) range with 10 °C step variation and (1.425 V to 1.575 V) with 0.025 V step variation. At each step temperature (-40 °C, -30 °C and so on), response bit of 100 samples are compared with that of 25 °C and hamming distance (i.e. no. of response bit variation) of each bit is calculated. Furthermore, reliability estimation is performed using the reliability model explained in Eqn. 5.2 and presented in Table 5.1.

Table 5.1: Reliability (%) estimation at different temperatures

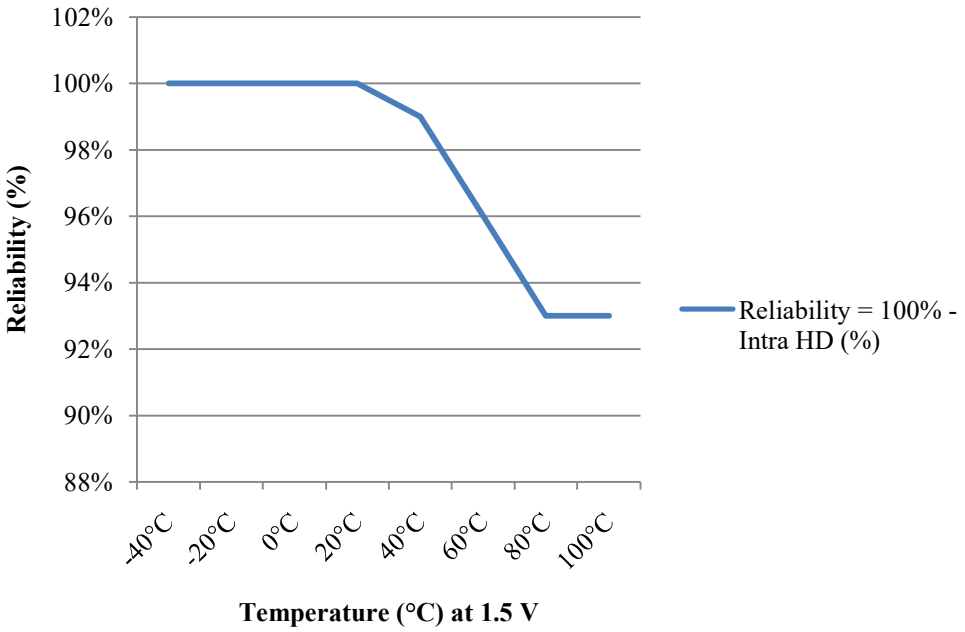
Temperature (°C) at 1.5 V	**HD**	**Intra HD = (HD/100)x100%**	**Reliability = 100% - Intra HD (%)**
-40 °C	0	0%	100%
-20 °C	0	0%	100%
0 °C	0	0%	100%
20 °C	0	0%	100%
40 °C	1	1%	99%
60 °C	4	4%	96%
80 °C	7	7%	93%
100 °C	7	7%	93%

Similarly, at each step voltage (1.425 V, 1.45 V and so on), response bit of 100 samples are compared with that of 1.5 V and reliability values are calculated as shown in Table 5.2.

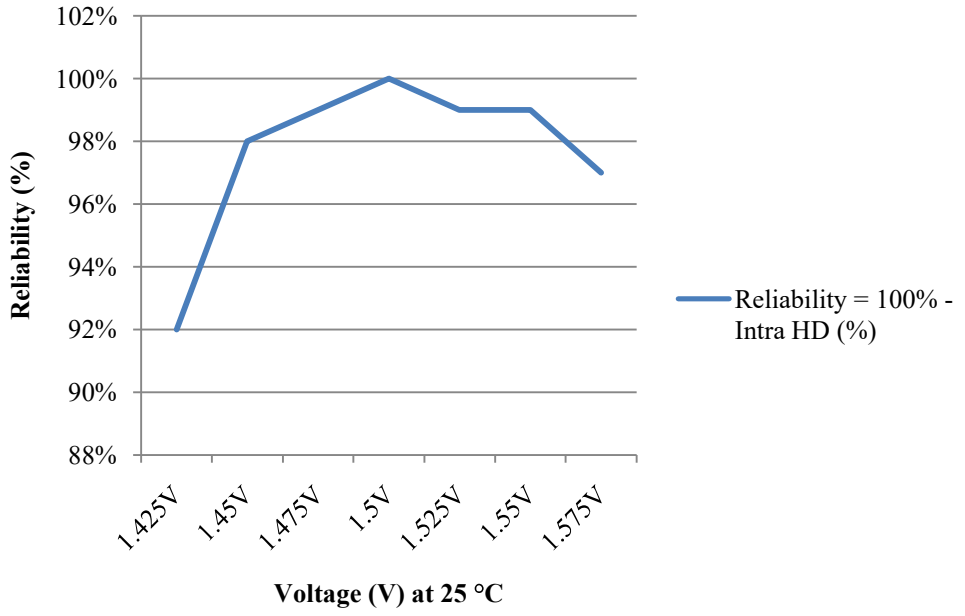
Table 5.2: Reliability (%) estimation at different voltages

Voltage (V) at 25 °C	**HD**	**Intra HD = (HD/100) x100%**	**Reliability = 100% - Intra HD (%)**
1.425 V	8	8%	92%
1.45 V	2	2%	98%
1.475 V	1	1%	99%
1.5 V	0	0%	100%
1.525 V	1	1%	99%
1.55 V	1	1%	99%
1.575 V	3	3%	97%

The results are plotted in Figure 5.7a and Figure 5.7b and it is clearly evident that the reliability values are at its maximum at typical operating conditions. It is reduced at high temperature and low voltage ranges as per the relationship between path delay, temperature and voltages. In addition, the reliability value also decreases at high voltage condition due to heat generation. Using this property, the proposed method supports the designer to estimate reliability values of PUFs during the design phase. Based on the results, the necessary error correction mechanisms shall be applied on PUF responses to correct any errors caused by the underlying physical processes. It also aids to reconstruct exactly the same key each time under all operating conditions.



(a)



(b)

Figure 5.7: Reliability estimation of PUF circuit using multi-corner timing analysis (a) Reliability estimation at different temperature and (b) Reliability estimation at different voltage

5.4 Summary

As the digital applications are highly vulnerable to various hardware security threats, the system designers rely more on cryptography-based authentication techniques. The generation of a secure binary signature using PUF circuits is one such solution. As mentioned, delays in CMOS circuits are highly sensitive to voltage and temperature variations which in turn affect PUF response bits. Hence, it is recommended to perform reliability analysis of PUFs before implementing them in the design. In this chapter, a technique to perform reliability estimation of PUF circuits implemented in FPGA devices using multi-corner timing analysis is successfully explored. The proposed method in this chapter enables the designer to perform simulation based reliability estimation at the design stage and this value can support reliability values calculated during experiments. Based on the results, the necessary error correction mechanisms such as cyclic redundancy check and forward error correction shall be applied on PUF output bits to reconstruct exactly the same keys each time under all operating conditions.

References:

[5.1] A. R. Sadeghi, S. Schulz, and C. Wachsmann, "Physical Security Primitives: A Survey on Physically Unclonable Functions (PUFs) and PUF-based Security Solutions," in Secure Smart Embedded Devices, Platforms and Applications, Springer, 2012.

[5.2] R. Kumar, H. Chandrikakutty, and S. Kundu, "On improving reliability of delay based physically unclonable functions under temperature variations," in Proc. Hardware Oriented Security and Trust, pp. 142-147, 2011.

[5.3] R. Maes, "An accurate probabilistic reliability model for silicon PUFs," in Proc. 15th Int. Workshop Cryptograph. Hardware Embedded Syst., pp. 73-89, 2013.

[5.4] A. Maiti, V. Gunreddy, and P. Schaumont, "A systematic method to evaluate and compare the performance of physical unclonable functions," in Proc. Embedded Syst. Des. FPGAs, pp. 245-267, 2013.

[5.5] S. Khoshroo, “Design and Evaluation of FPGA-based Hybrid Physically Unclonable Functions” Electronic Thesis and Dissertation Repository, Paper 1281, 2013.

[5.6] A. Garg, and T. T. Kim, “Design of SRAM PUF with improved uniformity and reliability utilizing device aging effect,” in Proc. Int. Symp. Circuits and Systems, pp. 1941-1944, 2014.

[5.7] Y. Lao, and K. K. Parhi, “Statistical analysis of MUX-based physical unclonable functions,” IEEE Trans. Comput. Aided Design Integr. Circuits Syst., vol. 33, no. 5, pp. 649-662, 2014.

[5.8] X. Xu, and D. E. Holcomb, “Reliable PUF design using failure patterns from time-controlled power gating,” in Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Nanotechnol. Syst. (DFT), pp. 135-140, 2016.

[5.9] A. Vijayakumar, V. C. Patil, and S. Kundu, “On improving reliability of SRAM-based physically unclonable functions,” J. Low Power Electron. Appl., vol. 7, no. 1, p. 2, Jan. 2017.

[5.10] P.H. Nguyen, D.P. Sahoo, C. Jin, K. Mahmood, and M. van Dijk, “MXPUF: Secure PUF Design against State-of-the-art Modeling Attacks,” IACR Cryptology ePrint Archive 2017: 572 (2017).

[5.11] M. Rostami, M. Majzoobi, F. Koushanfar, D. S. Wallach, and S. Devadas, “Robust and reverse-engineering resilient PUF authentication and key-exchange by substring matching,” IEEE Trans. Emerg. Topics. Comput., vol. 2, no. 1, pp. 37-49, 2014.

[5.12] R. Maes, Physically Unclonable Functions: Construction, Properties and Applications, Springer, 2013.

[5.13] C. Wachsmann, and A. R. Sadeghi, Physically Unclonable Functions (PUFs): Applications, Models, and Future Directions, 1st ed., Morgan & Claypool, 2014.

[5.14] J. Guajardo, S. Kumar, Geert-Jan Schrijen, and Pim Tuyls “FPGA intrinsic PUFs and their use for IP protection,” in Proc. Cryptographic Hardware and Embedded Systems, vol. 4727, pp. 63-80, Springer, 2007.

[5.15] D. Holcomb, W. Burleson, and Kevin Fu, “Initial SRAM state as a fingerprint and source of true random numbers for RFID tags,” in Proc. Workshop on RFID Security, Jul. 2007.

[5.16] R. Maes, P. Tuyls, and I. Verbauwhede, “Intrinsic PUFs from flip-flops on re-configurable devices,” in Proc. Benelux Workshop on Information and System Security, Nov. 2008.

[5.17] Vincent van der Leest, Geert-Jan Schrijen, Helena Handschuh, and Pim Tuyls, “Hardware intrinsic security from D flip-flops,” in Proc. ACM Workshop on Scalable Trusted Computing, pp. 53-62, 2010.

[5.18] Y. Su, J. Holleman, and B. P. Otis, “A digital 1.6 pJ/bit chip identification circuit using process variations,” IEEE Trans. Journal of Solid-State Circuits, vol. 43, no. 1, pp. 69-77, Jan. 2008.

[5.19] S. S. Kumar, J. Guajardo, R. Maes, G. J. Schrijen, and P. Tuyls, “Extended abstract: The butterfly PUF protecting IP on every FPGA,” in Proc. Hardware Oriented Security and Trust, pp. 67-70, Jun. 2008.

[5.20] L. Lin, D. Holcomb, D. K. Krishnappa, P. Shabadi, and W. Burleson, “Low-power sub-threshold design of secure physical unclonable functions,” in Proc. Int. Symp. Low-Power Electronics and Design, pp. 43-48, Aug. 2010.

[5.21] A. Maiti, J. Casarona, L. McHale, and P. Schaumont, “A large scale characterization of RO-PUF,” in Proc. Symp. Hardware Oriented Security and Trust, pp. 94-99, Jun. 2010.

[5.22] J. Anderson, “A PUF Design for Secure FPGA-based Embedded Systems,” in Proc. Asia and South-Pacific Design Automation Conf., pp. 40-48, 2010.

[5.23] R. Pappu, B. Recht, J. Taylor, and N. Gershen-Feld, “Physical one-way functions,” Science, vol. 297, pp. 2026-2030, 2002.

[5.24] P. Tuyls, B. Skoric, T. Ignatenko, F. Willems and G. J. Schrijen, Entropy estimation for optical PUFs based on context-tree weighting methods, Security with Noisy Data, Springer London, pp. 217-233, 2007.

[5.25] S. Vrijaldenhoven, Acoustical Physical Uncloneable Functions. M.S. thesis, Technische Universiteit Eindhoven, the Netherlands, pages 212, 2005.

[5.26] B. Škorić, S. Maubach, T. Kevenaar, and P. Tuyls, “Information-Theoretic Analysis of Capacitive Physical Unclonable Functions,” Journal of Applied Physics, vol. 100, no. 2, pp. 211, 2006.

[5.27] G. Sumathi, L. Srivani, D. Thirugnana Murthy, N. Murali, S. A. V. Satya Murty, and T. Jayakumar, “DSDPC: Delay Signatures at Different Process Corners based Hardware

Trojan Detection Technique for FPGAs," in Proc. IEEE Int. Conf. Robotics, Automation, Control and Embedded Systems, pp. 1-7, 2015.

6

Summary and Scope for Future Work of the Thesis

The present chapter summarizes the different conclusions derived from the research work carried out in the field of secure and reliable VLSI designs. It also provides the scope for future work in this genre to improvise the hardware security of digital systems and also in general.

6.1 Summary of the Thesis

The aim of the present dissertation is to study feasible hardware security threats in VLSI device based safety critical applications and to suggest feasible solutions to improvise the security measures. Incorporating a suitable anti-tamper solution in the VLSI design flow being so important against various hardware security threats, a number of approaches and its respective feasibility and performance measures based conclusion have been collected and summarized in this chapter.

a. Analysis on the potential hardware Trojan threats and defense solutions for PLD and ASIC life cycles is carried out. Here, various stages of PLD and ASIC life cycles are analyzed individually for the possible HT attacks. The state of the art HT prevention, detection and diagnosis techniques are mapped to the valid stages of PLD and ASIC life cycles. The summary basically deals with the likelihood of adhering to the proposed guidelines improves safety measures and protects safety systems against HT attacks.

b. To detect HTs inserted in field operating conditions of FPGAs by reverse engineering, DSDPC: Delay signatures at different process corners based HT detection technique for FPGAs is proposed. For this, the delay profile of original netlist has to be stored. At regular intervals, the field extracted netlist profile is compared with the stored profile to identify tampering. Since the simulation results of EDA tools are repetitive, mismatches among delay signatures clearly reveal the HT presence.

c. It is explicit that deriving single detection mechanism for all HT attacks is practically infeasible. Hence, investigation into hardware obfuscation based design security solutions is concentrated which shall ensure a certain level of design security by preventing against stealing of original design by analyzing and rebuilding during RE. Thus, to protect HDL code or IP core or IC design against various attacks, the structural modification based hardware obfuscation technique using different and random obfuscation cell structures is proposed for PLDs. This obfuscation scheme enables the circuit operation in two distinct modes such as obfuscated and functional modes. The mode control is performed by the application of a specific sequence of input vectors on initialization, called an “initialization key.” Without the initialization key, an adversary fails to comprehend the intended functional behavior of the circuit; hence, circuit tampering or malicious insertion by an adversary will have a high probability of either becoming functionally benign or easily detectable by conventional logic testing. In addition with logic obfuscation, usage of different and random obfuscation cells increases the complexity of RE. Results reveal that the proposed approach appears to be a quite useful technique for PLD designs and design overhead is well below the design constraints. The novel obfuscation metric calculations to quantify the complexity of RE

are recommended. Finally, the obfuscation process is completely automated and integrated with the PLD design flow using “perl”, “tcl” scripting languages, which will be highly helpful to the digital designers to obfuscate and protect their designs.

d. In addition to the proposed HT detection and logic obfuscation mechanisms, implementation of the device dependent secret key generation module in the PLDs is found to be one of the best solutions to avoid the side-channel attacks and various invasive attacks: The PUFs are widely used to address them. A PUF is a challenge-response module which generates unique, reliable and tamper-proof signatures for a given IC based on the process variations inherent in the IC manufacturing process, i.e. the output response is totally random and unpredictable. In this work, a method to perform simulation based reliability calculation of PUF modules at various temperature and voltage ranges is proposed using multi-corner timing analysis. The software implementation of the ring oscillator and Anderson hybrid PUF module is performed in FPGA devices and the multi-corner timing simulations were carried out using the vendor specific electronic design automation tools. From simulation results, the percentage of hamming distance is calculated and reliability estimation is performed. The proposed method supports the designer to estimate reliability values of PUFs during the design phase.

6.2 Scope for Future Work

The work presented in the thesis emphasizes on various hardware security threats and security mechanisms for VLSI designs. The thesis proposed new and efficient protection mechanism for hardware Trojans and hardware obfuscation based design security solutions for PLD based digital designs. Based on the successful completion of thesis objectives, the scope of

future work is outlined as follows: (1) Investigation into the detection mechanism of hardware Trojans inserted by foundry in blank FPGA devices i.e. to validate FPGAs before usage; (2) Experiments on DSDPC technique using FPGA based evaluation board; (3) Experiments to exploit the unused resources of PLDs to design the test logic; (4) Analysis of the effect of design obfuscation on circuit and system testability and reliability can also be investigated; (5) Integration of PUF circuit with the proposed logic obfuscation technique to generate the authentication or initialization sequence and (6) Evaluation of the hardware obfuscation technique against side-channel attacks.