5-20-2020

Physical Unclonable Functions for Authenticating and Preventing Reverse Engineering of Integrated Circuits and Electronics Hardware

Md Shahed Enamul Quadir
*University of Connecticut - Storrs, md.enamul_quadir@uconn.edu*

Follow this and additional works at: [https://opencommons.uconn.edu/dissertations](https://opencommons.uconn.edu/dissertations)

**Recommended Citation**
Physical Unclonable Functions for Authenticating and Preventing Reverse Engineering of Integrated Circuits and Electronics Hardware

Md Shahed Enamul Quadir, PhD
University of Connecticut, 2020

Electronics hardware is subject to a number of potential threats such as reverse engineering and counterfeiting. As a result, hardware authentication mechanisms and anti-reverse engineering techniques such as obfuscation and tamper-resistance are essential. In this thesis, we will present methods to approach these problems, and the primary research contributions of this thesis are a Low pass filter PUF for the authentication of PCBs and ICs; Key generation for hardware obfuscation using strong PUFs; and Session key generation using strong PUF modeling.

Physical Unclonable Functions (PUFs) are probabilistic circuit primitives that extract randomness from the physical characteristics of a device. In this work, we propose a novel PUF design based on resistor and capacitor variations for low pass filters (LoPUF). We extract the process variations present at the output of the filter with the use of an inverter to digitize the output and a counter to measure output pulse widths. We have created a process to select RC pairs that can be used to reliably generate authentication IDs. The
LoPUF has been evaluated in the context of both printed circuit boards and integrated circuits.

As a result of the increased use of contract foundries, IP theft, excess production and reverse engineering are major concerns for the electronics and defense industries. Hardware obfuscation and IP locking can be used to make a design secure by replacing a part of the circuit with a key-locked module. In order to ensure each chip has unique keys, we propose a strong PUF-based hardware obfuscation scheme to uniquely lock each chip that is less area intensive than previous work.

Communication with embedded systems can be problematic because they are limited in their capability to implement public key encryption and client-side authentication. In this work, we introduce a session key generation mechanism using PUFs. We propose a novel dynamic key generation method that depends on the ability to model certain PUF circuits using machine learning algorithms. Our proposed method also mitigates tampering attacks as no information is stored between subsequent keys. We have shown the effectiveness of our method with error-correcting capability to keep the outputs of the PUF from noise.
Physical Unclonable Functions for Authenticating and Preventing Reverse Engineering of Integrated Circuits and Electronics Hardware

Md Shahed Enamul Quadir

B.S., Bangladesh University of Engineering and Technology, 2009
MSEE, University of Alabama at Birmingham, 2012

A Dissertation
Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy at the University of Connecticut 2020
Copyright by

Md Shahed Enamul Quadir

2020
Physical Unclonable Functions for Authenticating and Preventing Reverse Engineering of Integrated Circuits and Electronics Hardware

Presented by
Md Shahed Enamul Quadir

Major Advisor
Prof. John A. Chandy

Associate Advisor
Prof. Lei Wang

Associate Advisor
Prof. Faqir Jain

University of Connecticut
2020
Dedicated to my wonderful parents

and my younger brother
Acknowledgement

First of all, I would like to express my sincerest gratitude to my major advisor Professor John Chandy during my Ph.D. work with his mentorship, patience, valuable feedback with his immense knowledge and motivation. I am very grateful to him because he contributed to my research by coming up with magnificent ideas and providing insightful suggestions and feedback as well as his support to overcome numerous obstacles. I sincerely thank my advisor for his continuous support and encouragement and couldn’t imagine a better mentor and advisor for my Ph.D.

Besides my advisor, I would like to thank Professor Lei Wang, Professor Faquir Jain, Professor Omer Khan, and Professor Helena Silva for their support, helpful comments, and encouragement that helped me to complete my thesis.

Furthermore, I am grateful to the Electrical and Computer Engineering department, and, University of Connecticut and their employees for their support during my graduate studies. I would like to thank all teaching faculties and laboratory technicians in the ECE and Physics departments for their help and support during my teaching. I would also like to acknowledge the financial support of the United States National Science Foundation, the ECE Department, the Department of Physics and the Graduate School of the University of Connecticut.

Also, I would like to acknowledge all of my teachers, lab mates, colleagues, and friends who were always beside me with their wisdom, assistance, and confidence.
Last but not the least, I would like to show my love to my dearest family: my great parents, and my dearest younger brother. I recognize and remember my father’s contributions in each and every achievement of my academic path. My dearest parents deserve my deep feeling of respect for their unconditional love and endless support in every step of my life. It is my greatest honor to dedicate my thesis work to my family.
# Contents

## 1 Introduction

1. Introduction ............................................. 1

## 2 Overview of reverse engineering and countermeasures

2.1 Background ............................................ 7

2.2 Equipment ............................................. 13

2.3 Chip-Level Reverse Engineering (RE) ......................... 15

2.3.1 Decapsulation ....................................... 16

2.3.2 Delayering .......................................... 17

2.3.3 Imaging ........................................... 19

2.3.4 Post-Processing .................................... 20

2.4 Chip-Level Anti-Reverse Engineering ......................... 22

2.4.1 Camouflage ........................................ 22

2.4.2 Obfuscation ........................................ 23

2.4.3 Other Techniques ................................... 24

2.5 Board-Level Reverse Engineering (RE) ...................... 25

2.6 PCB-level Anti-Reverse Engineering ......................... 31

2.7 System-level Reverse Engineering (RE) ..................... 32

2.7.1 Firmware/Netlist Information Representation ............. 33

2.7.2 ROM Reverse Engineering (RE) ..................... 35

2.7.3 EEPROM/Flash Reverse Engineering (RE) ............... 36

2.7.4 Reverse Engineering (RE) of FPGAs .................... 38

2.8 System-level Anti-reverse Engineering ...................... 41
2.8.1 Anti-reverse Engineering for ROMs .......................... 41
2.8.2 Anti-reverse Engineering for EEPROMs/Flashs .............. 44
2.8.3 Anti-reverse Engineering for FPGAs .......................... 46
2.8.4 Summary of Anti-RE Techniques for System-Level .......... 48
2.9 Reverse Engineering Summary .................................. 48

II Authentication of PCBs and ICs .............................. 50

3 Authentication of Printed Circuit Board ........................ 51
3.1 Introduction .......................................................... 51
3.2 Prior Related Work .................................................. 53
3.3 Proposed Methodology .............................................. 54
  3.3.1 Mux-Demux Switch ............................................. 55
  3.3.2 Low pass RC filter .............................................. 57
  3.3.3 Instrumentation Amplifier ..................................... 59
3.4 Results and Analysis using Analog Amplifiers .................. 61
  3.4.1 Experimental Setup ............................................. 61
  3.4.2 Experimental Results .......................................... 61
  3.4.3 Voltage Selection Algorithm for RC Pair ..................... 62
3.5 Digital Extraction of the input pulse from the RC filter ......... 66
  3.5.1 Extraction of PUF bits from the RC filter .................. 66
  3.5.2 Simulation Results ............................................. 69
3.6 Results and Analysis using Inverter ............................ 69
  3.6.1 Experimental Setup ............................................. 69
  3.6.2 Experimental Results .......................................... 70
  3.6.3 Pulse width Selection Algorithm for RC Pair ............... 71
3.7 Discussion ........................................................... 74
3.8 Conclusion .......................................................... 76

4 Authentication of IC based on RC variations ...................... 78
4.1 Introduction .............................................................. 78

4.2 Proposed Methodology .............................................. 79
   4.2.1 Low pass RC filter ............................................. 79
   4.2.2 Extraction of PUF bits from the RC filter ................. 80

4.3 Results and Analysis ................................................ 82
   4.3.1 Experimental Results .......................................... 85
   4.3.2 Pulse Width Selection Algorithm for RC Pair ............. 86
   4.3.3 Evaluation of LoPUF Under $V_{DD}$ and Temperature Variations ... 90
   4.3.4 Uniqueness .................................................... 92
   4.3.5 Randomness ................................................... 93
   4.3.6 Security measures of the LoPUF against Attacks .......... 94

4.4 Conclusion ............................................................. 94

III Hardware Obfuscation ............................................... 95

5 Key generation for Hardware Obfuscation using Strong PUF 96
   5.1 Introduction ...................................................... 96
   5.2 Prior PUF-Based Obfuscation Approaches ....................... 98
   5.3 Strong PUF-Based Key Generation .............................. 100
      5.3.1 Challenge Selection Process ................................ 102
      5.3.2 Chip Activation ............................................. 104
   5.4 Security Analysis .................................................. 105
      5.4.1 Manufacturing Attacks ..................................... 105
      5.4.2 In the Field Attacks ....................................... 105
   5.5 Performance Analysis ............................................. 106
      5.5.1 Hardware Complexity ....................................... 106
      5.5.2 Uniqueness and Reliability of the PUF .................... 106
   5.6 Evaluation .......................................................... 107
   5.7 Conclusions ........................................................ 110
IV Reliable Session Key Generation

6 Session Key generation using strong PUF modeling

6.1 Introduction

6.1.1 Transport Layer Security (TLS)

6.1.2 Physical Unclonable Functions

6.2 Proposed Approach

6.2.1 Enrollment

6.2.2 Key Reconstruction

6.2.3 Key Refresh

6.2.4 Error Correction

6.3 Discussion

6.4 Conclusions

7 An Efficient Algorithm for Extracting reliable Key from Noisy Data

7.1 Introduction

7.2 Related Works

7.3 Entropy Loss in Secure Sketch

7.4 Methodology

7.5 Conclusion

V Anti-RE of Flash Memory

8 Anti-RE Techniques for Flash Memories against Backside SCM Probing

8.1 Introduction

8.2 Related Work

8.3 Backside Protection Schemes

8.3.1 Scheme 1: Metal Connection from Floating Gate

8.3.2 Scheme 2: Adding p+ Layers

8.4 Conclusion
2.6 Standard (a) NAND gate and (b) NOR gate. These gates could be easily
differentiable by looking at the top metal layers. Camouflaged (c) NAND
gate and (d) NOR gate. These gates have identical top metal layers and
are, therefore, harder to identify [7]. . . . . . . . . . . . . . . . . . . . . . . 23

2.7 Marking convention on the Texas Instruments (TI) chips (a) first line and
(b) second line [8]. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26

2.8 Die marking on a Texas Instruments (TI) 65 nm processor [9]. . . . . . . . 27

2.9 PCB mounted in sample holder. . . . . . . . . . . . . . . . . . . . . . . . . 29

2.10 Layout design of the internal power layer. . . . . . . . . . . . . . . . . . . 29

2.11 Virtual slicing presents power layer. . . . . . . . . . . . . . . . . . . . . . 29

2.12 Reconstructed (a) top and (b) bottom layers. . . . . . . . . . . . . . . . . 29

2.13 Illustrations of (a) active layer programming ROM, (b) contact layer pro-
gramming ROM, (c) metal layer programming ROM, and (d) implant pro-
gramming ROM [10]. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

2.14 Illustrations of (a) EEPROM and(b) Flash [11]. . . . . . . . . . . . . . . . 35

2.15 FPGA hardware block diagram [12]. . . . . . . . . . . . . . . . . . . . . . 35

2.16 Optical inspection of (a) active layer programming rom [13], (b) contact
layer programming ROM [14], (c) metal layer programming ROM [15], and
(d) implant programming ROM before selective etch [15] . . . . . . . . . . . 36

2.17 Optical inspection of implant programming ROM after selective etch [15]. . 37

2.18 (a) SKPM scan and (b) SCM scan from the backside of Flash memory [16]. 37

2.19 Illustrations of one memory cell (a) anti fuse-OTP [17] and (b) FeRAM [18] 44

3.1 Physical Unclonable Function. . . . . . . . . . . . . . . . . . . . . . . . . . . 53

3.2 Block diagram of proposed LoPUF key generation. . . . . . . . . . . . . . . 55

3.3 8-to-1 Multiplexer [19] . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 55

3.4 Mux-demux switch. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56

3.5 Passive low pass filter. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 56

3.6 Transfer characteristics of Low pass filter(Ideal and Practical). . . . . . . . . 57

3.7 1st and 2nd order low pass filter. . . . . . . . . . . . . . . . . . . . . . . . . 59
6.7 Block diagram of proposed dynamic key generation with error correction 121
6.8 Session key generation protocol with error correction 122

7.1 Secure sketch and fuzzy extractor 126
7.2 Error-correcting code for the proposed PUF method 128
7.3 Generation Matrix of the (15,11) Hamming code 129
7.4 Probability of getting the correct codewords versus PUF output responses
   for (3,1) Hamming code 129
7.5 Probability of getting the correct codewords versus PUF output responses
   for (7,4) Hamming code 130
7.6 Probability of getting the correct codewords versus PUF output responses
   for (15,11) Hamming code 130

8.1 Front-side and backside of a chip (a) Simplified cross sectional view [20] (b)
   SEM cross section image of metal layers [6] 135
8.2 Cell structure of (a) EEPROM and (b) Flash [21] 139
8.3 SCM signal from backside of Flash memory [16] 140
8.4 Protection scheme 1 141
8.5 NAND flash layout 142
8.6 NOR flash layout 143
8.7 NAND flash layout with protection 144
8.8 NOR flash layout with protection 144
8.9 Cross-sectional image of NAND flash layout with protection 145
8.10 Top view image for flash layout with protection 145
8.11 Cadence layout verification for NAND flash with protection for 45 nm node 146
8.12 Zigzag metal protection method for NAND flash 147
8.13 Cross-sectional image of flash layout with P+ layer protection 147
## List of Tables

2.1 The Motivation for Reverse Engineering (RE) .................................. 8
2.2 Wet Etching Recipes for Different Types of Metals and Etching Process .......................... 18
2.3 Implementation Challenges of Anti-RE Techniques for Board-Level, where Very High = Most and Very Low = Least ........................................ 32
2.4 Comparison between SKPM and SCM Procedures ........................................ 38
2.5 Costs of Anti-RE Techniques and RE for System-Level, where Very High = ............. 49
3.1 Nominal values of resistors for the instrumentation amplifier ............................... 61
3.2 Total key values for different k, Δ, p_{eb} and acceptance regions with Analog Op-Amp .......................................................... 65
3.3 Total key values for different Δ, p_{eb} and acceptance regions with Inverter ............ 73
4.1 State of the art PUF area overhead comparison with our proposed method .......... 84
4.2 Probability of accepting 128 pairs for different Δ, where σ^i=0.3 .................. 90
4.3 Probability of accepting 128 pairs for different Δ, where σ^i=1 ..................... 91
4.4 Probability of accepting 128 pairs for different Δ, where σ^i=3 ..................... 91
4.5 % variation versus supply voltage for the RC pair of the LoPUF ....................... 91
4.6 % variation versus temperature for the RC pair of the LoPUF ......................... 92
5.1 ISCAS 85 circuit characteristics .......................................................... 108
5.2 Area overhead for obfuscation with key generation for different benchmarks suite with varying key size .......................................................... 109
5.3 Area (LUT) and slice registers for different PUF bits for different clock cycles. .... 111
8.1 Design and Manufacturing Costs and Implementation Challenges of Anti-Tampering Techniques, where Very High = Most and Very Low = Least . . 146
Part I

Introduction
Chapter 1

Introduction

Counterfeiting, piracy and reverse engineering have become a critical issue over the past decade due to offshore foundries and diversified supply chain [23, 24]. This has led to challenges in authenticating electronics, preventing reverse engineering, and ensuring that secure data on these electronics cannot be compromised. This dissertation presents a number of novel solutions and design strategies that address these challenges.

Authentication  Integrated circuit (IC) supply chain concerns include a) Trojan insertion and malicious modification, b) destructive and nondestructive reverse engineering and tampering to get important and critical information c) cloning. In 2010, a report published by Semiconductor Equipment and Materials International (SEMI) showed that 90% of companies have experienced infringement for Products, where 54% have faced severe infringement among them. IHS Technology has noted the cost of counterfeit product is over $169 billion [25]. Counterfeit and pirated integrated circuits (ICs) and electronics products also pose a great threat to the nation and government. It is reported that the Pentagon has purchased approximately 15% of spare and replacement semiconductors which are counterfeit [26]. Rogue counterfeiter entities sell pirated and cloned electronic devices as authentic. However, those devices will have issues with reliability, functionality, and performance [27, 28]. Similarly, Printed Circuit Boards (PCBs) are subject to intense counterfeiting.

Both ICs and PCBs have a diversified supply chain because they are a basic component
of electronic systems. The supply chain could have multiple untrusted parties - for example, trading partners, distributors, and retailers. Therefore, it is possible for a number of different forms of counterfeiting attacks on ICs and PCBs. The most prevalent attack on PCBs is cloning because they can be easily counterfeited by reverse engineering. Counterfeit ICs result from overproduction, cloning, and recycling. If a critical system consists of counterfeit product, it could cause serious degradation of performance and threat of security [29]. These counterfeited electronics can lead to failure of a system or theft of sensitive information. Therefore, anti-counterfeiting solutions for PCBs and ICs are an emerging concern for electronic system security and privacy to researchers as well as for industry [28].

Physically unclonable functions (PUFs) have been proposed as a mechanism to address counterfeits by providing a unique identifier that can authenticate an integrated circuit [30][31]. The PUF extracts a unique signature from the random manufacturing process variability of a device. The PUF based multiple-input multiple-output function is theoretically very hard to predict. A set of output responses are derived from input challenges known as Challenge-Response Pairs (CRPs) (See Fig. 1.1). Because the PUF is derived from a random variation of manufacturing variability, it should be very hard to predict PUF response for a specific challenge. Therefore, PUFs are an excellent choice for authentication and verification. In addition, PUF is used as a promising security measure because it could generate unique signatures. For example, authentication of the device and generation of ID is proposed using PUFs [30]. The authentication method by a designer

![Figure 1.1: Physical Unclonable Function.](image-url)
for a particular chip for a given CRP is shown in Fig. 1.1 and those CRPs will be recorded in the system. From the trusted database, the response could be then used for verification of the IC. The PUF response that is used for authentication should be random, unique and reliable in different environmental conditions. A number of PUF designs have been proposed in the literature including those based on arbiters, on ring oscillators, gate glitches, scan chains and memory arrays etc.

Furthermore, PUFs are divided into two main categories such as “Strong PUFs” and “Weak PUFs”. A strong PUF is a PUF with a very large challenge space that makes it ideal for authentication as described above. The CRPs for a strong PUF are stored in a database during enrollment. Therefore, end-users could authenticate the device by verifying the CRP with the database. In practice, many strong PUFs as implemented have been broken since they can be effectively modeled using machine learning techniques, thus defeating the unpredictable claim. A weak PUF, on the other hand, has a limited set of challenges (typically one), and is used as a way to generate unique keys or fingerprints for use internal to the chip.

In Chapters 3 and 4, we discuss a novel authentication method that utilizes resistor and capacitor pair variations to create a PUF output that can create a unique signature for authentication. The approach is suitable to both PCB and IC authentication.

**Hardware Obfuscation** There are countermeasures that are proposed in the literature to protect against reverse engineering such as obfuscation such as logic locking, camouflaging and other methods. Hardware obfuscation is an approach to prevent IC piracy and reverse engineering. Hardware obfuscation could be categorized into two types: Logic or functional locking and camouflage. The main idea behind logical locking obfuscation is that part of the design is replaced with a configurable module at the design stage. If the module is not activated by the designer, the chip will not function properly. During the post-fabrication activation process in a trusted design house, the chips can be activated by unlocking the obfuscated function with a secret key that may be burned into on-chip fuses. Those unlocked chips can then be sold to the open market. The stored key cannot be recovered without direct access to the on-chip fuses such as with probing attacks.
Therefore, an attacker cannot reverse engineer the design because of the obfuscation, and the chip cannot be overproduced without knowledge of the key. Furthermore, layout level techniques such as cell camouflage [7] could be used as hardware obfuscation and dummy contacts are used to protect against attackers. The layout of standard cells with different functionalities is made to appear identical in the camouflage technique. Camouflaging can make it more difficult to identify camouflaged gates with automated image tools. A problem with many of these approaches is that the key is not unique across all instances of the chip. Thus, if an attacker is able to retrieve the key by some method, it can unlock all chips and effectively overproduce the chips. Recently, PUFs have been used as a way to provide unique keys for obfuscation [43, 44, 45].

In Chapter 5, we present an improved PUF-based hardware locking scheme that uses a strong PUF with a subset of bits used to generate a key that can unlock a locked circuit. A mathematical probability model is developed which determines the feasibility of the PUF key.

Session Key Generation  Networks of consumer electronics and embedded systems are involved in most parts of our daily life. Therefore, it is essential that these systems ensure data confidentiality, integrity and privacy as well as provide mechanisms to trust device identity and authenticity. The authenticity of the electronic devices is an important factor because it confirms and strengthens the safety of the home electronics network and gives access to the user to control the network securely [46]. However, most consumer home electronics services rarely have mechanisms for client-side authentication such as certificates or public keys. As a result, communications could be attacked in a number of ways - e.g. a man-in-the-middle attack, denial of service attacks, static key leakage, etc.

In Chapter 6, we propose a session key generation method using strong PUFs without the requirement of secure memory, expensive tamper-resistant hardware, or a PUF access channel. The core of the approach is twofold: (1) Use a model of the PUF to simulate the behavior of the PUF without having direct access to PUF channels (2) Refresh keys periodically using a response challenge feedback loop.
Anti-reverse engineering techniques for flash  EEPROM/Flash memory technology had long been considered immune to advanced tampering techniques. For example, attackers cannot detect the memory contents by non-destructive X-Ray technology, because EEPROM/Flash logic states are represented by the presence/absence of electrons in a floating gate. Thus, they are not visible by simple imaging techniques as would circuit structure be revealed by the physical and geometric appearance [24]. Furthermore, traditional front-side destructive analysis using a scanning electron microscope (SEM) and transmission electron microscopes (TEM) will disturb the electron distribution in the floating gate (FG) [47]. However, more recently, Scanning Kelvin Probe Microscopy (SKPM) and Scanning Capacitance Microscopy (SCM) procedures have been proposed in [47] [16] [48] to extract the information of the EEPROM and Flash memory accurately. In Chapter 8, we present potential approaches to address reverse engineering and tampering of flash devices.

In the following chapter, we present a survey of various reverse engineering techniques.
Chapter 2

Overview of reverse engineering and countermeasures

2.1 Background

Reverse engineering (RE) is the process by which an object is examined in order to gain a full understanding of its construction and/or functionality. RE is now widely used to disassemble systems and devices in a number of different contexts, such as industrial design, cloning, duplication, and reproduction [49]. In this work, we will be focusing on the reverse engineering of electronic systems, which can be achieved by extracting their underlying physical information using destructive and nondestructive methods [50] [51].

The motivation for RE could be “honest” or “dishonest” as shown in Table 2.1 [6] [27] [52]. Those with “honest” intentions tend to perform RE for the following reasons: verification, fault analysis, research and development, and education about the workings of an existing product. In many countries, RE is legal as long as patents and design copyrights are not violated [53]. When RE is performed to clone, pirate or counterfeit a design, to develop an attack, or insert a hardware Trojan, these are considered “dishonest” intentions. If the functionality of a cloned system is close enough to the original, for example, then the “dishonest” entity or individuals could sell large amounts of counterfeit products without the prohibitive research and development costs required by the owner of the original [54].
Table 2.1: The Motivation for Reverse Engineering (RE)

<table>
<thead>
<tr>
<th>“Honest” Intentions</th>
<th>“Dishonest” Intentions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failure analysis and defect identification</td>
<td>Fault injection attacks</td>
</tr>
<tr>
<td>Detection of counterfeit products [27] [55]</td>
<td>Counterfeiting</td>
</tr>
<tr>
<td>Circuit analysis to recover manufacturing defects</td>
<td>Tampering</td>
</tr>
<tr>
<td>Confirmation of intellectual property (IP)</td>
<td>IP piracy and theft</td>
</tr>
<tr>
<td>Hardware Trojan detection [52]</td>
<td>Hardware Trojan insertion</td>
</tr>
<tr>
<td>Analysis of a competitor’s product, Obsolete product analysis</td>
<td>Illegal cloning of a product</td>
</tr>
<tr>
<td>Education and research</td>
<td>Development of attacks</td>
</tr>
</tbody>
</table>

There are several examples of reverse engineering (RE) and cloning of systems throughout history. During World War II, an American B-29 bomber was captured, reverse-engineered, and copied by the former Soviet Union [1]. The original and the clone (Tupolev Tu-4 bomber) are shown in Figure 2.1. The only difference between the B-29 and Tu-4 are the engines and cannons. Another example of RE took place during the Vietnam War [56]. In the late 1960s, the AQM-34G-R model of the United States Unmanned Aerial Vehicles (UAV), also known as a “drone” was lost in the mainland of China. The American technology was analyzed to create a replica called the WuZhen-5. The WuZhen-5 was subsequently used in China’s invasion of Vietnam in 1979. Even today, the US Department of Defense (DoD) is concerned about RE attempts being made against U.S. weapons systems [57].

Figure 2.1: An example of reverse engineering (RE) from World War II; (a) United States Air Force B-29 bomber and (b) Soviet Union Tupolev Tu-4 bomber which is a reverse-engineered copy of the B-29 [1].

Aside from RE of large systems, secret information such as critical design and personal information can also be extracted or cloned from electronic chips and printed circuit boards (PCBs). For example, the simple structure and increasing reliance on commercial-off-
the-shelf components makes it very easy to RE and clone a PCB. Reverse engineering of PCB and ICs could also result in the development of future attacks against them. For example, many smartcards today contain ICs that store personal information and perform transactions [56]. “Dishonest” parties could reverse engineer these ICs to access the confidential information of the card holder, commit financial crimes, etc.

Another concern in the electronics industry is IC piracy using RE [56] [58] [59]. In 2010, Semiconductor Equipment and Materials International (SEMI) published a survey about intellectual property (IP) infringement. The survey revealed that, of the 90% of companies that have experienced IP infringement, 54% faced serious infringement of their products [60]. Many “dishonest” companies can illegally copy the circuit and technology in order to mass-produce and sell pirated copies in the open market without authorization. On a related note, counterfeiting of ICs through RE is also a concern for military and industrials sectors. Counterfeit electronics also result in unrecoverable losses for the IP owner. Counterfeit ICs and systems may be tampered or otherwise less reliable, resulting in vulnerabilities and life-threatening issues.

To summarize, reverse engineering (RE) is a longstanding problem that is of great concern to today’s governments, militaries, and various industries due to the following: (1) the attacks and security breaches that could occur through the RE of classified military systems, financial systems, etc.; (2) the safety issues and costs resulting from unintended use of counterfeit products in critical systems and infrastructures; (3) the loss in profits and reputation for IP owners, which can result from the counterfeiting of products through the use of RE; (4) the negative impact that RE has on new product innovations, incentives for research and development, and - by extension - the worldwide job market.

As a result of these concerns, researchers, companies, and the defense departments of many nations are persistently seeking anti-RE techniques to prevent adversaries from accessing their protected products and systems. For example, the United States DoD is currently conducting research on anti-RE technologies that may prevent classified data, weapons, and IP from being compromised by foreign adversaries [61]. The objective of the DoD’s anti-tamper program is to obstruct unapproved technology transfer, maximize the costs of RE, enhance U.S./coalition military capacities, train the DoD community, and
educate the DoD community on anti-tampering technologies \cite{57}. Unfortunately, most of this work is classified and, therefore, is not available to the industrial sector or the wider research community.

Anti-RE techniques should have the ability to monitor, detect, resist, and react to invasive and noninvasive attacks. Several techniques could be used as anti-RE techniques. For example, tamper resistant materials and sensors have been used to resist theft or reverse engineering (RE) \cite{62}. Hard barriers like ceramics, steel, and bricks have been used to separate the top layer of the electronic devices so that tampering or RE attempts might be foiled by the destruction of the protective devices. To protect against pico-probing attempts, single chip coatings have also been applied. Many different packaging techniques could be used to protect a device: brittle packages, aluminum packages, polished packages, bleeding paint, as well as holographic and other tamper responding tapes and labels \cite{62}. Sensors of interest include voltage sensors, probe sensors, wire sensors, PCB sensors, motion sensors, radiation sensors, and top layer sensor meshes. Materials like epoxy with potting, coating, and insulating have been used to block X-ray imaging attempts.

Also, obfuscation software and hardware security primitives have been used for the protection of systems and software. These anti-RE techniques would be helpful for protecting confidential information from different types of RE attempts. Some other methods for protecting these systems are as follows: bus encryption, secure key storage, side channel attack protection, and tamper responding technology \cite{62,39}.

Figure 2.2: (a) Simplified cross sectional view \cite{2} and (b) layout of a CMOS inverter \cite{3}.

In this work, we shall cover the reverse engineering (RE) of electronic devices from chip
to system levels:

1. **Chip-level reverse engineering (RE):** A chip is an integrated circuit comprised of electronic devices that are fabricated using semiconductor material. A chip has package material, bond wires, a lead frame, and die. Each die has several metal layers, vias, interconnections, passivation, and active layers [2] [63]. In Figure 2.2(a) and Figure 2.2(b), a simplified cross sectional view and layout of a CMOS inverter are shown respectively. In Figure 2.2(a), polysilicon gates (G) of NMOS and PMOS transistors are connected together somewhere off the page to form the input of the inverter. The source (S) of the PMOS of the inverter is connected to a metal $V_{DD}$ line, and the source (S) of the NMOS is connected to a metal ground (GND) line. The drains (D) of the PMOS and NMOS are connected together with a metal line for the output of the CMOS inverter. The chip could be analog, digital, or mixed signal. Digital chips include application-specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), and memories. RE of chips can be nondestructive or destructive [56]. X-ray tomography is a nondestructive method of RE, which can provide layer-by-layer images of chips and is often used for the analysis of internal vias, traces, wire bonding, capacitors, contacts, or resistors. Destructive analysis, on the other hand, might consist of etching and grinding every layer for analysis. During the delayering process, pictures are taken by either a scanning electron microscope (SEM) or a transmission electron microscope (TEM).

2. **PCB-level reverse engineering (RE):** Electronic chips and components are mounted on a laminated non-conductive printed circuit board (PCB) [64] and electrically interconnected using conductive copper traces and vias [65]. The board might be single- or multi-layered depending on the complexity of the electronic system. Reverse engineering of PCBs begins with the identification of the components mounted on the board, its traces on the top and bottom (visible) layers, its ports, etc. After that, delayering or X-ray imaging could be used to identify the connections, traces, and vias of the internal PCB layers.
3. **System-level reverse engineering (RE):** Electronic systems are comprised of chips, PCBs, and firmware. A system’s firmware includes the information about the system’s operation and timing and is typically embedded within non-volatile memories (NVMs), such as ROM, EEPROM, and Flash. For more advanced designs with FPGAs (such as Xilinx FPGAs), the firmware-like netlists are also stored within the NVM memories. By reading out and analyzing the contents in the memory, reverse engineering can provide a deeper insight into the system under attack.

Based on the discussions above, the taxonomy of reverse engineering (RE) is shown in Figure 2.3. First, reverse engineering is performed to tear down the product or system in order to identify the sub-systems, packages, and other components. The sub-systems could be electrical or mechanical. In this work, we will focus on electrical sub-systems. The electrical sub-systems under analysis consist of hardware and firmware. A reverse engineer could analyze the FPGA, board, chip, memory, and software to extract all information. This work is concerned with RE when it is done with bad intentions and with anti-RE as a remedy against this form of RE. We examine this type of RE and anti-RE for each level, including equipment, techniques, and materials.

The rest of the study is organized as follows: In the next section, we will introduce the imaging and other specialized equipment that can be used for RE. In next Sections, we shall focus on RE and anti-RE at the chip-level. We will discuss board-level RE and anti-RE techniques in next Sections, respectively. After that, we will present system-level RE, and discuss anti-RE at the system-level.

Figure 2.3: Taxonomy of reverse engineering (RE).

![Figure 2.3: Taxonomy of reverse engineering (RE).](image)

Figure 2.4: IC parts (a) top view [4] and (b) cross sectional view [5].
2.2 Equipment

Advanced RE requires different kinds of specialized equipment. Throughout the chapter, we will refer to this equipment. Therefore, a short summary of each is provided below:

**Optical high/super resolution microscopy (Digital):** The limitations of conventional digital microscopy include limited depth of field, a very thin focus field, and keeping all parts on an object simultaneously in focus [66]. To overcome these limitations, optical high-resolution microscopes are now being used. Optical super resolution microscopes take a series of images and put them together to create a 3D image that reflects different heights. However, the use of optical microscopes can only be used to analyze PCB and chip exteriors because the resolution is too low for current chip feature sizes (<100 nm).

**Scanning Electron Microscopes (SEM):** In a scanning electron microscope (SEM), focused beams of electrons are used to produce images [67]. For a sample, the electrons interact with atoms, a process that produces signals for detection. Reverse engineers should start with a cross-section of an unknown chip. The scanning electron microscopes (SEM) could be used for analyzing the cross-section, as well as the composition and thickness of each layer of the die. The object could be magnified by 10 times to approximately 30,000 times. The scanning electron microscope (SEM) provides the following advantages over traditional microscopes:

- **Higher resolution:** The SEM has higher resolution and, with high magnification, it can resolve the features on the sub-micron level.

- **Large depth of field:** When a specimen (such as the internal elements of a chip) is focused for an image, the height of the specimen is called the depth of field. The SEM has a depth of field that is more than 300 times greater than that of a light microscope, which means that a specimen’s otherwise unobtainable details can be obtained with a SEM.

**Transmission Electron Microscopes (TEM):** With transmission electron microscopes (TEM), a beam of electrons is transmitted through and interacts with a sample [68]. Like SEMs, transmission electron microscopes (TEM) have a very high spatial resolution, which
can provide detailed information about the internal structures of a sample [69]. Also, TEM can be used to view a chip’s cross-section and its internal layers.

**Focused Ion Beam (FIB):** The working principle of a focused ion beam (FIB) is the same as a scanning electron microscope (SEM) except that, instead of using an electron beam, an ion beam is used. The ion beam enables one to do material deposition and removal with nanometer resolution, which can be used for TEM sample preparation, circuit editing, etc. There are different types of ion sources for the ion beam, but the most popular one is Gallium (Ga) liquid metal. The new generation of these tools is called Plasma FIB (PFIB), which works at a higher power and results in shorter material processing time.

**Scanning Capacitance Microscopy (SCM):** For the illustration of dopant profiles on the 10 nm scale of semiconductor devices, scanning capacitance microscopy (SCM) is used because of its high spatial resolution [51]. A probe electrode is applied at the top of the sample surface, and this electrode then scans across the sample. The change in electrostatic capacitance between the surface and the probe is used for obtaining information about the sample [70].

**High-Resolution X-Ray Microscopy:** X-ray microscopy is used to nondestructively test a sample, such as a chip or a PCB board. With this method, X-rays are used to produce a radiograph of the sample, which shows its thickness, assembly details, holes, vias, connectors, traces, and any defects that might be present [71].

**Probe Stations:** Probe stations are highly-precise manual probe units for wafers and substrates. They support a wide variety of electrical measuring, device and wafer characterization (DWC), failure analysis (FA), submicron probing, optoelectronic engineering tests and more. There are up to 16 positioners in these kinds of systems located on a vibration-isolated frame, which stabilizes the platen. These features enable a highly reliable and repeatable testing process down to the submicron level. A pull-out vacuum chuck stage holds the testing samples and the motorized platen, while the chuck and positioners provide enough flexibility to perform tests on many different samples.

**Logic Analyzers:** A logic analyzer is an electronic instrument that can observe and record multiple signals on a digital system or digital circuits simultaneously. The use of a logic
analyzer can facilitate reverse engineering (RE) at the chip, board, and system levels. In the case of FPGA bitstream reverse engineering (RE), the logic analyzer can be adopted to measure the JTAG communication signals between FPGA and external memory.

**Computer Numerical Control (CNC):** The need for automating machining tools, which are typically controlled manually, led to the creation of the computer numerical control (CNC) where computers control the process. CNCs can run mills, lathes, grinds, plasma cutters, laser cuts, etc. The motion is controlled along all three main axes, which enables three dimensional process.

### 2.3 Chip-Level Reverse Engineering (RE)

An integrated circuit (IC) typically consists of a die, a lead frame, wire bonding, and molding encapsulant \[14\] as shown in Figure 2.4.

The package of a chip can be classified in different ways. The materials that are used can be ceramic or plastic \[72\]. As ceramics are costly, plastics are commonly used as the package material. Packaging can also be wire-bond or flip-chip \[73\]. In wire-bond packaging, wires are connected to the lead frame. There are several types of wire-bonding: concentric bond rings, double bonds, and ball bonding. In contrast, flip-chip packaging is an IC technique that allows for a direct electrical connection between face-down (“flipped”, so that its top side faces down) electronic components and substrates, circuit boards, or carriers. This electrical connection is formed from conductive solder bumps instead of wires. Flip-chips have several advantages over wire-bond packaging: superior electrical and thermal performance, higher input-output capability, and substrate flexibility. However, flip-chips are often considered more costly than wire-bonds \[73\].

At the chip-level, the goal of the RE process is to find package materials, wire bonding, different metal layers, contacts, vias and active layers, and interconnections between metal layers. The RE process has several different steps:

- **Decapsulation:** Decapsulation exposes the internal components of the chip, which allows for the inspection of the die, interconnections, and other features.
• **Delayering:** The die is analyzed layer by layer destructively to see each metal, passivation, poly, and active layer.

• **Imaging:** An image is taken of each layer in the delayering process by using SEM, TEM, or SCM.

• **Post-processing:** In this process, the images from the previous step are analyzed, schematic and high level netlists are created for functional analyses, and the chip is identified.

Each of these steps is discussed in greater detail in the subsections below.

### 2.3.1 Decapsulation

First, reverse engineers identify the package materials and remove the chip’s packaging. Depot is the traditional method by which acid solution is used for removing the package [51]. A package may be made from different kinds of materials, so one has to be precise when choosing the acid. These acid solutions are used to etch off the packaging material without damaging the die and interconnections. Mechanical and thermal methods are used to remove a die from ceramic packages. These methods are applied to both polish the ceramic materials and remove the lids [51].

To remove the die package, one can use selective or non-selective methods. Wet chemical etching and plasma etching can be used as selective techniques, while non-selective techniques would be thermal shock, grinding, cutting, and laser ablation [14].

After decapsulation, the die needs to be cleaned before delayering and/or imaging can be performed because dust may be present, resulting in artifacts [74]. Different methods for cleaning the dust are outlined below [14]:

• **Spray cleaning:** A syringe filled with acetone is attached to a very fine blunt-tip needle. The syringe is then used to spray particles off of the die.

• **Acid cleaning:** To remove organic residues, fresh acid can be used after decapsulation.
Figure 2.5: SEM cross section image of metal layers of a chip for CMOS technology [6].

- **Ultrasonic cleaning:** Water, detergent (lab grade), or solvents can be used for cleaning after bare die decapsulation.

- **Mechanical swabbing:** The die should be gently brushed with an acetone-soaked lab wipe which should be lint-free to avoid contaminating the die. The sample is scratched carefully to avoid loosening the bond wires.

### 2.3.2 Delayering

Modern chips are made up of several metal layers, passivation layers, vias, contact, poly, and active layers. Reverse engineers must perform cross-section imaging of a chip using SEM or TEM to identify the number of layers, metal material, layer thickness, vias, and contacts. Figure 2.5 shows the cross-section of a CMOS chip with three metal layers. The knowledge from cross-sectional imaging is critical as it determines how the delayering must be performed (i.e., how thick are the layers, what types of conductors are used, etc.).

Several methods can be used simultaneously when a chip is delayered - methods such as wet/plasma etching, grinding, and polishing. A reverse engineer should determine the etchants needed and the time needed to remove each layer because the layout could be
Table 2.2: Wet Etching Recipes for Different Types of Metals and Etching Process

<table>
<thead>
<tr>
<th>Material to be Etched</th>
<th>Chemicals</th>
<th>Ratio</th>
<th>Etching Process and Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aluminum (Al)</td>
<td>$H_3PO_4 :$ Water : Acetic Acid : $HNO_3$</td>
<td>16:2:1</td>
<td>PAN Etch; 200 nm/min @ 25 C; 600 nm/min @ 40 C</td>
</tr>
<tr>
<td>Aluminum (Al)</td>
<td>NaOH : Water</td>
<td>1:1</td>
<td>May be used @ 25 C but etches faster at a higher temperature</td>
</tr>
<tr>
<td>Silicon (Si)</td>
<td>HF : $HNO_3 :$ Water</td>
<td>2:2:1</td>
<td>-</td>
</tr>
<tr>
<td>Copper (Cu)</td>
<td>$HNO_3 :$ Water</td>
<td>5:1</td>
<td>-</td>
</tr>
<tr>
<td>Tungsten (W)</td>
<td>HF : $HNO_3$</td>
<td>1:1</td>
<td>-</td>
</tr>
<tr>
<td>Polysilicon (Si)</td>
<td>$HNO_3 :$ Water : HF</td>
<td>50:20:1</td>
<td>Remove oxide first; 540 nm/min @ 25 C</td>
</tr>
<tr>
<td>Polysilicon (Si)</td>
<td>$HNO_3 :$ HF</td>
<td>3:1</td>
<td>Remove oxide first; High etch rate: 4.2 microns/min</td>
</tr>
<tr>
<td>Silicon dioxide $(SiO_2)$ – thermally grown</td>
<td>HF : Water</td>
<td>1:100</td>
<td>Very slow etch; 1.8 nm/min @ 25 C</td>
</tr>
<tr>
<td>Silicon dioxide $(SiO_2)$ – thermally grown</td>
<td>HF</td>
<td>-</td>
<td>Very rapid etch; 1.8 microns/min @ 25 C</td>
</tr>
<tr>
<td>Silicon nitride $(Si_3N_4)$</td>
<td>Refluxing phosphoric acid</td>
<td>-</td>
<td>Use at 180C; 6.5 nm/min @ 25 C; Plasma etching is preferred for removing $Si_3N_4$</td>
</tr>
</tbody>
</table>

dependent on the specific technology, which could be either CMOS or bipolar. For example, memory device vias are much higher than others, so etching is challenging because one has to remove a large amount of material. Several types of metals and required wet etchants are shown in Table 2.2.

Once the etchants are determined for delayering a specific layer and metal, a reverse engineer will begin with etching the passivation layer; then the reverse engineer will take an image of the highest metal layer; and, after that, the reverse engineer will etch the metal layer. This same process is repeated for each layer, including the poly and active layers. When delayering a chip, the layer surface has to be maintained as planar, and, one at a
time, each layer should be etched carefully and accurately. Also, the layer thickness of a chip could vary because of manufacturing process variations. The best approach is to have one die for every level of delayering. For example, when delayering is done for a four-layer chip, a reverse engineer could use four dies for each metal layer of the chip.

To delayer a chip accurately, an advanced laboratory should have one or more of the following pieces of mechanical equipment: a semi-automated polishing machine, a semi-automated milling machine, a laser, a gel etch, a computer numerical control (CNC) milling machine, and an ion beam milling machine.

When the chip has been delayered, one could face the following challenges:

- **Planarity of the layer**: The planarity of the layer could be conformal or planarized. In a conformal layer, some portion of the different layers and vias could appear on the same plane. But, in a planarized layer, only one layer appears at a time. Conformal layers are more challenging.

- **Material removal rate**: The equipment could be slow or fast and could underetch or overetch.

- **Die size**: Thickness, length, and width can vary.

- **Number of samples**: There may not be enough parts to image each layer separately (i.e., information on a layer could be missing if delayering is not done accurately).

- **Selectivity of the material**: One must be careful to remove one material but not another (e.g., removing a metal layer without affecting the vias).

2.3.3 Imaging

During the delayering process, thousands of high-resolution images are taken to capture all the information contained in each layer. Later these images can be stitched together and then studied to recreate the chip. For the purposes of imaging, many high-resolution microscopes and X-ray machines could be used as discussed in Section 2.2.
2.3.4 Post-Processing

The post-processing or circuit extraction after delayering consists of the following steps: (i) image processing, (ii) annotation, (iii) gate-level schematic extraction, (iv) schematic analysis and organization, and (v) high-level netlist extraction from the gate-level schematic. Each of these steps is described in greater detail below.

Image Processing

Taking images manually is becoming increasingly difficult because the size of the ICs is shrinking, along with many of their features [51]. Advanced electrical labs now use automated instruments (X-rays, SEMs, digital microscopes), which are equipped to take images of entire layers of ICs and PCBs. Then, the automated software can be used to stitch the images together with minimal error and synchronize the multiple layers without misalignment. Also it is important to establish the lineup of the layers’ contacts and vias before the extraction.

Annotation

After the completion of the aligned layers and stitched images, the extraction of the circuit starts. This stage in the process includes making note of transistors, inductors, capacitors, resistors, diodes, other components, the interconnection of the layers, vias, and contacts. The circuit extraction could be an automated or a manual process. For example, Chipworks has an ICWorks Extractor tool that can look at all the imaged layers of the chip and align them for extraction [51]. The tool can be used to view several layers of a chip in multiple windows simultaneously. The ICWorks extractor tool might also be used for the annotation of wires and devices. Image recognition software (2D or 3D) is used for the recognition of standard cells in digital logic. Automated image recognition software helps facilitate the extraction of large blocks of digital cells quickly.
Gate-Level Schematic Extraction

Sometimes the images are imperfect, as the images may be taken manually. Also the annotation process and image recognition for digital cells could be erroneous. Therefore, verification is needed before the creation of a schematic. Design rule checks could be used to detect any issues related to minimum-sized features or spaces, wire bonding, vias, and connections \[51\]. After this stage, tools such as ICWorks can extract an interconnection netlist from which a flat schematic could be created. The schematic could be checked for any floating nodes, shorted input or output, or supplies and nets that have no input or output. The annotations, netlist, and schematic are dependent on each other, so changing one could affect the others.

Schematic Analysis and Organization

The schematic analysis should be done thoughtfully and carefully with proper hierarchy and design coherence. For the analysis and organization of a schematic, the reverse engineer could use public information on the device, such as its datasheet, technical report, marketing information, and patents. This could help facilitate an analysis of the architecture and circuit design. Some structures, such as differential pairs and bandgap references, could be easily recognizable.

High-Level Netlist Extraction from Gate-Level Schematic

After circuit extraction is performed on the stripped IC (derivation of circuit schematic diagram), several techniques \[75\] \[76\] \[77\] \[78\] could be applied to get the high-level description for analysis and validation of the functionality of the chip using simulation. \[75\] propose reverse engineering (RE) from a gate-level schematic of ISCAS-85 combinational circuits to get the circuit functionality by computing truth tables of small blocks, looking for common library components, looking for structures with repetition, and identifying bus and control signals. \[76\] present RE of gate-level netlists to derive the high-level function of circuit components based on behavioral pattern mining. The approach is based on a combination of pattern mining from the simulation traces of the gate-level netlist and in-
interpreting them for the pattern graph. The authors in [77] propose an automatic way to
derive word-level structures which could specify operations from the gate-level netlist of
a digital circuit. The functionality of logic blocks is isolated by extracting the word-level
information flow of the netlist while considering the effect of gate sharing. A variety of
algorithms is used in [78] to identify the high-level netlist with module boundaries. The
algorithms are applied for verification to determine the functionality of components such
as register files, counters, adders, and subtractors.

2.4 Chip-Level Anti-Reverse Engineering

There are several approaches for the anti-reverse engineering of integrated circuits, which
include camouflage, obfuscation, and other techniques. These methods are described in
more detail below.

2.4.1 Camouflage

Layout level techniques such as cell camouflage [7] [79] and dummy contacts could be used
to hinder adversaries who want to perform RE on a chip. In the camouflage technique,
the layout of standard cells with different functionalities is made to appear identical. One
can introduce camouflage to a standard gate by using real and dummy contacts, which can
enable different functionalities, as shown in Figure 2.6. In Figure 2.6(a) and Figure 2.6(b),
the layouts of two-input NAND and NOR gates are shown. These gates functionalities
can be easily identified by their layouts. In contrast, Figure 2.6(c) and Figure 2.6(d)
show camouflaged two-input NAND and NOR gates with layouts that appear identical.
If regular layouts are used for standard gates, automated image processing techniques
can easily identify the functionality of the gates (see Figure 2.6(a) and Figure 2.6(b)).
Camouflaging (see Figure 2.6(c) and Figure 2.6(d)) can make it more difficult to perform
RE with automated tools. If the functionality of the camouflage gates of the design is not
correctly extracted, the adversary will end up with the wrong netlist.
2.4.2 Obfuscation

Obfuscation techniques entail making a design or system more complicated in order to prevent reverse engineering (RE), while also allowing the design or system to have the same functionality as the original. There are several different obfuscation approaches in the literature [80] [42]. The HARPOON (HARdware Protection through Obfuscation Of Netlist) method could be used against piracy and tampering, and the technique could provide protection at every level of the hardware design and manufacturing process [42]. The proposed approach is achieved by obfuscating the functionality by systematically modifying the state-transition function and internal logic structure of the gate-level IP core. The circuit will traverse obfuscated mode to reach normal mode only for specific input vectors, which are known as the “key” for the circuit.

[80] proposed a technique of interlocking obfuscation in the Register Transfer Level (RTL) design which could be unlocked for a specific dynamic path traversal. The circuit has two modes: entry mode (obfuscated) and functional mode. The functional mode will be operational when there is a formation of a specific interlocked Code-Word. The Code-Word is encoded from input to the circuit, which is applied in entry mode to reach the functional mode. This Code-Word is interlocked into the transition functions and is protected from
reverse engineer by increasing the interaction with the state machine. Furthermore, the additional benefit is that any minor change or alteration to the circuit made by an adversary will be magnified due to the interlocking obfuscation. The proposed technique has a large area overhead, so there is a trade-off between the area overhead and the level of protection. Higher protection levels require larger overheads.

2.4.3 Other Techniques

Today, most companies are fabless, meaning that the fabrication of chips is outsourced. A semiconductor foundry is given the design [81] to fabricate the chips. To accomplish post-fabrication control of the ICs that are produced in such plants, IC hardware metering protocols have been put in place to prevent IC piracy [82] [58]. ICs can be identified by active metering, which is a process by which parts of the chip can be used for locking and unlocking by the design house. Physical unclonable functions (PUFs) can be used as secret keys to protect from cloning [83] [82]. PUF is very difficult to duplicate. Therefore, RE and cloning of the whole chip could be possible, but the reverse engineer would not be able to activate the cloned chip.

The authors in [60] have proposed a reconfigurable logic barrier scheme which separates information flow from the inputs to the outputs. This technique is used in the IC pre-fabrication stage for protection against IC piracy. The information could flow with the correct key, but the barrier would interrupt flow for the incorrect key. The main difference between the logic barrier scheme and the obfuscation techniques described in Section 2.4.2 is that the logic barrier scheme is based on the proper locking locations of the barrier in the design instead of randomized ones. This technique is used for effectively maximizing the barrier with minimum overhead by utilizing better-defined metrics, node positioning, and enhancing the granularity from XOR gates to look-up tables (LUTs).

An external key could be placed in every chip for protection against IC piracy. This method is called EPIC (End Piracy of Integrated Circuits) [39]. This key is produced by the IP holder and is unique. Manufacturers must send the ID to the IP holder for the chip to become functional, and the IP holder must then send the activation key to enable the activation of the chip with the ID. The random ID is generated by several techniques. This
ID is generated before the testing of the IC. This key prevents cloning of the IC from reverse engineering (RE) and controls how many chips should be made. The EPIC technique’s limitations include complex communication to the IP holder, which could impact test time and time to market. Also, this technique requires higher levels of power consumption.

[84] proposed a bus-based IC locking and activation scheme for preventing unauthorized manufacturing. The technique involves the scrambling of the central bus so that the design can be locked at the manufacturing site as a means of guaranteeing the chip’s uniqueness. The central bus is controlled by both reversible bit-permutations and substitutions. A true number generator is applied to establish the code for the chip, and the Diffie-Hellman key exchange protocol is employed during activation.

2.5 Board-Level Reverse Engineering (RE)

The goal of board-level RE is to identify all components on the board and the connections between them. All of the components used in a design are called the bill of materials (BOM) [49]. The components and parts of a printed circuit board (PCB) could be any of the following: microprocessors, microcontrollers, decoupling capacitors, differential pairs, DRAMs, NAND flashes, serial EEPROMs, serial NOR flashes, and crystals/oscillators. There could be silkscreen markings, high-speed serial/parallel ports, program/debug ports, JTAGs, DVIs, HDMI, SATAs, PCIs, Ethernets, program/debug ports, and display ports [51] [85]. To identify the components, test points, and parts of the PCB, silkscreen markings are often used [49]. For example, D101 may be a diode, and Z12 might be a zener diode.

IC Identification via Chip and Die Markings: Some electronic components mounted on the PCB can be identified easily through the use of IC markings, but fully custom and semi-custom ICs are difficult to identify. Using standard off-the-shelf parts with silkscreen annotations will assist the RE process. If the ICs have no markings, then the manufacturer’s logo can give an idea of the functionality of the chip. Custom devices, which are developed in-house, are difficult to identify [49] because a custom device could be undocumented, or documentation could be provided only under a non-disclosure agreement.

IC markings can be divided into the following four parts [86]:

25
- The first is the prefix, which is the code that is used to identify the manufacturer. It could be a one- to a three-letter code, although a manufacturer might have several prefixes.

- The second part is the device code, which is used to identify a specific IC type.

- The next part is the suffix, which is used to identify the package type and temperature range. Manufacturers modify their suffixes frequently.

- A four digit code is used for the date, where the first two digits identify the year and the last two identify the number of the week. And, manufacturers could cipher the date into a form only known by them.

The marking conventions of a Texas Instruments (TI) chip for the first and second line is shown in Figure 2.7. The TI chips could have an optional third and fourth line with information related to the trademark and copyright. After identifying the manufacturer and IC markings, the reverse engineer could find the detailed functionality of the chip from the datasheets, which are available on the Internet [87] [88].

![Figure 2.7: Marking convention on the Texas Instruments (TI) chips (a) first line and (b) second line](8)

If the IC marking is not readable because it has faded away due to prior usage in the field or the manufacturer did not place a marking for security purposes, the reverse engineer could strip off the package and read the die markings to identify the manufacturer and the chip’s functionality [86]. The die marking could help identify the mask number, part number, date of the die mask completion or copyright registration, company logo, and the trademark symbol. An example of the die marking on a Texas Instruments (TI) 65
nm baseband processor is shown in Figure 2.8. A die marking could match the package marking depending on the manufacturer. Then, the datasheet information could be used to assess the die. Die markings are similar within families of chips made by the same manufacturer [89], so if someone can find the functionality of one chip, then they can also identify the functionality of the chip family because of the almost similar die markings that are shared by the chips in that family. For example, the Qualcomm MSM8255 processor is identical to the MSM7230 in both functionality and design, and both chips are from the Snapdragon family of ICs [89]. The only difference between these two chips is their clock speed.

![Die marking on a Texas Instruments (TI) 65 nm processor](image)

Figure 2.8: Die marking on a Texas Instruments (TI) 65 nm processor [9].

After identifying the components of the PCB, the reverse engineer would want to identify the PCB type, which could be any of the following: single-sided (one copper layer), double-sided (two copper layers), or multi-layered. In multi-layered PCBs, chips are connected to each other on the front and the back, as well as through the internal layers. Some of the internal layers are used as power and ground layers. Conductors of different layers are connected with vias, and delayering is needed to identify these connections.

**Destructive Analysis of PCBs:** Before PCB delayering, images of the placement and orientation of all the outer layers’ components are captured [49]. Then, the components could be removed, drilled hole positions could be observed, and it could be determined whether there are any buried or blind vias. The PCB delayering process is similar to the one described for chips and, therefore, will not be discussed further. After the PCB
is delayered, images of each layer can be taken [85, 89]. Then the composition and the thickness of the layers should be noted. It is important to track the impedance control of high-speed signals and the characteristics of the PCB. The dielectric constant, prepreg weave thickness, and resin type should also be determined [49].

**Non-destructive 3D Imaging of PCBs Using X-ray Tomography:** X-ray tomography is a non-invasive imaging technique that makes it possible to visualize the internal structure of an object without the interference of over- and under-layer structures. The principle of this method is to acquire a stack of two dimensional (2D) images and then use mathematical algorithms such as the direct Fourier transform and center slice theory [91] to reconstruct the three dimensional (3D) image. These 2D projections are collected from many different angles depending on the quality needed for the final image. The object properties, such as dimension and material density, are important to consider in the selection of the tomography process parameters: source/detector distance to object, source power, detector objective, filter, exposure time, number of projections, center shift, and beam hardening. Internal and external structures will be ready to analyze when the 3D image is reconstructed [85]. More information on tomography parameters is available in [92].

As an example, we have analyzed the traces and via holes of a four-layer custom PCB using a Zeiss Versa 510 X-ray machine. To make sure that we can observe features on the board, we selected a fine pixel size, which gives us high enough image quality. After several rounds of optimization, the tomography parameters for obtaining the best quality images are selected. The process is completely automated after setting the parameters and can be performed without the need for oversight, and it should be widely applicable to most PCBs.

For the four-layer custom board in Figure 2.9, all traces, connections, and via holes are clearly captured. In order to validate the effectiveness of the tomography approach, the results are compared with the board design files previously used to produce the PCB. The board includes a frontside, backside, and two internal layers. The internal layers correspond to power and ground. The via holes connect the traces on two sides of the board and are
also connected to either power or ground layers. The internal power layer is presented in the design layout in Figure 2.10.

Figure 2.9: PCB mounted in sample holder.

![Figure 2.9: PCB mounted in sample holder.](image)

Figure 2.10: Layout design of the internal power layer.

![Figure 2.10: Layout design of the internal power layer.](image)

The 3D image of the board is reconstructed using a combination of thousands of virtual 2D slices. These slices can be viewed and analyzed separately. The thickness of each of these is same as the pixel size (that is, 50 microns). In Figure 2.11 one slice is provided, which shows the information of the internal power layer.

Figure 2.11: Virtual slicing presents power layer.

![Figure 2.11: Virtual slicing presents power layer.](image)

(a) (b)

Figure 2.12: Reconstructed (a) top and (b) bottom layers.

![Figure 2.12: Reconstructed (a) top and (b) bottom layers.](image)

By comparing the tomography results and the design layout of the board, one can see a clear difference between the via holes that are connected and those that are not connected to the internal layer. Soldered joints constitute a highly X-ray absorbing material and result in white contrast for the associated pixels; however, plastic has a lower density and is more X-ray transparent, which results in a dark contrast. So, one can easily determine which via holes are connected to an internal layer. The same principle will let us detect
the traces on the side layers of the board due to the attendance of copper on the traces, as shown in Figure 2.12.

**Netlist Extraction After Imaging:** After capturing images of the PCB via delayering or X-ray tomography, connections between all the components could be discovered, which would yield a PCB layout netlist. Then, commercial tools could be used for converting the layout back into schematic [93]. To create the netlist from the collected images, one should verify the following:

- Connection between the components of original board; a datasheet could be helpful to find the connection for original functionality
- Unexpected shorts and hanging VDD
- Pin connections between components

Several techniques have been used for analyzing X-ray images in prior work [94] [95] [96] [97] [98]. In [94], a visual inspection system is used for PCBs. The elimination-subtraction method is used, which subtracts the perfect PCB image (the template) from the inspected image and locates the defects in the PCB. An image of the raw PCB is read in [95] and then a structuring element is applied to an input image using a morphological operation. After that, a dilation and erosion function is applied so that a fine-segmented image of the PCB tracks can be achieved. [96] applied an automatic Verilog HDL Model Generator, which includes the image processing technique that is used to identify the components and their connections. After that, a circuit graph is obtained, which corresponds to a primitive schematic circuit of the board. Finally, verilog HDL is generated from the circuit graph. A verilog XL simulator is used for testing the performance. In [97], the layers of the Circuit Card Assemblies/Printed Circuit Boards (CCA/PCB) are separated using X-ray stereo imaging. The focus is to identify the solder joints and traces on the different layers of a multi-layered PCB. In the automated process technique [98], photos are taken from one- or two-layer printed circuit boards (PCBs). Then, a C++ program is used to automatically reverse engineer the netlist.
2.6 PCB-level Anti-Reverse Engineering

Ensuring the complete protection from PCB-level RE is a difficult task, thus the goal of anti-RE methods is to simply make RE prohibitively expensive and time consuming. A summary of PCB-level anti-RE techniques is provided below [49]:

1. Tamper-proof fittings (such as torx), custom screws shapes, adhesively-bonded enclosures, and fully potting the space around a PCB could be used for protection against physical attacks.

2. Custom silicon, unmarked ICs, missing silkscreens with minimum passive components, and a lack of information from the internet could complicate RE. Also, the elimination of JTAG and debug ports from silicon can make the RE process harder.

3. Ball grid array (BGA) devices are better because such devices do not have exposed pins. Back-to-back BGA placement in a PCB board could be most secure because of the inaccessibility of the unrouted JTAG pins with controlled depth drilling on any side of the PCB board. For back-to-back BGA placement, the PCB needs to be multilayered, which will increase the RE cost for layer-by-layer analysis. The problem is that back-to-back BGA packaging is complex and expensive.

4. If the devices are operating in an unusual fashion (for example, if there are jumbled addresses and data buses), then it would be hard to find the functionality of the device. Obfuscation (for instance, wiring connections between unused pins to unused pins, having spare inputs and outputs from processors to route signals, dynamically jumbling buses, and jumbling the PCB silkscreen annotations) could complicate the RE process. However, such techniques also require the use of more complex chip and complicated design methods.

Many of the above methods are difficult to implement and could significantly increase design and manufacturing costs. Table 2.3 shows the effectiveness of anti-RE techniques at the board-level [49]. A total of five levels are used for scaling based on identifying design cost, manufacturing impact, and reverse engineering (RE) cost.
Table 2.3: Implementation Challenges of Anti-RE Techniques for Board-Level, where Very High = Most and Very Low = Least

<table>
<thead>
<tr>
<th>Anti-RE Techniques</th>
<th>Design Cost</th>
<th>Manufacturing Impact</th>
<th>RE cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tamper-proof fittings such as torx and custom screws shapes</td>
<td>Moderate</td>
<td>Low</td>
<td>Very low</td>
</tr>
<tr>
<td>Fully potting the space around a PCB</td>
<td>Low</td>
<td>Moderate</td>
<td>Low</td>
</tr>
<tr>
<td>Missing silkscreen with minimum passive components</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Custom silicon, and unmarked IC</td>
<td>Low</td>
<td>Moderate</td>
<td>Low</td>
</tr>
<tr>
<td>BGA (ball grid array) devices</td>
<td>Low</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Routing signals for inner layers only</td>
<td>Moderate</td>
<td>High</td>
<td>Moderate</td>
</tr>
<tr>
<td>Multilayer PCB</td>
<td>High</td>
<td>Moderate</td>
<td>Very high</td>
</tr>
<tr>
<td>Using blind and buried vias</td>
<td>Moderate</td>
<td>Very high</td>
<td>Moderate</td>
</tr>
<tr>
<td>Dynamically jumbled buses</td>
<td>Low</td>
<td>Very low</td>
<td>Low</td>
</tr>
<tr>
<td>Route through ASIC</td>
<td>Very high</td>
<td>Moderate</td>
<td>High</td>
</tr>
<tr>
<td>Route through FPGA</td>
<td>Moderate</td>
<td>Moderate</td>
<td>Moderate</td>
</tr>
<tr>
<td>Elimination of JTAG and debug ports</td>
<td>Low</td>
<td>Moderate</td>
<td>Low</td>
</tr>
</tbody>
</table>

2.7 System-level Reverse Engineering (RE)

With chip- and PCB-level RE processes, the purpose is to obtain the netlist of the chip and board in the embedded system, which represents the function and interconnections of the design. To make the design fully functional, the system operation codes and control instructions, which are defined by firmware, should be retrieved, as well. We refer to this as system-level RE.

Parallel to the embedded system design involving ASICs and MCU/DSPs are designs based on FPGAs, whose share of market has been increasing in modern product design. Considering the fact that the hardware functionality and interconnection (referred to as the netlist) are enclosed in the binary configuration file (called the bitstream), the RE process
of FPGA is completely different from the ASIC chip-level RE, which is mainly based on geometrical characteristics of the chip layout (see Section 2.3). In this section, FPGA RE is categorized into the system-level RE, as well, since both the firmware in MCUs, DSPs, etc. and netlist information are stored in the nonvolatile memory (NVM) devices. Note that we primarily focus on the SRAM-based FPGAs in this section due to its largest market share among the reconfigurable hardware devices.

In this section, we will first introduce the storing in these NVM devices, and then describe the RE methods used to extract the firmware/netlist accordingly.

**2.7.1 Firmware/Netlist Information Representation**

Firmware and netlist information can be stored via *read-only memory (ROM)*, *electrically erasable programmable ROM (EEPROM)*, or *Flash memory*. ROM is a type of memory whose binary bits are programmed during the manufacturing process. Currently, ROM is still among the most popular storage media due to its low cost per cell, high density, and fast access speed. From the perspective of ROM physical implementation, ROM devices can be typically classified into four types [10] as shown in Figure 2.13.

![Figure 2.13](image-url)  
(a) (b) (c) (d)

Figure 2.13: Illustrations of (a) active layer programming ROM, (b) contact layer programming ROM, (c) metal layer programming ROM, and (d) implant programming ROM [10].

- **Active layer programming ROM**: The logic state is represented by the presence
or absence of a transistor. As shown in Figure 2.13(a), a transistor is fabricated by simply bridging polysilicon over the diffusion area.

- **Contact layer programming ROM**: A bit is encoded by the presence or absence of a via, which connects the vertical metal bitline with the diffusion area as illustrated in Figure 2.13(b).

- **Metal layer programming ROM**: The binary information is encoded by short-circuiting the transistor or not as shown in Figure 2.13(c).

- **Implant programming ROM**: The different logic state is achieved by different doping levels in the diffusion area (see Figure 2.13(d)). Generally, higher doping levels will raise the on/off voltage threshold, which will disable the transistor.

Compared with ROM, EEPROM provides the users with the capability to reprogram the contents. One bit cell of EEPROM is composed of two transistors—floating gate transistor (FGT) and select transistor (ST). The floating gate transistor is feathered with two stacked gates: a control gate (CG) and a floating gate (FG). The logic state of the bitcell is encoded in the FGT by the presence or absence of electrons stored in the FG. Being isolated electrically, the FG can retain the electrons when powered off. Flash memory has almost the same structure as EEPROM except for the absence of ST, which is irrelevant to the logic state and just allows EEPROM to be byte addressable.

An FPGA bitstream is essentially a vector of bits encoding the netlist information in FPGA, which defines hardware resources usage, interconnection, and initial states at the lowest level of abstraction. The logic blocks will be configured to represent the basic digital circuit primitives, such as combinational logic gates and registers. The connection blocks and switch blocks are configured to be the interconnections between different logic blocks. Other hardware resources, such as I/O buffers, embedded RAM, and multipliers, can be programmed according to different requirements. Therefore, all the information about the netlist can be obtained from the bitstream file.
2.7.2 ROM Reverse Engineering (RE)

To reverse engineer the ROM contents, one can take advantage of modern optical and electron microscopy to observe the binary states of each cell.

- **Active layer programming ROM**: The metal layer and poly layer need to be removed using the delayering approaches discussed in Section 2.3 for they will obscure the active layer underneath. In Figure 2.16(a), the two different states can be visible.

- **Contact layer programming ROM**: It is much easier to reverse engineer this kind of ROM since there is often no need to delayer the metal layer and the poly layer. In the relatively old ROM technology, the contact layer is clearly visible, but, in more modern technologies, some delaying is still needed to expose the contact layer before observation. The presence and absence of contacts are shown in Figure 2.16(b).

- **Metal layer programming ROM**: This type of ROM can be directly observed under a microscope without having to perform any delayering process, as shown in Figure 2.16(c).

- **Implant programming ROM**: This type of ROM is inherently resistant to optical microscopy since different logic states appear identical as in Figure 2.16(d). To
observe the impact of different doping levels, additional dopant-selective crystallographic etch techniques [99] should be utilized to separate the two logic states as in Figure 2.17.

Generally, ROM only provides limited protection against reverse engineering (RE). Among all types of ROM, the metal layer programming ROM offers the worst security due to the fact that the metal layer is easy to obtain with little effort, while the implant programming ROM provides the highest level of protection available.

![Figure 2.16: Optical inspection of (a) active layer programming ROM [13], (b) contact layer programming ROM [14], (c) metal layer programming ROM [15], and (d) implant programming ROM before selective etch [15]](image)

2.7.3 EEPROM/Flash Reverse Engineering (RE)

Since EEPROM and Flash memory have similar structures and the same logic storage mechanism (as discussed above), they often can be reverse engineered by the same procedures. Due to the fact that EEPROM/Flash represents different states by the electrons -
not by the geometric difference, X-Ray technology cannot be used to detect the contents. Further, any attempt to delayer and measure the electrons in the floating gate, such as SEM and TEM, will change the electron distribution, thereby disturbing the contents inside.

For a quite long time, the EEPROM/Flash technology has been regarded as the most robust memory defense against RE. Recently, several methods \cite{48, 47, 16}, though very expensive and requiring specialized equipment, were proposed to extract the contents in EEPROM/Flash correctly. Note that both the below methods occur from the backside of the memory, since traditional frontside delaying and imaging will cause the charges in the floating gate (FG) to vanish \cite{48}.

![Optical inspection of implant programming ROM after selective etch](image)

Figure 2.17: Optical inspection of implant programming ROM after selective etch \cite{15}.

![SKPM scan and SCM scan from the backside of Flash memory](image)

Figure 2.18: (a) SKPM scan and (b) SCM scan from the backside of Flash memory \cite{16}.

**Scanning Kelvin Probe Microscopy (SKPM) Procedure**

The SKPM procedure \cite{100} directly probes the floating gate (FG) potential through the tunnel oxide layer with a thickness of 10nm, which isolates the FG with the transistor channel as illustrated in Figure 2.14(a). So the first step is to remove the silicon from the backside of the memory and leave the tunnel oxide layer undamaged to avoid charging/discharging of the FG. Then, the bit value can be read under the SKPM scan by applying a DC voltage to the probe tip. As shown in Figure 2.18(a), the scanning data from SKPM shows the two-dimensional distributions of potential difference between the tip and the memory cell. The potential difference between the charged FG (associated with ‘0’) and the tip is much higher than that between the uncharged FG (associated with ‘1’) and the tip, which leads to a brighter area for the bit ‘0’ (circled in black in Figure 2.18(a)).
Scanning Capacitance Microscopy (SCM) Procedure

Unlike the SKPM procedure, the SCM procedure will measure the capacitance variations between the tip with the sample in the contact mode and the high-sensitivity capacitance sensor equipped on the SCM [101]. Given the fact that the holes will be coupled in the transistor channel with the existing electrons in the FG, the SCM sensor will detect the logic states via probing the carrier (hole) concentration. Thus, the backside delayering should keep a silicon thickness of 50-300 nm to leave the transistor channel undamaged. Then the bit information can be read as depicted in Figure 2.18(b). The SCM signal shows that the charged FG (associated with ‘0’) has a darker signal (circled in black), which is consistent with high density of holes.

Comparisons between the SKPM procedure and the SCM procedure are summarized in Table 2.4. Note that, with technology scaling, the electrons stored in the FG have been reduced to fewer than 1000 electrons for 90nm-node NAND Flash [16]. In this case, the SKPM procedure can no longer recognize two logic states accurately, while the SCM still performs well.

Table 2.4: Comparison between SKPM and SCM Procedures

<table>
<thead>
<tr>
<th>Property</th>
<th>SKPM Procedure</th>
<th>SCM Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delayering position</td>
<td>Backside</td>
<td>Backside</td>
</tr>
<tr>
<td>Delayering depth</td>
<td>Entire silicon</td>
<td>50–300 nm thickness</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Measured Carriers</td>
<td>Electrons</td>
<td>Holes</td>
</tr>
<tr>
<td>Measured parameter</td>
<td>Potential</td>
<td>Capacitance</td>
</tr>
<tr>
<td>Operation mode</td>
<td>Non-contact</td>
<td>Contact</td>
</tr>
<tr>
<td>Application</td>
<td>All EEPROM and some Flash</td>
<td>All EEPROM and Flash</td>
</tr>
</tbody>
</table>

2.7.4 Reverse Engineering (RE) of FPGAs

FPGA reverse engineering (RE) involves analyzing the configuration bitstream file and transforming the bitstream file into the hardware netlist, which consists of all the components and interconnections at the register transfer level (RTL). To fulfill this goal, hackers need to go through the following steps: get access to the bitstream file from the Flash memory, decrypt the bitstream (if encrypted), and finally build the mapping relationship.
between the bitstream file and the netlist.

**Bitstream Access**

SRAM-based FPGA stores the logic cells states in the SRAM, which cannot retain the data after power loss. Therefore, an external NVM device (typically Flash) is adopted to hold the configuration bitstream file and transfer the bitstream file at system boot-up to initiate the SRAM in FPGA. The separation between the bitstream file and FPGA makes it easy to dump the contents of the bitstream file. By using a logic analyzer, one can easily wire-tap the JTAG data and command lines to capture the communication between the FPGA and Flash memory during startup.

**Bitstream Decryption**

To increase the security level of FPGA, most FPGA manufacturers will encrypt the bitstream file before storing it in the Flash memory with the encryption standards, such as triple Data Encryption Standard (DES) and Advanced Encryption Standard (AES) [102]. Now the wire-tapped encrypted bitstreams will not yield any information for reverse engineering (RE) as long as the cryptographic key remains hidden inside the FPGA.

The bitstream decryption process in FPGA RE depends entirely on the attacker’s ability to discover the key. Typically, the keys are stored in the embedded NVM by programming the FPGA before loading the encrypted bitstream into FPGA. The invasive and destructive attacks to find out the cryptographic key are usually infeasible, since they will trigger tamper detection in the FPGA to zeroize the secret keys. So far, no public report exists on a successful invasive attack towards SRAM-based FPGA.

Recently, it has been reported that the bitstream encryption of several mainstream FPGA series [103] [104] [105] is vulnerable to the side-channel attacks [106]. Basically, a side-channel attack (SCA) is a non-invasive attack to exploit the relationship between physical information (power, timing, and electromagnetic emanation) and certain hardware operations in the FPGA implementation. In [103], the triple DES encrypted bitstream file from Xilinx Virtex-II Pro FPGA has been first successfully cracked by the side-channel attack. The leaked timing and power consumption information is collected when the en-
crypted bitstream is decrypted by the dedicated hardware engine within the FPGA. By analyzing the collected power consumption and timing behavior, the hypothetical structure of the internal triple DES module can be verified. Finally, the divide-and-conquer approach is applied in order to guess and verify a small portion of the key (e.g., 6-bit for triple DES), which reduces the computation’s complexity. This process is repeated until the entire key is obtained. The more recent Xilinx FPGAs (Virtex-4 and Virtex-5), which employ a more advanced encryption module (AES-256), have been cracked in [104] by a more sophisticated type of correlation power analysis [107].

In a similar way, the FPGA power consumption or electro-magnetic radiation (EM) is measured while the decryption block is operating in the FPGA. More recently, the cryptographic keys in the Altera’s Stratix II and Stratix III FPGA families have also been revealed by the same side-channel attack [105]. The fact that all the above attacks can be conducted within several hours reveals the vulnerability of the bitstream encryption.

**Bitstream Reversal**

Prior to converting the bitstream file into the corresponding hardware netlist, one should first understand the bitstream structure which is usually documented by FPGA vendors and is accessible online. Typically, a bitstream file consists of four parts [108]: command header, configuration payload, command footer, and start-up sequence. In the case of Xilinx FPGA, the configuration payload determines the configuration points (LUT – or Lookup Table, memory, register, multiplexer, etc.) and the programmable interconnection points (switch box). The goal of the bitstream reversal is to find out the mapping relationship between the configuration payload with the configuration points and the programmable interconnection points. However, this mapping relationship is proprietary and undocumented, which makes the bitstream file itself serve as an obfuscated design to protect the hardware netlist. In the last decade, there have been several attempts to achieve a bitstream reversal.

**Partial bitstream reversal**: This kind of bitstream reversal only focuses on extracting some specific configurable blocks in FPGA, such as LUT (Lookup Table), CLB (Configurable Logic Block), and multiplier from the bitstream file. [109] shows the possibility to identify the embedded IP-cores by extracting the contents of LUT in Xilinx Virtex-II
**FPGA.**

**Full bitstream reversal:** [110] makes the first public attempt to convert the bitstream file into the netlist. The set-theoretic algorithm and cross-correlation algorithm [110] were used to build a database linking the bitstream bits to the associated resources (configuration points and programmable interconnect points) in FPGA. Then, the database is utilized to produce the desired netlist based on any given bitstream file in Xilinx Virtex-II, Virtex-4 LXT and Virtex-5 LXT FPGAs. This method, however, cannot fully create the netlist because it only relies on the information from the accessible XDL file (Xilinx Design Language) generated from the Xilinx EDA tool, which only provides information on the active configurable resources. The missing information on the static, unused configurable resources in the FPGA places it some distance away from full bitstream reversal. In [111], XDLRC (Xilinx Design Language Report), a more detailed file generated from Xilinx EDA tool, is used to enhance the creation of the mapping database. Unlike XDL, the XDLRC file can offer all of the information available about active and static configurable resources. However, the test results in [111] indicate new issues that the cross-correlation algorithm cannot perfectly relate all the resources in FPGA with the bits in bitstream file. Therefore, the immature technique of the bitstream reversal makes the FPGA embedded system more robust against FPGA reverse engineering (RE) compared with ASIC design MCU designs.

### 2.8 System-level Anti-reverse Engineering

In this section, the solutions to increasing the cost of RE on firmware and FPGA bitstreams are analyzed and discussed.

#### 2.8.1 Anti-reverse Engineering for ROMs

The most effective solution for increasing the complexity and difficulty of RE against ROM is to use the camouflage method. Simply speaking, the designer will make all the memory cells identical under optical inspection, no matter what the contents. This type of solution, though it increases the costs of manufacture, will force the attacker to spend considerably more time, money, and effort to get access to the ROM contents. Recall that,
for the implant programming ROM in Section 2.7.1, the use of different doping levels to
code information constitutes one kind of camouflage technique. Several other camouflage
techniques are provided below.

**Camouflage Contacts**

Different from the contact layer programming ROM (see Figure 2.13(b)), where the absence
or presence of contact will expose the logic states, the camouflage contacts act as false
connections between the metal layer and active layer to make the true contacts and the
false contacts indistinguishable under optical microscopy [112]. To decode the contents,
careful chemical etching has to be applied to find the real contacts, and this is very time
consuming. From the viewpoint of time/cost, this technique will also increase production
periods and lower the manufacturing yield.

**Camouflage Transistors**

To improve the security of active layer programming ROM (see Figure 2.13(a)), false tran-
sistors are made to confuse the RE attempts, instead of using the absence of transis-
tors [113]. The false transistors, essentially with no electrical functions, have the same
top-down view as the true transistors under optical microscope. To crack the information,
the attackers have to use more advanced electrical microscopes to analyze the top view and
even the cross sectional view of the ROM, which is usually economically prohibitive. This
kind of design will definitely increase the difficulty of RE on the large scale, while it only
requires minimal effort during manufacturing.

**Camouflage Nanowires**

Through the use of nano material, ROM cells are fabricated within the vertical connections
between the bit lines and the word lines of a ROM array [114]. The real connections between
bit lines and word lines act as transistors, while the non-electrical dummy connections only
play the role of design camouflage. Due to the small dimensions of the nanowires, the tiny
differences between the dummy connections and real connections are indiscernible even
under advanced electrical microscopy. The biggest challenge with camouflage nanowires,
however, is to manufacture the ROM at a high enough volume and a high enough yield, given the restrictions of our current technology.

Practically, all the above camouflage techniques only need to be adopted on a portion of the whole ROM. To develop a stronger anti-RE ROM, more than one anti-RE technique can be used at once.

**Antifuse One-time Programming**

Admittedly, traditional ROMs are inherently vulnerable to RE procedures. Even ROMs equipped with auxiliary anti-RE designs can only offer limited protection against destructive and invasive RE, while they make the design and fabrication process much more complicated. Currently, ROM replacements (such as antifuse one-time programming (AF-OTP) memory devices) are gaining considerable interest.

The AF-OTP memory exploits whether the gate oxide is in breakdown or is intact to indicate two logic states. Gate oxide breakdown is achieved after fabrication by applying high voltage to the gate of the transistor. Among several proposed structures [115] [116] [17], the split channel 1T transistor antifuse [17] exhibits many advantages over the conventional ROM with respect to cell area, access speed, and immunity to RE. As shown in Figure 2.19(a), the anti-fuse transistor acts like a capacitor when unprogrammed, but a conductive path will be formed once the oxide is ruptured following the programming operation. Due to the angstrom level difference between the programmed and unprogrammed antifuse, existing RE techniques (such as delayering from either frontside or backside, FIB based voltage contrast [117], and top-down view or cross-sectional view from electrical microscopy) won’t expose any information contained, not to mention the fact that it is very difficult to locate the oxide breakdown. Additionally, the anti-fuse memory is compatible with the standard CMOS technology, thus no additional masks or processing steps are required for fabrication. Considering the security, performance, and cost, the anti-fuse memory may eventually replace current ROM devices with the feature size continuously scaling down [116].
2.8.2 Anti-reverse Engineering for EEPROMs/Flashes

To reverse engineer the EEPROM/Flash memory, attackers prefer to delayer from the backside to avoid disturbing the floating charges. Thus, the most effective countermeasure would be to prevent backside attacks. Here we will first briefly introduce some backside attack detection methods, and then we will review one alternative to EEPROM/Flash, which can inherently tolerate the backside attacks.

Circuit Parameter Sensing

Performing the delayering process from the backside will thin the bulk silicon. By burying two parallel plates in the bulk silicon to form a capacitor, the capacitance sensing [118] will detect the capacitance reduction when the attacker polishes from the backside. When the capacitance reaches below a certain threshold, it will trigger the EEPROM/Flash memory to activate an erase operation. The capacitor, perpendicular to the bulk silicon, was previously a challenge to achieve. Fortunately, the emergence of the through-silicon via (TSV) technique [119], makes it much easier to fabricate. Similarly, other parameters, such as resistance [120], can be measured and compared with the pre-defined reference resistance threshold.

Light Sensing

By optically monitoring the backside of the chip, the light sensing method will equip at least one pair of light-emitting and light-sensing devices in the front side of chip and light
reflection module at the bottom of the silicon bulk \[121\]. The light-emitting device is configured to emit light, which can penetrate the bulk, be reflected by the light reflection module, and then be collected by the light-sensing device. Once the delayering is applied, the changes in light distribution at the light-sensing device can trigger the self-destruction of the data contained in the memory. This method can certainly make the RE process more time consuming; however, the costs associated with manufacturing and the power consumption from continuous light emitting and sensing make it less attractive in practice.

It is worth mentioning that for the sensing methods introduced in Section 2.8.2, once the detection signal generated from the above sensing methods is activated, the memory will automatically erase all or part of its contents. This policy, however, will not cause too much trouble for the RE attackers. For example, the attack can either isolate the charge pump, which provides the power to erase, or ground the detection signal by using a Focused Ion Beam (FIB) to eventually render all detection-erasure methods useless. In addition, even if the memory successfully erases all the contents, the attackers still have the chance to determine the actual values according to the residual electrons on the floating gate due to data remanence \[122\].

**FeRAM Memory**

As previously mentioned, the use of electrons on floating gates to represent the logic states makes the EEPROM/Flash memory vulnerable to reverse engineering (RE). Recently, Ferroelectric RAM (FeRAM) has been shown to be a promising candidate for replacing EEPROM/Flash memory. The motive for FeRAM development is to substantially shorten write time as well as lower write power consumption. Recently, it was reported that FeRAM can still possess very strong protections for the contained state \[123\].

Distinct from the EEPROM/Flash storage mechanism, FeRAM stores data by the polarization states of molecules. These kinds of molecules, located at the middle layer of an FeRAM cell, are capacitors filled with a ferroelectric crystalline material, usually a lead-zirconium-titanate (PZT or Pb(ZrTi)O3) compound. As shown in Figure 2.19(b), the two polarization states, simply the shift up/down of Zr/Ti atom in PZT, represent two
different logic states. Due to the high dielectric constant of PZT, the states remain and only flip under the external electric field.

Due to the special state representations, the difference between two states under optical and electrical inspection is invisible. This is because the distance of the shift up/down (see Figure 2.19(b)) is in the scale of nanometer, thereby exposing nothing to the top-down view. One possible attack to reveal the contents, though economically prohibitive, is to carefully slice and analyze the cross-sectional view under SKPM/SCM cell by cell to inspect the difference between the two states.

2.8.3 Anti-reverse Engineering for FPGAs

The fact that the encrypted SRAM FPGA can provide enough reverse engineering (RE) resilience leaves less space for the research and development of anti-RE techniques compared with the ASIC design. Nevertheless, we still categorize the existing FPGA anti-RE techniques into three groups according to the FPGA RE procedure.

Bitstream Hiding

By integrating the bitstream storage memory with FPGA, the Flash FPGA and antifuse FPGA do not require external configuration memory, leaving the direct wire-tapping useless. Unlike the SRAM FPGA, the Flash FPGA does not need bitstream download during power-up due to the Flash memory nonvolatility. The antifuse FPGA has been widely used in military applications because of its higher RE resilience. As we have discussed in Sections 2.7.3 and 2.8.1, an attempt to delayer the Flash memory and antifuse memory - let alone the Flash FPGA and antifuse FPGA - to read out the memory contents is quite challenging and requires specialized equipment. Though these FPGAs require more fabrication steps than SRAM FPGA and lack enough programmability due to limited writing times of the Flash/antifuse memories, they are becoming the dominant choice in critical applications.
Side-channel Resistance

The recent success of side-channel attacks on FPGA prove that the leakage of information poses a large threat to FPGA security. Thus, it is necessary to develop the side-channel resistance designs to protect the cryptographic keys. Intuitively, the most effective side-channel resistance design is to remove the dependency between deciphering operations and power consumption. [125] presents a dynamic and differential CMOS logic implementation which has a constant power consumption and circuit delay irrespective of different circuit operations. [126] proposes to adopt the asynchronous logic design to obtain power consumption independent of computations and data. These methods, while effective against SCA, lead to much larger area and power consumptions compared with the standard CMOS logic.

Another group of side-channel resistance designs can be found in the noise addition group. By introducing random power noise to make the power consumption of decryption non-deterministic, it is quite difficult for the attacker to determine which part of the power consumption is from the decryption. Again, this kind of method will introduce new power consumption. In [127], the power reduction technique is proposed to lower the power consumption overhead from noise generation.

Bitstream Anti-reversal

Until now, full bitstream reversal has only been theoretically possible. As one can imagine, the invasive attacks in the future may successfully find out the entire mapping between the encoding bits from the bitstream file and the hardware resources in the FPGA. FPGA vendors should study potential countermeasures in order to impede bitstream reversal under non-invasive attacks.

Currently, bitstream reversal strongly depends on the amount of publically available information (e.g., user guides) and undocumented information (e.g., files generated by EDA tools). FPGA vendors should take the possibility of reverse engineering (RE) attacks into account when releasing new information in order to hinder potential bitstream reversal attempts.
Another consideration is partial configuration. The critical configuration bits in the bitstream file (such as the IP core) are stored in the Flash memory within the FPGA, while other non-critical parts are still loaded from the external memory. This partial configuration only leaves the wire-tapper partial information about the whole FPGA mapping information, thereby fundamentally eliminating the potential of bitstream reversal.

2.8.4 Summary of Anti-RE Techniques for System-Level
Table 2.5 illustrates the cost and the associated yield loss of the system level anti-RE techniques discussed below. To assess the feasibility of the anti-RE techniques, we roughly classify the costs of RE/anti-RE into five levels based on the previous discussions: very low, low, moderate, high, and very high. It is worth mentioning that the costs of anti-RE techniques mainly consist of the design and manufacturing costs, while the yield loss is estimated from the manufacturing perspective; other factors, such as power, area, and reliability, are not included for lack of open literature. Note also that Table 2.5 only reflects present RE/anti-RE costs. With more effective RE/anti-RE techniques emerging in the future, both RE and anti-RE costs will vary accordingly. In practice, the techniques with lower costs for anti-RE but higher costs for RE in Table 2.5 will be more preferably accepted. For ROM, the best choice is clearly antifuse OTP which has low anti-RE costs but makes RE very challenging. For EEPROM/Flash, the options are limited, but FeRAM appears to be the most promising. Finally, for FPGAs, bitstream hiding stands out as the best candidate.

2.9 Reverse Engineering Summary
It is very hard to defeat reverse engineers in their attacking attempts, but a complex and protective anti-RE system could be devised that would be so time-consuming, difficult, and expensive that it could deter most forms of RE. In the meantime, if the reverse engineering of adversaries is successful, the technology could be superseded by its next-generation version [57]. There are several anti-RE technologies discussed in Sections 2.4, 2.6 and 2.8 but it is worth mentioning that some of these techniques are vulnerable to RE attacks. For
Table 2.5: Costs of Anti-RE Techniques and RE for System-Level, where Very High = Most and Very Low = Least

<table>
<thead>
<tr>
<th>Anti-RE Techniques</th>
<th>Anti-RE Cost</th>
<th>RE Cost</th>
<th>Yield Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROM</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Camouflage contacts</td>
<td>High</td>
<td>Moderate</td>
<td>Low</td>
</tr>
<tr>
<td>Camouflage transistors</td>
<td>Low</td>
<td>High</td>
<td>Moderate</td>
</tr>
<tr>
<td>Camouflage nanowires</td>
<td>High</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Antifuse one-time programming</td>
<td>Low</td>
<td>Very high</td>
<td>Very low</td>
</tr>
<tr>
<td>EEPROM/Flash</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Circuit parameter sensing</td>
<td>Moderate</td>
<td>Low</td>
<td>Moderate</td>
</tr>
<tr>
<td>Light sensing</td>
<td>High</td>
<td>Low</td>
<td>Moderate</td>
</tr>
<tr>
<td>FeRAM memory</td>
<td>Moderate</td>
<td>Very high</td>
<td>Very low</td>
</tr>
<tr>
<td>FPGA</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bitstream hiding</td>
<td>Very low</td>
<td>High</td>
<td>−</td>
</tr>
<tr>
<td>Side-channel resistance</td>
<td>Moderate</td>
<td>High</td>
<td>−</td>
</tr>
<tr>
<td>Bitstream anti-reversal</td>
<td>Low</td>
<td>High</td>
<td>−</td>
</tr>
</tbody>
</table>

example, camouflage tries to make imaging, etc. difficult. Therefore, if someone delayers the chip, board, or system destructively, they could find the full functionality. Obfuscation doesn’t prevent imaging but makes the functionality of the design ambiguous or locked. After conducting a destructive analysis, a reverse engineer could extract high-level netlists so that they could find the functionality [11]. There are other, chip-level techniques like hardware metering, EPIC, and reconfigurable logic barriers that could be used as anti-RE techniques, but most of these methods are used for anti-piracy. Metering, reconfigurable logic barriers, and EPIC have limited uses for anti-RE because, if someone can extract the key by applying backside attacks, RE is trivial. Also, applying anti-RE techniques to the board-level can be very challenging because the board is much more vulnerable to RE due to its simple laminated structure.
Part II

Authentication of PCBs and ICs
Chapter 3

Authentication of Printed Circuit Board

3.1 Introduction

Very few effective PCB detection methodologies have been described in the literature. One method used by Applied DNA Sciences named as ”DNA marking” [128]. Unique and unclonable botanical DNA are used in this method to verify the product. Laser reader could be used for authentication on the supply chain. [129] and [130] are commercial solutions which depend on the dedicated secure chip and this chip need to be integrated into to the PCB. But, those methodologies only can detect the individual chips on the PCB. Hence, this technology is only able to detect the soldered ICs but the PCB can not be verified. Radio Frequency Identification (RFID) technology can also used for verification of electronic products [131]. This method is wireless and radio-frequency electromagnetic fields are used for non-contact transfer of data. Therefore, tracking tags attached to the object could be used for identification of the object. However, RFID can also be cloned easily, meaning that the cloned RFID and authentic RFID are hard to distinguish.

Physical Unclonable Functions (PUF) have emerged as a mechanism for authentication of electronic parts [30, 31]. A PUF uses random manufacturing process variability to extract a unique signature from each production unit. A PUF is a multiple-input multiple-
output function with very hard to predict outputs. A set of outputs are mapped from the PUF’s input and those output responses are described as Challenge-Response Pairs (CRPs) (See Fig. 3.1). It is very hard to predict the responses to a particular challenge because the PUF is derived from the random variation of the process. Therefore, a PUF has excellent resilience for authentication and verification. In addition, PUFs have been used for promising security measures because of its ability to generate unique signatures. For example, PUFs have been used for authentication of the device and generation of ID[30]. Fig. 3.1 shows that system designer authenticates the particular chip for a given challenge and those challenge response pair will be recorded in the system. Then, the response could be verified using CRPs from the trusted data base. A number of existing PUF designs have been proposed included those based on ring oscillators[30], arbiters[33], gate glitches[34], memory arrays[36, 37, 38], scan chains[35], etc.

Furthermore, PUF design has two major categories which could be mentioned as "strong PUFs" and "weak PUFs". Weak PUFs can directly digitize a "fingerprint" of the circuit and those digital signatures could be integrated into cryptographic primitives. The "fingerprint" (CRPs) should be stored in a database at enrollment. The end user can verify the authenticity of the product by checking the CRPs from the database. Therefore, PUFs should be unique, random and reliable[32]. The PUF should have a unique response and could produce a random signature in different environmental conditions.

To date, however, most of these prior PUF designs have been embedded into an integrated circuit die or chip. Therefore, these methods can not be implemented on a PCB. Because the variation sources mentioned due to intrinsic variability with the chip, not a variation from the PCB. Therefore, authentication needs to be proposed which is applicable for PCB level.

Therefore, authentication needs to be derived from manufacturing variations from PCB for security standpoint. Those variations should be able to generate an ID for differentiation between counterfeit and authenticate PCB. Few techniques have been proposed in the literature recently for PCB identification. The trace impedance variations are measured using dedicated testing equipment in [28] and used for counterfeit PCB detection. The main assumption of this authentication method attacks is that PCB is vulnerable to coun-
terfeiting from PCB designers to system designers. This method could be used for effective PCB authentication. But, the methodology is difficult to implement and after deployment counterfeit attack has not considered. Another method has been proposed in [132] which is based on manufacturing variations of capacitive copper patterns of fabricated PCB. However, the complexity of the design make very hard to implement [132]. Therefore, in this work, we propose a novel counterfeit PCB authentication approach that utilizes the RC pair variations on a manufactured PCB to create a unique signature. We have also shown experimental results to show the uniqueness and robustness of our proposed PUF. It is also worth mentioning that our proposed ID generation technique is based on fully digital solution that does not require any analog circuitry to process the input analog signals from the PCB RC circuit.

The rest of our chapter is organized as follows. The proposed methodology for authentication of PCB is shown in Section 2. The experimental results and analysis are provided in Section 3. Finally, in Section 4, we discuss the temperature and supply voltage variations, and security analysis.

### 3.2 Prior Related Work

In past few years, PUFs have attracted a considerable attention, and as a result, several kinds of approaches have been proposed for ID generation, such as SRAM PUF [36] [133], RO-PUF [134], Arbiter PUF, DRAM PUF, Clock PUF, etc. For identification and authen-
tication of ICs, many of these methods have been used which are shown in [30, 135].

In Arbiter PUF, outputs will be determined by a digital race condition. And this will be determined by manufacturing process variations of two paths of the IC. Based on random startup values of the capacitor, the DRAM PUF has been proposed in [38]. Based on delay characters on silicon RO PUF has been proposed and different Ro pair will have a difference in delay. The SRAM PUF relies on the random starting value of because of mismatch of two cross-coupled inverters during power up.

Furthermore, PUF design has two major categories which could be mentioned as "strong PUFs" and "weak PUFs". Weak PUFs could directly digitize "fingerprint" of the circuit and those digital signatures could be integrated into cryptographic primitives. The "fingerprint" (CRPs) should be stored in a database by PUF designer [38]. The end user could verify the authenticity of the product by checking the CRPs from the database. Therefore, PUFs should be unique, random and reliable [32]. The PUF should have a unique response and could produce a random signature in different environmental conditions. Those existing PUF designs are proposed for chip-level authentication and can’t be used for PCB level authentication. Because the variation sources mentioned due to intrinsic variability with the chip, not a variation from the PCB. Therefore, authentication needs to be proposed which is applicable for PCB level.

3.3 Proposed Methodology

Our proposed method is based on the variability of resistor-capacitor (RC) pairs on a PCB. The PCB is populated with a number of RC pairs configured as a low-pass filter as shown in Fig. 3.5. Those RC pairs should be identical, but manufacturing variability causes slight physical variations in each pair. By applying a sinusoidal input with a frequency near the cut-off frequency of the low-pass filter, we should be able to measure variations in the response of the filter because of the variability in the RC pairs. These filter responses can be used to serve as the unique response of the LoPUF.

For reliable ID generation, our proposed method consists of 1) Mux-demux switch 2) Low pass filter and 3) Instrumentation amplifier which is shown in Fig. 3.2. The detail
A multiplexer is an electronic device that selects one of the multiple input signals and the selected input is forwarded into a single line. The selector lines determine which input to be connected to the output. Multiplexers could be used in both analog and digital applications. Multiplexers are classified into different types. An 8-to-1 Multiplexer is shown in Fig. 3.3 which consists of 8 channel input lines, one common output line and 3 selector lines.

In contrast, De-multiplexer is a device with one input and multiple outputs. The demultiplex known as data distributors. A mux takes multiples signals and forwards them into a single line, but a demux does the opposite function what a mux does, this is the main difference between a mux and demux. If Fig. 3.3 is used as a 1-to-8 demultiplexer, the channel in/out terminals are the outputs (A-H), the common out/in terminals is the
input (Z) and selector lines are same (S0-S2).

Therefore, to make the mux-demux switch for our experiment, the output of the mux is connected to the input of a demux. The schematic diagram of the Mux-demux switch is shown in Fig. 3.4. For our experiment, we have used 8 to 1 mux to select the input resistors. Then, the output of the Mux is used as an input of the Demux. The output of the Demux is connected with capacitors. Therefore, that kind of switch configuration, there will be total 64 RC pair combinations possible ($8 \times 8$). In real implementation, 64 RC pairs will be used instead of Mux-Demux switch.

![Mux-demux switch](image1)

**Figure 3.4: Mux-demux switch.**

![Passive low pass filter](image2)

**Figure 3.5: Passive low pass filter.**
3.3.2 Low pass RC filter

If one had an ideal low-pass filter, one could extract a discriminating bit from the filter such that if the signal passes, we get a 1 and if not we get a 0. However, as shown in Fig 3.6, real filters do not have such sharp cutoff features. One could use higher-order filters to achieve sharper cutoffs, but the complexity of these circuits make them impractical to implement. Filters can be categorized into two major types which are active filters and passive filters. Those filters could be classified into the high pass and low pass filter based on different frequency bands they pass. If resistor and capacitor are used, then the filter is called passive filter.

Figure 3.6: Transfer characteristics of Low pass filter(Ideal and Practical).

In our proposed method, we have used a passive low pass filter consisting of a series RC (Resistor-Capacitor) circuit. In this type of passive filter arrangement, an input signal (Vin) is applied to the RC pair and output (Vout) is measured across the capacitor. This type of passive filter categorized as a “first-order filter”. The basic characteristic of a low pass filter is that it will only allow signals from 0 Hz to the cut-off frequency, $\omega_c$ point, but will block any higher frequency. The frequency between passband and stopband is known as cut-off frequency ($\omega_c$). The transfer function for ideal and practical low-pass filter is shown in Fig. 3.6 (K=1 for first order filter). For the RC low-pass filter, the attenuation of the filter as shown in Fig. 3.6 is
\[
\frac{V_{\text{out}}}{V_{\text{in}}} = \frac{1}{\sqrt{(RC\omega)^2 + 1}} \tag{3.1}
\]

The cut-off frequency of the filter defined as the 3dB point or the half-power frequency where the attenuation reaches \( \frac{1}{\sqrt{2}} \). Thus, for the low-pass filter, the cutoff frequency is \( \frac{1}{RC} \) radians/s or \( \frac{1}{2\pi RC} \) Hz.

However, in the case of practical low pass filter passband and stopband could not clearly defined. Therefore, transfer function of the filter \( |H(jw)| \) (where \( |H(jw)| = \frac{V_{\text{out}}}{V_{\text{in}}} \), \( V_o=\text{output} \) and \( V_i=\text{input} \)) changes continuously from its maximum voltage to zero voltage.

Therefore the cut-off frequency of the practical filter defined as \( \frac{1}{\sqrt{2}} \) of maximum value of \( |H(jw)| \).

Using the voltage divider formula in Fig. 2, open circuit output voltage can be found:

\[
V_{\text{out}} = (V_{\text{in}}) \times \frac{(X_c)}{\sqrt{R^2+X_c^2}}, \text{ for absolute voltage}
\]

where, \( X_c = \frac{1}{2\pi fC} \) and the cut-off frequency, \( f_c = \frac{1}{2\pi RC} \).

Now, the phase at the cut-off frequency, \( \phi=-\arctan(\frac{\pi RC}{f_c}) \).

Furthermore, it is possible to cascade RC filters to make higher order filter to get more manufacturing variability for PUF instance. The resultant transfer after cascading (\( |H(jw)| \)) will be simply equal to the product of all transfer functions of those two-port networks.

\[
|H(jw)| = |H_1(jw)| \times |H_2(jw)| \times ...
\]

But, the higher order filter will have higher roll-off near the 3-dB point (See Fig. 3.7). Therefore, 3-dB voltage zone which we are using for variability source will be shrinking. Also, higher order filter will need more RC pair, so area overhead will be a major concern to implement. Therefore, parameter choice is important to implement the design for maximum variability and area overhead.

However, passive filters do not contain amplifying devices to strengthen the signal. At cut-off frequency, passive RC filter has a gain of less than 1 (0.707). But, active filter consists of amplifying devices which could be used to strengthening the signal from the previous stage. Therefore, we have used instrumentation amplifier in the next stage to amplify the signal which is from the previous stage of low pass filter.
3.3.3 Instrumentation Amplifier

An Instrumentation amplifier resembles a differential amplifier and additionally the input stage is buffered by two operational amplifiers. Instrumentation amplifier is used in analog circuit design, sensor signal processing, industry and precision measurement applications because of high CMRR (Common mode rejection ratio), low offset voltage, high input resistance, high gain, and low offer drift due to temperature. The circuit configuration of a most common instrumentation amplifier using op-amp what used for our experiment is shown in Fig. 3.8.

![Figure 3.8: Three op-amp based instrumentation amplifier.](image-url)
In this configuration, op-amps A1 and A2 are input buffers and share the current through feedback resistors R5, Rg, and R6. Here, Rg is a gain resistor.

The overall voltage gain of the instrumentation amplifier can be given by the equation below.

Voltage gain, \( (A_v) = \frac{V_{out}}{V_{dc}-V_f} \) and

\[
V_{out} = (V_{dc} - V_f) \frac{R_2}{R_1} \left(1 + \frac{2 \times R_5}{R_g}\right)
\]  \hspace{1cm} (3.2)

Those resistor ratios (R2/R1=R4/R3) are chosen to increase the CMRR. CMRR is measured by changing the common mode voltage and observing the change in output voltage. Therefore, CMRR could be defined as the differential gain to common mode gain:

CMRR = \( \frac{A_d}{A_{cm}} \)

CMRR sometimes represented in decibels (dB) for easy interpretation and comparison. CMRR is dependent on a few amplifier design factors which are [8]:

- Process variations of designs, for example, resistor matching, transconductances, and leakage currents.
- Output impedance
- And changes with frequency due to current source shunt capacitance

Because of input buffers and after choosing the proper ratio matching of the resistors instrumentation amplifier rejects the common mode voltage, therefore giving a high CMRR.

Note that all RC pairs and on-board IC will add area overhead. Area assesses the impact on the overall area of the LoPUF design and reliability is a measure of the scheme’s authentication. Therefore, the scheme offers a choice between the area and resistance against counterfeiting attacks.
3.4 Results and Analysis using Analog Amplifiers

3.4.1 Experimental Setup

We will describe experimental setup and results for the data collection of the proposed PUF in this section. The RC pairs are mounted on and wired to a protoboard. We have also used an Arduino Uno board for analog to the digital conversion of the data. The Arduino is also programmed to control Mux and demux switch to select the RC pairs. The digitized data is finally collected using the MATLAB.

3.4.2 Experimental Results

For evaluation of our proposed approach, we have used 16 resistors and 16 capacitors. As it mentioned before, the resistor and capacitor are used for the experiment 47KΩ and 47nF respectively. Therefore, for our experiment, we will have total 256 RC pair combinations (16×16). Two CD4051B CMOS single 8-channel analog Mux/Demux have been used to make the switch. As we have 8 channel Mux/Demux chip, we could get 64 (8 × 8) RC pair combinations for one setup. Therefore, we will need total 4 setups to get full 256 RC pair combinations.

Three TL074 low-noise JFET operational amplifiers are used to make proposed instrumentation amplifier. The resistors are chosen in a way that gain of the amplifier is 100. Table 3.1 shows the nominal values of the resistors used for instrumentation amplifier. Therefore, after using the following equation, we get the gain of the amplifier which is:

\[
\text{Amplifier gain} = \frac{R_2}{R_1} \left(1 + \frac{R_5}{R_g}\right) = \frac{22}{2.2} \left(1 + \frac{2 \times 10}{2.2}\right) \approx 100
\]

<table>
<thead>
<tr>
<th>Resistors</th>
<th>Values (KΩ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1=R3</td>
<td>2.2</td>
</tr>
<tr>
<td>R2=R4</td>
<td>22</td>
</tr>
<tr>
<td>R5=R6</td>
<td>10</td>
</tr>
<tr>
<td>Rg</td>
<td>2.2</td>
</tr>
</tbody>
</table>

The instrumentation amplifier has a good stability which is needed for PUF reliability. Furthermore, the high gain is modeled to get more separation between the voltages for the
different RC pairs from the process variation.

Finally, the output of the instrumentation amplifier is connected to Arduino Uno for digitization. And Arduino Uno is connected through the serial port of a computer using the MATLAB to get the data. It is worth mentioning that Arduino can read only voltages between 0 to 5 V with 10-bit quantization. Therefore, we have devised a way so that the differential voltage input \((V_{dc} - V_f)\) will give the lower output peak in this range. The maximum possible sampling rate of Arduino Uno is 9KHz. As the nominal cut-off frequency of the RC pair is 72 Hz, the Arduino Uno could sample the data accurately. Total 500 sample of lower peaks are used for each pair to get the nominal average voltage. And standard deviation are also recorded for every pair. The distribution of the voltage of 256 RC pairs is shown in Fig. 3.9. The mean voltage of the distribution is 2.40 V.

![Figure 3.9: Distribution of the voltages of RC pairs from the experiment.](image)

Then, we have checked the distribution trend of our experimental data. R software is used for that purpose and the data passed kstest and adtest for normal distribution. The cumulative distribution function of the data is shown in Fig. 3.10.

### 3.4.3 Voltage Selection Algorithm for RC Pair

The enrollment process is used to select the bits for unique ID generation of a PUF. To make the key, we have assumed that if the voltages are greater than 2.40 it will be 1 and if less than 2.40 it will be 0. However, a few voltages could be overlapping with mean voltage 2.40 (\(\bar{\mu}\)). The ideal case for RC pair key generation is that the standard deviation
of the voltages is close to zero, but practically we got the standard deviation within tens of millivolt range. We will define error due to overlapping is $\Delta$. We could control $\Delta$ for how much error we could tolerate. Now, we need to define following parameters to find $\Delta$.

$$\alpha_{left} = \int_{-\infty}^{\bar{\mu}} f(x) \, dx$$  (error of bit 1 due to overlapping distribution to left side)

and

$$\alpha_{right} = \int_{\bar{\mu}}^{+\infty} f(x) \, dx$$  (error of bit 0 due to overlapping distribution to right side)

Where, the distribution function, $f(x) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2} \left(\frac{x-\mu_i}{\sigma_i}\right)^2}$

and $\mu_i$ and $\sigma_i$ is mean and standard deviation of each individual voltage respectively ($i = 1$ to $128$).

we have calculated the standard deviation of each voltage and worst case standard deviation is accounted for error formulation. Our worst case standard deviation for an individual voltage was 40 mV and Fig. 3.11 is shown for overlapping for a voltage of 2.4170 V as an Example.

Now, the maximum overlapping area for a single voltage could be defined as $\Delta$ and maximum error due to overlapping,

$$\Delta_{max} = 3 \times \sigma_i$$  (we assumed that $\Delta_{max}$ is same in both side from the mean $\bar{\mu}$)

Therefore, the region of acceptance could be defined as
\[ p_{\text{acceptance}} = 1 - \int_{\mu - \Delta}^{\mu + \Delta} F(x) \, dx \]

Where \( F(x) \) is the distribution of 256 sample voltages. If the voltages are not in the accepted region, it will be discarded. Then, we will need few more RC pair to get the desired ID.

Now, we have used the following integration in this chapter to find the probability of getting the error bits,

\[ p_{\text{error}} = 2 \int_{\mu + \Delta}^{\infty} \int_{-\infty}^{\mu - \Delta} f(y, \mu, \sigma) f(x, y, \sigma_i) \, dx \, dy, \]

where \( \sigma \) and \( \sigma_i \) are standard deviation of 256 sample voltages and each individual voltage respectively.

Let's define \( k \) as the ratio between \( \Delta \) and \( \sigma_i \)

\[ k = \frac{\Delta}{\sigma_i}, \text{ therefore } \Delta = k \sigma_i, \text{ where } k_{\text{max}} = 3. \]

Our approach shows that different \( \Delta \) parameter and constraints will give different acceptance region and therefore different extra RC pairs to get the same desired key value.

If we get \( m \) correct bits out of \( n \) bits, the binomial probability density function for a given pair of parameters \( n \) and \( P_{\text{Acceptance}} \) is

\[ f(m|n, p_{\text{acceptance}}) = \binom{n}{m} p_{\text{acceptance}}^m (1 - p_{\text{acceptance}})^{n-m} I_{(n=0 \text{ to } 128)} \]

where \( q = 1 - p_{\text{acceptance}} \). The resulting function, \( f(m|n, p_{\text{acceptance}}) \), is the probability of observing \( m \) correct bits from \( n \) bits from the acceptance region. The function \( I_{(n=0 \text{ to } 128)} \) only take integer values from 0 to 128. In Table 3.2, we show the total key values needed to get the expected key values for different \( k, \Delta, p_{\text{error}} \) and acceptance region. For lower \( \Delta \), the

Figure 3.11: An example of overlapping of voltage which could create error bit.
accepted key bits are smaller because of wide acceptance region. As delta is increased the
constraints make acceptance region narrower- therefore more bits will be needed. Fig. 3.12
shows accepted key bits when accepted region varies. For larger accepted region (lower $\Delta$),
we will need a relatively small number of accepted key bits with less reliability.

Table 3.2: Total key values for different $k$, $\Delta$, $p_{eb}$ and acceptance regions with Analog
Op-Amp

<table>
<thead>
<tr>
<th>$k$</th>
<th>0.375</th>
<th>0.75</th>
<th>1.5</th>
<th>1.875</th>
<th>2.25</th>
<th>2.625</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta$</td>
<td>0.015</td>
<td>0.03</td>
<td>0.06</td>
<td>0.075</td>
<td>0.09</td>
<td>0.105</td>
<td>0.12</td>
</tr>
<tr>
<td>$p_{eb}$</td>
<td>0.023</td>
<td>0.026</td>
<td>0.034</td>
<td>0.035</td>
<td>0.043</td>
<td>0.048</td>
<td>0.053</td>
</tr>
<tr>
<td>Accepted region</td>
<td>0.988</td>
<td>0.976</td>
<td>0.951</td>
<td>0.94</td>
<td>0.92</td>
<td>0.915</td>
<td>0.903</td>
</tr>
<tr>
<td>Accepted bits (n=36)</td>
<td>36</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>38</td>
<td>39</td>
<td>39</td>
</tr>
<tr>
<td>Accepted bits (n=48)</td>
<td>48</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>51</td>
<td>52</td>
<td>53</td>
</tr>
<tr>
<td>Accepted bits (n=64)</td>
<td>64</td>
<td>65</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>Accepted bits (n=128)</td>
<td>129</td>
<td>131</td>
<td>134</td>
<td>136</td>
<td>138</td>
<td>139</td>
<td>141</td>
</tr>
<tr>
<td>Accepted bits (n=256)</td>
<td>259</td>
<td>262</td>
<td>269</td>
<td>272</td>
<td>276</td>
<td>279</td>
<td>283</td>
</tr>
</tbody>
</table>

Fig. 3.13 shows the acceptance regions with varying $k$. With increasing $k$, acceptance
region decreases which gives less accountable key bits. However, for higher $k$, the con-
straints are stricter which gives more reliable key bits. Therefore, we need to add more RC
pairs for strict constraints to get our desired key. More RC pairs will add more area in the
board. Thus, constraints offer choice between area and reliability.
Figure 3.12: Accepted bits for different acceptance region for key bits (n) of 36, 48, 64, 128 and 256.

Figure 3.13: Accepted region for different k values.

3.5 Digital Extraction of the input pulse from the RC filter

3.5.1 Extraction of PUF bits from the RC filter

In order to use this RC filter, we need to extract the random variations in the R and C values to create usable 1/0 bits to create a unique identifier. A simple approach is to apply
Analog input signal \[ \text{On Board RC pairs} \] CMOS inverter Key

Figure 3.14: Block diagram of proposed LoPUF key generation.

A sinusoidal input to the RC filter, measure the voltage output, and use that output as our unique ID. As R and C change due to the manufacturing variations, the output will also change as well. The problem with this approach is that, assuming that the R and C variations are normally distributed, the output values will show clustering around the nominal value as described by Eq. 3.1. For a true unique identifier, we need the output values to be uniformly distributed. A solution to the clustered distribution problem is to instead generate a single 0/1 bit based on whether the output is greater or lesser than the nominal value. With this approach, we would need multiple RC filters to generate multiple identifier bits. Since the RC pairs can be distributed across the PCB using micro embedded components, the number of RC pairs is not a serious problem.

A more serious problem is that the design as presented requires a number of analog components including a frequency generator to create the input sinusoidal wave and an A/D converter and possible amplifier to measure the output voltage. In many embedded systems designs, adding this extra analog circuitry is not feasible. Thus, our design solution needs to be completely digital (other than the actual RC pairs on the PCB) (See block digram Fig. 3.14).

An inverter is connected to the output of the RC filter for our proposed method. Instead of using an analog input, we use a square wave input that can be easily generated in any digital circuit. After passing the square wave through the RC filter, we end up with a series of exponential responses for each square wave pulse as shown in Fig. 3.15. In these simulations, the input square wave input is 100 Hz, and the R and C values are 47K and 47nF respectively for a cutoff frequency of 72 Hz.

Given this exponential RC filter output, we still need to extract digital bits. As mentioned above, using an A/D to measure the peak voltage of the output is not feasible.
Instead of an A/D, we use an inverter at the output. As the analog output passes through the inverter, it gets converted to a series of digital pulses as shown in Fig. 3.16. The threshold of the inverter is 2.5V. Thus, whenever the RC output is above 2.5V, the digital output is a 0, and 1 otherwise. Variations in R and C cause changes in the length of the 0 pulse Fig. 3.17. As a result, instead of looking at the RC output voltage, we instead look at the length of the 0 pulse (which can be easily measured with a digital counter) as a means to extract the RC variations. Similar to what we discussed before, we can generate 0/1 bits by comparing the output against a nominal value.
3.5.2 Simulation Results

We have performed simulations to check the output voltage distribution of the RC pairs. The resistors and capacitors are chosen for simulation so that it could be also applicable to the experiment. The nominal values of resistor and capacitor are used $47\, \Omega$ and $47\, \text{nF}$ respectively. The cut-off frequency for this arrangement of the filter is $72\, \text{Hz}$. The input voltage is chosen $1\, \text{V}$ for the simulation. The voltage distribution of the low pass filter is shown in Fig. 3.18 respectively using MATLAB simulation. From Fig. 3.18 we can see that the voltage separation between RC pairs is quite small, it will not be differentiable to generate key bits. As we mentioned in the previous section, instead we use an inverter after the output of RC pairs to convert the input signal into fully differentiable digital signal which is discussed elaborately in later sections.

3.6 Results and Analysis using Inverter

3.6.1 Experimental Setup

We will describe experimental setup and results for the data collection of the proposed PUF in this section. The RC pairs are mounted on and wired to a protoboard and an inverter is connected after the RC filter. We have also used an Arduino Uno which is
programmed to measure the pulse width from the output of the inverter. Note that in an actual implementation, this pulse width measurement could also be done with a digital counter circuit.

### 3.6.2 Experimental Results

For evaluation of our proposed approach, we have used 8 resistors and 8 capacitors. As it mentioned before, the resistor and capacitor are used for the experiment 47KΩ and 47nF respectively. Therefore, for our experiment, we will have total 64 RC pair combinations. A CD4069UB CMOS Hex Inverter has been used as the inverter. The transition point of the CD4069UB is 2.5 V. Therefore if the input signal less than 2.5 V the output is high state and vice versa.

Finally, the output of the inverter is connected to the Arduino Uno to measure the pulse width. Note that pulse width is defined by the elapsed time between the rising and falling edges of the single pulse of a signal. A total of 200 samples of the pulse widths for each RC pair are used to get the nominal average value. In addition, the standard deviations are also recorded for every pair. The distribution of the pulse width of 64 RC pairs is shown in Fig. 3.19. The mean pulse width of the distribution is 8113 µs.
We fitted a distribution to our experimental data using the R statistical package \textit{kstest} and \textit{adtest} tools. If \textit{kstest} and \textit{adtest} accept the null hypothesis for our experimental data, then the data can be said to fit a normal distribution. Both tests give us ”not rejected” which indicates that our data accepts the null hypothesis for normal distribution.

### 3.6.3 Pulse width Selection Algorithm for RC Pair

The enrollment process is used to select the bits for unique ID generation of a PUF. To make the key, we have assumed that if the pulse widths are greater than 8113 µs it will be 1 and if less than 8113 µs it will be 0. However, a few pulse widths could be overlapping with mean pulse width 8113 µs (\(\bar{\mu}\)). The ideal case for RC pair key generation is that the standard deviation of the pulse widths is close to zero, but practically we got the standard deviation for the same 200 samples of the pulse width. We will define error due to overlapping is \(\Delta\). We could control \(\Delta\) for how much error we could tolerate. Now, we need to define following parameters to find \(\Delta\).

\[
\alpha_{\text{left}} = \int_{-\infty}^{\bar{\mu}} f(x) \, dx \tag{3.3}
\]

Equation (3.3) is for error of bit 1 due to overlapping distribution to left side.

and

\[
\alpha_{\text{right}} = \int_{\bar{\mu}}^{+\infty} f(x) \, dx \tag{3.4}
\]

Similarly, Equation (3.4) is for error of bit 0 due to overlapping distribution to right side.

Where, the distribution function,

\[
f(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-\frac{1}{2} \left(\frac{x - \mu_i}{\sigma_i}\right)^2} \tag{3.5}
\]

and \(\mu_i\) and \(\sigma_i\) is mean and standard deviation of each individual pulse width respectively (\(i = 1 \text{ to } 64\)).

we have calculated the standard deviation of each pulse width and worst case standard deviation is accounted for error formulation. Our worst case standard deviation for an
individual pulse width was 10 $\mu$s or 0.01 ms and Fig. 3.20 is shown for overlapping for a pulse width of 8120 $\mu$s or 8.12 ms as an Example.

![Figure 3.20: An example of overlapping of pulse width which could create error bit.](image)

Now, the maximum overlapping area for a single pulse width could be defined as $\Delta$ and maximum error due to overlapping,

$$\Delta_{max} = 3 \times \sigma_i$$

(we assumed that $\Delta_{max}$ is same in both side from the mean $\bar{\mu}$)

Therefore, the region of acceptance could be defined as

$$p_{acceptance} = 1 - \int_{\bar{\mu}-\Delta}^{\bar{\mu}+\Delta} F(x) \, dx \quad (3.6)$$

Where $F(x)$ is the distribution of 64 sample pulse widths. If the pulse widths are not in the accepted region, it will be discarded. Then, we will need few more RC pair to get the desired ID.

Now, we have used the following integration in this chapter to find the probability of getting the error bits,

$$p_{eb} = 2 \ast \int_{\bar{\mu}+\Delta}^{+\infty} \int_{-\infty}^{\bar{\mu}} F(y, \bar{\mu}, \bar{\sigma}) f(x, y, \sigma_i) \, dx \, dy, \quad (3.7)$$

where $\bar{\sigma}$ and $\sigma_i$ are standard deviation of 64 pulse widths and each individual pulse width respectively.

Lets define $k$ as the ratio between $\Delta$ and $\sigma_i$
Table 3.3: Total key values for different $\Delta$, $p_{eb}$ and acceptance regions with Inverter

<table>
<thead>
<tr>
<th>$\Delta$</th>
<th>1</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_{eb}$</td>
<td>0.0318</td>
<td>0.0266</td>
<td>0.0209</td>
<td>0.0162</td>
<td>0.0123</td>
<td>0.0092</td>
<td>0.0067</td>
</tr>
<tr>
<td>Accepted region</td>
<td>0.994</td>
<td>0.971</td>
<td>0.944</td>
<td>0.916</td>
<td>0.887</td>
<td>0.860</td>
<td>0.833</td>
</tr>
<tr>
<td>Accepted bits (n=36)</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
<td>41</td>
<td>43</td>
</tr>
<tr>
<td>Accepted bits (n=48)</td>
<td>48</td>
<td>49</td>
<td>50</td>
<td>52</td>
<td>54</td>
<td>55</td>
<td>57</td>
</tr>
<tr>
<td>Accepted bits (n=64)</td>
<td>64</td>
<td>65</td>
<td>67</td>
<td>69</td>
<td>72</td>
<td>74</td>
<td>76</td>
</tr>
<tr>
<td>Accepted bits (n=128)</td>
<td>128</td>
<td>131</td>
<td>135</td>
<td>139</td>
<td>144</td>
<td>148</td>
<td>153</td>
</tr>
<tr>
<td>Accepted bits (n=256)</td>
<td>257</td>
<td>263</td>
<td>271</td>
<td>279</td>
<td>288</td>
<td>297</td>
<td>307</td>
</tr>
</tbody>
</table>

$k = \frac{\Delta}{\sigma_i}$, therefore $\Delta = k \sigma_i$, where $k_{\text{max}} = 3$.

Our approach shows that different $\Delta$ parameter and constraints will give different acceptance region and therefore different extra RC pairs to get the same desired key value.

If we get $m$ correct bits out of $n$ bits, the binomial probability density function for a given pair of parameters $n$ and $P_{\text{Acceptance}}$ is

$$f(m \mid n, p_{\text{acceptance}}) = \binom{n}{m} p_{\text{acceptance}}^m q^{n-m} I_{(n=0 \text{ to } 64)}$$ (3.8)

where $q = 1 - p_{\text{acceptance}}$. The resulting function, $f(m \mid n, p_{\text{acceptance}})$, is the probability of observing $m$ correct bits from $n$ bits from the acceptance region. The function $I_{(n=0 \text{ to } 64)}$ only take integer values from 0 to 64. In Table 3.3, we show the total key values needed to get the expected key values for different $\Delta$, $p_{eb}$ and acceptance region. For lower $\Delta$, the accepted key bits are smaller because of wide acceptance region. As delta is increased the constraints make acceptance region narrower- therefore more bits will be needed. Fig. 3.21 shows accepted key bits when accepted region varies. For larger accepted region (lower $\Delta$), we will need a relatively small number of accepted key bits with less reliability.

Fig. 3.22 shows the acceptance regions with varying $k$. With increasing $k$, acceptance region decreases which gives less accountable key bits. However, for higher $k$, the constraints are stricter which gives more reliable key bits. Therefore, we need to add more RC pairs for strict constraints to get our desired key. More RC pairs will add more area in the board. Thus, constraints offer choice between area and reliability.
Figure 3.21: Accepted bits for different acceptance region for key bits (n) of 36, 48, 64, 128 and 256.

3.7 Discussion

We have completed the evaluation of LoPUF under Different $V_{DD}$ and temperature variations. We have examined the pulse width stability of LoPUF under various supply voltage

Figure 3.22: Accepted region for different k values.
and temperature conditions. We examined the pulse width variation at different supply voltages (4.5 V, 5.5 V) and temperatures (25°C to 85°C). From the experiment, we have found that at higher supply voltage (5.5 V), the pulse width increases and at lower supply voltage, the pulse width decreases. Thus, power supply variation can alter the pulse width of the LoPUF. Therefore, a controlled voltage regulator is needed to regulate the supply voltage at the nominal value (5 V) for LoPUF stability. For the temperature variation, Temptronic TP04100A ThermoStream Thermal Inducting System is used from 25°C to 85°C. The test setup is shown in Fig. 3.23. We have used temperature condition from 25°C to 85°C with 10°C interval, as Thermostream system could deliver controlled temperature precisely. As temperature increases, % variation of the pulse width increases, as shown in Fig. 3.24. Therefore, authentication of the PCB using the RC pair method for the LoPUF must be done at room temperature.

Security measures of the LoPUF are also evaluated against different kind of attacks. We have presented LoPUF based on resistor and capacitor variation of low pass filter for system and board level authentication. In the literature, machine learning attacks have been
reported for strong PUFs [138]. As our proposed design is a weak PUF, it is not vulnerable to machine learning attacks. Another vulnerability of the design is that the resistor and capacitor are used as discrete components that could be easily replaced/removed. While this might disable authentication, it will not allow an attacker to fake authentication. Nevertheless, to address this issue, we could use "buried" capacitors as used in BoardPUF [132]. Even if we use the BoardPUF buried capacitors, the main advantage of our method is the area efficiency. The BoardPUF uses a Schmitt trigger and comparator circuit to create the PUF ID, which can take up a large amount of circuitry compared to the minimal inverters that we use.

3.8 Conclusion

It is very important for a company to prevent unrecoverable losses due to counterfeiting. In this chapter, we have presented a PUF based authentication technique to prevent counterfeiting. LoPUF generate key bits which are reliable and unique and those key bits are generated utilizing the manufacturing variations of resistors and capacitors. An algorithm also described to find reliable bits for enrollment. This algorithm is applied to different key values and there is a trade off between reliable key values and area overhead. Although our proposed LoPUF has many advantages, we have to consider many other challenges, for
example, area overhead, power, environmental and supply voltage variations, cost etc. In future work, we will fabricate the board to investigate the performance in real situations.
Chapter 4

Authentication of IC based on RC variations

4.1 Introduction

In this chapter, we propose a novel IC authentication method that utilizes internal resistor and capacitor pair variations during the manufacturing process of an IC to create a unique signature. We have shown the simulation result using a 45 nm technology node in Cadence to show the uniqueness and robustness of our proposed PUF. It is worth mentioning that our proposed ID generation method is fully digital and doesn’t require any analog circuitry for processing input analog signals of the ICs (See Fig. 4.1). The remaining of our chapter is organized as follows. In section 4.2 we have shown our proposed methodology for IC authentication. The simulation results using Cadence Virtuoso 45 nm technology node have been provided in section 4.3. Finally, the conclusion and discussion are described in section 4.4.

Figure 4.1: Block diagram of proposed LoPUF key generation.
4.2 Proposed Methodology

We have proposed our method based on the variability of resistor-capacitor (RC) pairs on an IC. The internal circuit of the IC will be populated with a number of RC pairs that are configured as a low-pass filter as shown in Fig. 4.2. The resistor-capacitor pairs should be identical, however, variability because of manufacturing will cause slight physical variations in each pair. If a sinusoidal input is applied a frequency that is near the cut-off frequency of the low-pass filter, we will be able to measure variations in the output response of the RC filter because of manufacturing variability in the RC pairs. The responses from those filters then could be used as a unique identifier of the low pass filter PUF (LoPUF).

4.2.1 Low pass RC filter

One could extract the discriminating bit from the filter when the signal passes through the ideal low-pass filter, we get a 1 and if not, we consider as 0. However, real filters don’t have such sharp cutoff features that are shown in Fig 4.3. To achieve sharper cutoffs, the designer could use higher-order filters, but those circuits impractical to implement because of their complexity.

![Passive low pass filter](image)

Figure 4.2: Passive low pass filter.

We have considered a passive low pass filter consisting of a series RC (Resistor-Capacitor) circuit in our proposed method that is shown in Fig. 4.2. In this type of passive RC filter circuit arrangement, an input signal (Vin) is applied to the RC pair to measure the output...
(Vout) across the capacitor. The main characteristic of a low pass filter – it will only pass input signals from 0 Hz to the cut-off frequency which could be defined as \( \omega_c \) point but will block any higher frequency signals. In Fig. 4.3 the transfer function for the ideal and practical low-pass filter is depicted (for first-order filter, \( K=1 \)). The attenuation function of the RC low-pass filter that is shown in Fig. 4.3 is

\[
\frac{V_{\text{out}}}{V_{\text{in}}} = \frac{1}{\sqrt{(RC\omega)^2 + 1}}
\]  

(4.1)

The half-power frequency where the attenuation reaches \( \frac{1}{\sqrt{2}} \) defined as cut-off frequency of the filter or the 3dB point. Thus, the cutoff frequency is \( \frac{1}{RC} \) radians/s or \( \frac{1}{2\pi RC} \) Hz for the low-pass filter.

4.2.2 Extraction of PUF bits from the RC filter

Since we don’t have an ideal filter, we need a way to extract the random manufacturing variations in the R and C values in order to use this RC filter to generate usable 1/0 bits for a unique identifier. One possible method is to apply a sinusoidal voltage input to the RC filter, measure the voltage output and use those RC output values to make the unique ID. The problem with this approach is that if the R and C variations are normally distributed, the output of the RC values will show clustering around the nominal value which is described by Equation 4.1 However, we need the output values to be uniformly distributed so that they can be used as a truly random unique identifier. This clustering
can be seen in Fig 4.4. Using MATLAB, we have simulated the output voltage of the RC filter using nominal values of 10 KΩ for the resistor and 120 pF for the capacitor with a 10% variation (total 10,000 RC pairs). The filter has a cut-off frequency of 132 KHz for this arrangement and the peak input voltage of the sine wave for the filter is set to 1 V for the simulation. randn MATLAB function is also used in this simulation that returns a random scalar drawn from the standard normal distribution of the resistor and capacitor. Note also, as can be observed in Fig. 4.4, the output voltage separation between RC pairs is quite small and hence, those voltages cannot be easily differentiable to generate ID bits.

Figure 4.4: Output voltage distribution of low pass filters from the MATLAB simulation.

A solution to this clustered distribution problem is to generate a single 0 or 1 based on whether the output value of the filter is greater or lesser than the nominal expected voltage (In Fig 4.4 0.707V). Therefore, we would require multiple RC filters to generate multiple identifier ID bits with this approach. Requiring multiple RC pairs is not a serious problem, since they can be easily distributed inside the chip.

However, a more serious problem with that approach is that we need a number of analog components such as a frequency generator to create the input sinusoidal wave and an analog comparator with a possible signal amplifier to compare the output voltage with the nominal value. Adding those extra analog circuitries is not feasible to place inside
in many IC designs. Hence, other than the actual resistor-capacitor pairs, the proposed solution needs to be completely digital.

To start, instead of using an analog sinusoidal input, we use a square wave input that can be easily generated in any digital circuit. If the square wave is passed through the RC filter, we get a series of exponential responses for each square wave input pulse as shown in Fig. 4.6. In these Cadence simulations, the input square wave input is 1.2V, 4 $\mu$s or 250 kHz, and the nominal resistor and capacitor values are 10 K$\Omega$ and 120 pF respectively for a cutoff frequency of 132 kHz.

![Figure 4.5: Schematic diagram of our proposed method with the inverter.](image)

We still need to extract digital bits from this exponential RC filter output. An analog-to-digital converter to measure the peak voltage is not feasible. Instead, we use an inverter at the output. When the output of the RC filter passes through the inverter, the signal gets converted to a series of digital pulses as shown in Fig. 4.5. The inverter essentially acts as a comparator comparing the filter output voltage to the switching voltage - normally $V_{DD}/2$.

In Fig. 4.6, $V_{DD}/2 = 0.6V$. Variations in R and C will vary the rise time of the exponential filter outputs. That translates into variations of the width of the 0-pulse. In order to capture these variations, we measure the width of the 0-pulse using a digital counter. The widths themselves will have a normal distribution. Instead, we generate 0/1 bits, similar to what we discussed before, by comparing the 0-pulse width against a nominal value.

### 4.3 Results and Analysis

In this section, we will discuss simulation results of the proposed PUF using Cadence Virtuoso for the 45 nm technology node. The proposed method is implemented in Cadence
and pulse width measurement is done with a digital counter circuit. The schematic of the implemented circuit is shown in Fig. 4.7. When the output of the inverter goes high (i.e. as a result of an exponential pulse from the filter), the multiplexer selects a high-speed clock that activates the counter. In this case, we have a 12-bit counter comprised of 12 D-flip-flops and half-adders. The number of bits in the counter define the resolution of the measurement of the pulse width. When the output of the inverter goes low, the counter is stopped, and the resulting count is a measure of the pulse width of the inverter output.

The area of this arrangement consists of 1 RC pair and 744 transistors for the inverter, multiplexer, logic gates, half adders and flip-flops. The largest component of the transistor count is due to the 12 flip-flops and 12 half-adders which have a total of 480 transistors.
and 216 transistors respectively. However, in a PUF where we may want to generate $n$ ID bits, the counter, mux, and stopping flip-flop can be shared between the $n$ RC pairs at the cost that bits will be generated serially rather than in parallel. Since the PUF ID generation is usually a one-time operation, this is not a significant cost. Thus, the total transistor count for an $n$-bit LoPUF is

$$742 + 4(n - 1) + 2 \log n + 2n \quad (4.2)$$

where the middle two terms count the transistors in an $n : 1$ pass-transistor based multiplexor, and the last term is the number of transistors in the $n$ digitizing inverters. If $n=128$, using Eq. 4.2 the number of transistors is 1520, or 11.875 transistors per bit. Using a rough estimate of 100 $F^2$ per transistor for a typical layout, the total area due to the digital logic is 152000 $F^2$ or 1187.5 $F^2$ per bit. Using measurements from [139], the sheet resistance for a P+ poly based resistor is $522 \ \Omega \ \text{sq}$ for a 40 nm technology node. Thus, a minimum sized 10 KΩ resistor would have an area of $0.03 \ \mu m^2$ ($19.2 \ F^2$, W=40 nm, L=760 nm). Using measured values of the vertical overlap capacitance between the bottom of poly to the substrate (99.11 [$aF/\mu m^2$]), a 120 pF capacitor would have an area of over 1 mm$^2$. This is clearly infeasible. A more practical implementation of a large IC capacitor is to use free space in the metal layers which would not take any area on the silicon substrate [140, 141, 142]. Thus, we do not include the capacitor size in our area calculations. Adding the transistor area plus the resistor area, we arrive at 1206.7 $F^2$ per bit. A comparison with other analog compatible PUFs is shown in Table 4.1. Because the counter and mux are shared, the area overhead for those transistors is less compared to other proposed methods in literature.

<table>
<thead>
<tr>
<th>Technology</th>
<th>[143]</th>
<th>[144]</th>
<th>[145]</th>
<th>[146]</th>
<th>Proposed method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area/bit ($F^2$)</td>
<td>2.2×10$^6$</td>
<td>727</td>
<td>6000</td>
<td>6500</td>
<td>1207</td>
</tr>
</tbody>
</table>

Table 4.1: State of the art PUF area overhead comparison with our proposed method
4.3.1 Experimental Results

For evaluation of our proposed approach, we have used Verilog-A code to model a resistor and capacitor pair with manufacturing variances. As mentioned before, the resistors and capacitors for the simulation have nominal values of 10 KΩ and 120 pF respectively. In our simulation, we have used a total of 100 RC pair combinations using the Verilog-A model to provide variations in capacitor and resistor values with a normal distribution. The output of the RC filter goes through the inverter, and the transition point of the inverter is 0.6 V. Therefore, if the input signal of the inverter is more than 0.6 V the output is a low state and vice versa.

Finally, the inverter output is connected to the counter. The clock of the counter is set to 1.05 GHz, and with a 12-bit counter, we can measure pulses up to 3.89 µs long with a resolution of 0.95 ns. For example, if a pulse width of the output signal of the inverter is 2.1 µs, the equivalent counter value would be 2.1 µs / 0.95 ns = 2210. We have used equivalent counter values to show our final distribution of the pulse width.

The pulse widths (in counter values) were recorded for 100 RC pairs using the Cadence simulation and the distribution of the pulse widths is shown in Fig. 4.8. The mean pulse width of the distribution is 2077 (1.97 µs). We call this overall mean $\bar{\mu}$.

![Histogram](image)

Figure 4.8: Distribution of the pulse widths of RC pairs from the experiment.
The resultant data was fitted to a distribution using the R statistical package `kstest` and `adtest` tools. The data could be said to fit a normal distribution if `kstest` and `adtest` accept the null hypothesis of our simulated data. The test results have shown us “not rejected” which indicates that our data has accepted the null hypothesis for normal distribution. Since the distribution is normal, we can use a similar approach as described earlier to select output bits for the PUF. If the pulse width is greater than the mean (1.97 µs), the PUF output will be a 1 and it will be a 0 if not. To generate a multi-bit output (for example for a key or unique ID), we will require multiple RC pairs.

4.3.2 Pulse Width Selection Algorithm for RC Pair

As stated before, if the pulse width for a particular RC pair is greater than the mean, the PUF output will be defined as a 1 and if less than the mean it will be 0. This assumes that for a particular RC pair, it will always create a pulse with a consistent width. In other words, when we measure the pulse width, it will always be the same, thus generating a consistent 1 or 0. However, the measurement of the pulse width for a single RC pair may have measurement error that itself has a normal distribution. Fig. 4.9 shows the overall pulse width distribution superimposed with the individual pulse width distributions for each measurement. Ideally, the standard deviation of these individual measurement distributions is zero, but in practice, the standard deviation is non-zero though fairly small as compared to the distribution of the measurements from different RC pairs. As a result, RC pairs that have pulse widths that are near the overall mean, \( \bar{\mu} \), could be measured to have a pulse width on the other side of \( \bar{\mu} \) because of the error. In other words, the error distribution of a single pulse width measurement could overlap with \( \bar{\mu} \). Fig. 4.10 shows an example for a measurement distribution for RC pair \( i \), where \( \mu_i = 2085 \), and part of the distribution overlaps the nominal 0/1 cutoff at \( \mu = 2077 \). The shaded area under the curve is the probability that we get the wrong bit for RC pair \( i \).

To minimize error, we define a parameter \( \Delta \) such that no RC pairs are selected that are within \( \Delta \) counts of \( \bar{\mu} \). A large \( \Delta \) produces less error but causes fewer RC pairs to be selected. In order to help arrive at an optimal \( \Delta \) to select RC pairs that will be used to generate valid PUF bits, we build on the probabilistic model defined in our prior work [147].
The analysis is similar to the IOMBA bit allocation methodology presented in [148].

We start by calculating the probability that we get an error because the measured pulse width for RC pair $i$ is less than $\bar{\mu}$ when the expected pulse width for RC pair $i$ should be greater than $\bar{\mu}$. In other words, it measures the area under the distribution curve that is to the left of $\bar{\mu}$.

\[
\alpha_{left} = \int_{-\infty}^{\bar{\mu}} f(x, \mu_i, \sigma_i) \, dx
\]  

(4.3)

$f$ is the normal distribution function for the measurements of each individual RC pair $i$

\[
f(x, \mu_i, \sigma_i) = \frac{1}{\sqrt{2\pi} \sigma_i} e^{-\frac{1}{2} \left( \frac{x-\mu_i}{\sigma_i} \right)^2}
\]  

(4.4)

where $\mu_i$ and $\sigma_i$ are the mean and standard deviation of each individual pulse width measurement respectively.

Likewise, Equation [4.5] is the error probability due to an overlapping distribution on the right.
Figure 4.10: An example of overlapping of pulse width distribution which could create an error bit.

\[ \alpha_{right} = \int_{\bar{\mu}}^{+\infty} f(x, \mu_i, \sigma_i) \, dx \]  

(4.5)

If we integrate over the full RC pair distribution, we arrive at the overall error bit probability as follows:

\[ p_{eb} = \int_{\bar{\mu}+\Delta}^{+\infty} \int_{-\infty}^{\bar{\mu}} f(x, y, \sigma_i) f(y, \bar{\mu}, \bar{\sigma}) \, dx \, dy + \int_{-\infty}^{\bar{\mu}} \int_{\bar{\mu}+\Delta}^{+\infty} f(x, \mu_i, \sigma_i) f(y, \bar{\mu}, \bar{\sigma}) \, dx \, dy \]

\[ p_{eb} = 2 \int_{\bar{\mu}+\Delta}^{+\infty} \int_{-\infty}^{\bar{\mu}} f(x, y, \sigma_i) f(y, \bar{\mu}, \bar{\sigma}) \, dx \, dy \]  

(4.6)

where \( \bar{\sigma} \) and \( \sigma_i \) are the standard deviation of the full population of pulse widths and each individual pulse width respectively (For our sample set of simulation experiments, \( \bar{\sigma} = 40.19 \)).

As we mentioned above, we only accept RC pairs with nominal pulse widths that are more than \( \Delta \) from \( \bar{\mu} \). Therefore, the probability that an RC pair will be accepted is
\[ p_{\text{acceptance}} = 1 - \int_{\bar{\mu} - \Delta}^{\bar{\mu} + \Delta} f(y, \bar{\mu}, \bar{\sigma}) \, dy \quad (4.7) \]

If a RC pair’s pulse width is not in the accepted region, it will be discarded. As a result, we will need more RC pairs to get a desired number of PUF output bits. If we have \( n \) RC pairs, we can estimate the probability of accepting at least \( m \) PUF output bits as follows:

\[ P_{nm} = \sum_{i=m}^{n} \binom{n}{i} p_{\text{acceptance}}^i q^{n-i} \quad (4.8) \]

where \( q = 1 - p_{\text{acceptance}} \). \( P_{nm} \) is the probability that at least \( m \) bits will be accepted from \( n \) RC pairs.

For simplicity, we can assume that the standard deviations for all measurements are equal - i.e. \( \sigma = \sigma_i \forall i \). Moreover, let us define \( \Delta \) as some multiple \( k \) of \( \sigma \) - i.e. \( \Delta = k\sigma \).

Fig. 4.11 shows the acceptance probability with varying \( k \). With increasing \( k \), the acceptance probability decreases which gives less acceptable PUF bits. However, for higher \( k \), the constraints are stricter which gives more reliable PUF bits. Therefore, we need to add more RC pairs for strict constraints to get our desired number of PUF output bits. More RC pairs will require more area in the IC. Thus, constraints offer choice between area and reliability.

In Table 4.2, Table 4.3, and Table 4.4 we show the bit error probability and acceptance probability for different \( \Delta \). The \( \sigma_i \) used in Table 4.2, Table 4.3, and Table 4.4 are 0.3, 1 and 3 respectively. We vary \( k \) from 1 to 3 to 5. As \( \Delta \) increases, the \( p_{\text{eb}} \) values improve but the acceptance probability decreases. For a desired \( m \) of 128 - i.e. we want 128 accepted RC pairs, we show the probability that we will accept at least 128 RC pairs for varying \( n \). As can be seen, we can generally greater than 99.9% likelihood of at least 128 accepted RC pairs if \( n \) is at least 256. If a lower \( p_{\text{eb}} \) is acceptable, \( n \) can be reduced accordingly. As a result, there is a tradeoff between error probability and the overall chip taken by \( n \) RC pairs.
4.3.3 Evaluation of LoPUF Under $V_{DD}$ and Temperature Variations

We examined the pulse width stability at different supply voltages (1.0 V, 1.4 V) and temperature variations from ($-50^{\circ}C$ to $100^{\circ}C$). From our simulations, we found that the pulse width increases at higher supply voltages and the pulse width decreases at lower supply voltages (See Table 4.5). The absolute value of the % variation of pulse width for different supply voltages is shown in Fig. 4.12. Hence, an unstable power supply could vary the nominal pulse width of the LoPUF. Thus, a controlled voltage regulator may be needed to stabilize the power supply voltage at a nominal value (1.2 V) for LoPUF stability. Alternatively, the $\Delta$ term could be increased to account for voltage instability.

Table 4.2: Probability of accepting 128 pairs for different $\Delta$, where $\sigma_i=0.3$

<table>
<thead>
<tr>
<th>$\Delta$</th>
<th>0.3</th>
<th>0.9</th>
<th>1.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_{eb}$</td>
<td>0.0569</td>
<td>0.0543</td>
<td>0.0517</td>
</tr>
<tr>
<td>$P_{acceptance}$</td>
<td>0.994</td>
<td>0.982</td>
<td>0.970</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=128, m=128$)</td>
<td>0.462</td>
<td>0.098</td>
<td>0.020</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=136, m=128$)</td>
<td>1.000</td>
<td>0.999</td>
<td>0.978</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=144, m=128$)</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
</tbody>
</table>
Table 4.3: Probability of accepting 128 pairs for different $\Delta$, where $\sigma_i=1$

<table>
<thead>
<tr>
<th>$\Delta$</th>
<th>1</th>
<th>3</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_{eb}$</td>
<td>0.0549</td>
<td>0.0433</td>
<td>0.0320</td>
</tr>
<tr>
<td>$p_{acceptance}$</td>
<td>0.980</td>
<td>0.940</td>
<td>0.900</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=128, m=128$)</td>
<td>0.075</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=136, m=128$)</td>
<td>0.998</td>
<td>0.570</td>
<td>0.065</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=144, m=128$)</td>
<td>1.000</td>
<td>0.994</td>
<td>0.728</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=160, m=128$)</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
</tbody>
</table>

Table 4.4: Probability of accepting 128 pairs for different $\Delta$, where $\sigma_i=3$

<table>
<thead>
<tr>
<th>$\Delta$</th>
<th>3</th>
<th>9</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_{eb}$</td>
<td>0.0459</td>
<td>0.0282</td>
<td>0.0091</td>
</tr>
<tr>
<td>$p_{acceptance}$</td>
<td>0.940</td>
<td>0.822</td>
<td>0.707</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=128, m=128$)</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=136, m=128$)</td>
<td>0.065</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=144, m=128$)</td>
<td>0.728</td>
<td>0.019</td>
<td>0.000</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=160, m=128$)</td>
<td>1.000</td>
<td>0.799</td>
<td>0.005</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=192, m=128$)</td>
<td>1.000</td>
<td>1.000</td>
<td>0.903</td>
</tr>
<tr>
<td>$P_{nm}$ ($n=256, m=128$)</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
</tbody>
</table>

For temperature variations, we used the Cadence temperature function to vary the temperature from $-50^\circ C$ to $100^\circ C$ (See Table 4.6). As temperature increases or decreases, the variation of the pulse width increases, as shown in Fig. 4.13. If the temperature is too high (more than $100^\circ C$) or too low (less than $-50^\circ C$), % variation increases. Note, however, that unlike with the voltage changes, the variations due to temperature are less than 1%, within the expected standard deviations for measurements, and as a result, should be handled with our $\Delta$ acceptance region.

Table 4.5: % variation versus supply voltage for the RC pair of the LoPUF

<table>
<thead>
<tr>
<th>Voltage (V)</th>
<th>% change</th>
<th>Nominal</th>
<th>% change</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>1.1</td>
<td>1.15</td>
</tr>
<tr>
<td>% Variation</td>
<td>-14.57</td>
<td>-7.16</td>
<td>-3.62</td>
</tr>
</tbody>
</table>
Figure 4.12: Absolute value of the % variation for different supply voltages of the proposed low pass filter method.

Table 4.6: % variation versus temperature for the RC pair of the LoPUF

<table>
<thead>
<tr>
<th>Temp (°C)</th>
<th>% change of pulse width</th>
<th>Nominal</th>
<th>% change of pulse width</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-50</td>
<td>-10</td>
<td>0</td>
</tr>
<tr>
<td>% Variation</td>
<td>0.15</td>
<td>0.10</td>
<td>0</td>
</tr>
</tbody>
</table>

4.3.4 Uniqueness

We have evaluated the uniqueness of the low pass filter PUF. The uniqueness of a PUF defined by its ability to uniquely generate random responses. The uniqueness of a PUF circuit is evaluated from a population of PUF circuits and it depends on several factors such as process variations of a particular manufacturing process as well as defects in the manufacturing systems and the evaluation metrics for the uniqueness. From our 100 random RC pair LoPUF responses, 51% of the responses are 0 and 49% of the responses are 1, which is close to the ideal 50%.
4.3.5 Randomness

The randomness of the PUF responses is very important for PUF design because it could prevent the prediction of the data bits from the attackers. In other words, perfectly random PUF responses mean that the PUF bits of RC pairs are independent of each other and the bit value of the LoPUF cannot be predicted. The intrinsic randomness of the PUF is very attractive to the designers because those PUF circuits could be included in security design without the need for any modifications to the manufacturing process. It is worth mentioning that the uniqueness measurement discussed above which is 50%, it does not necessarily mean the PUF response data is random. To measure the unpredictability or randomness of PUF responses, statistical tests such as NIST randomness test, Min-entropy, Shannon entropy or Machine Learning (ML) techniques could be used to check the PUF data bits. Here, we have used a randomness test as a metric using MATLAB to find the unpredictability (randomness) of our LoPUF data. The MATLAB function returns a test decision from \( h = \text{runstest}(x) \) for a null hypothesis that the data \( x \) of the PUF come in random order, against the alternative decision that they do not random. If this test rejects the null hypothesis at the 5% significance level, the result of \( h \) will be 1, otherwise,
it will be 0. The returned value $h = 0$ from our PUF data indicates that the statistical randomness test doesn’t reject the null hypothesis at the default 5% significance level and therefore the LoPUF data are random.

### 4.3.6 Security measures of the LoPUF against Attacks

In this paper, we have proposed a low pass filter PUF based on RC pair variations for chip-level authentication. Machine learning attacks have been shown in the literature against strong PUFs [138]. However, our proposed LoPUF is a weak PUF, therefore the proposed method is not vulnerable to machine learning attacks. Another attack could be performed using destructive reverse engineering to remove/replace the resistor and capacitor sources of the IC. However, this destructive analysis will be very costly to buy state-of-the-art equipment and perform reverse engineering. If this kind of attack is successful, it could possibly disable authentication for only one chip, but the attack would not allow the rogue attacker to falsely authenticate a chip.

### 4.4 Conclusion

Physical unclonable functions (PUFs) have been proposed in recent years as a mechanism to generate unique fingerprints for integrated circuits (IC). These PUFs can be used to identify authentic chips as well as generate keys that are unique to each part. In this paper, we present a simple circuit structure based on a low pass filter that can be used as a weak PUF for use in anti-counterfeit and key generation applications. Our simulations have shown that the generated key of the LoPUF is unique and random and reliable under most conditions. The PUF properties are dependent on the manufacturing variations of resistors and capacitors. The simulation results using Cadence shows that it will be a practical and viable method for IC authentication. Even though our proposed LoPUF is viable solution which has a lot of advantages, we may have to consider other challenges such as power consumption, area overhead, and supply voltage and supply voltage variations, and cost, etc. We would like to fabricate the chip in the future to check the performance analysis of the LoPUF in real-world situations.
Part III

Hardware Obfuscation
Chapter 5

Key generation for Hardware
Obfuscation using Strong PUF

5.1 Introduction

Security has become a major concern for integrated circuits (ICs) because of globalization and outsourced offshore production. Throughout the lifecycle of the IC, the chip could be under attack from overproduction during the manufacturing stage to unauthorized recycling after customer disposition [27]. Furthermore, a manufacturer could make extra chips by cloning during the manufacturing stage after obtaining the design by reverse engineering [149, 24]. State of the art IC reverse engineering is so advanced that the chips could be reverse engineered within a few weeks. There are dedicated companies that do reverse engineering of new industrial chips. Therefore, a scheme is required to prevent the production of illegal cloned chips by reverse engineering.

Several conceptually new and interesting approaches have been proposed to introduce forms of obfuscation or logic locking to prevent reverse engineering and piracy. For example, additional exclusive OR (XOR) gates can be inserted in combinational designs with configurable bits that need to set correctly to activate the chips [41]. Dummy states can be inserted into finite state machines in sequential designs, requiring certain sequences of inputs to be applied for the circuits to function correctly [150]. Scrambling the interconnec-
A problem with many of these approaches is that the key is not unique across all instances of the chip. Thus, if an attacker is able to retrieve the key by some method, it can unlock all chips and effectively overproduce the chips. Recently, physically unclonable functions (PUFs) have been used as a way to provide unique keys for obfuscation. PUFs are inherent circuit primitives that extract randomness from the physical characteristics of a system at the manufacturing stage by applying input challenges and observing random output responses. PUFs are easy and simple to implement but their random nature ideally makes its behavior hard to predict and model for an attacker.

Some of these PUF-based obfuscation approaches require a full characterization of the PUF input-output response, thereby necessitating a relatively small output space for the PUF. Such PUFs are called weak PUFs. An untrusted foundry could do the same characterization and could store the challenge-response pairs (CRP) for all chips before sending them to the design house. If a key leaked from an activated chip, the untrusted foundry could recover the entire design with the leaked key and, using its PUF characterizations, be able to unlock all chips whether authorized or not.

Strong PUFs, on the other hand, have a very large input/output space, making characterization impractical and thus much more secure. However, at the same time, it makes some PUF-based obfuscations approaches infeasible. It would be ideal to have an obfuscation scheme that can take advantage of these strong PUFs to improve the security of the approach.

In this work, we have proposed using a strong PUF with a subset of bits used to generate a key that can unlock a locked circuit. A mathematical probability model is developed which determines the feasibility of the PUF key.

In Figure 5.1, the design and fabrication flow is shown for a prevention method of
piracy and overproduction of chips using a PUF-based method. The designer uses logic synthesis, technology mapping, and place and route tools to produce a graphic database system II (GDSII) file that can be sent to the fabrication facility. Fabricated chips will be manufactured and tested at the foundry and will be sent back to the design house for activation. The design house will activate the chip using the PUF-based key and will send to the market for sale.

The rest of this chapter is organized as follows. First, we describe prior related work. Second, we provide in detail a description of the proposed PUF-based hardware obfuscation method. Next, we describe attack analysis for our proposed method. After that we analyze our technique based on performance analysis. And finally, we evaluate our technique on different ISCAS 85 benchmarks.

![Figure 5.1: Design and fabrication flow of the physical unclonable functions (PUF)-based obfuscation.](image)

### 5.2 Prior PUF-Based Obfuscation Approaches

A PUF is an unclonable hardware function based on process variation during fabrication. A PUF has a set of challenge response pairs (CRP) which are the input and output respectively. The CRPs represent a random function which is unique to each chip. It is impossible to make identical PUFs with the same CRPs [154]. PUFs have been used in different applications including secret key generation, device authentication, and anti-counterfeiting [147]. Below, we describe prior weak PUF and strong PUF-based approaches to prevent reverse engineering and piracy.

Wendt et al. [44] created a unique key per chip using a weak PUF. A part of the circuit’s functionality was replaced with a PUF and a lookup table (LUT). The functionality of the PUF and the LUT together was equivalent to the original circuit functionality. The
content of the LUT for each chip was determined by the designer after obtaining the PUF’s CRPs. Thus, the LUT behaved as a key that would be uniquely configured by the designer in the post-fabrication activation process. A problem with this method was that an untrusted manufacturer could have access to the PUF during the fabrication process. The manufacturer could obtain the CRPs of the weak PUF easily and store those results. If the attacker could find a leaked LUT key of an activated chip in the aftermarket, in combination with the previously stored PUF functionality of the leaked chip, it was trivial to then determine LUT keys for any device.

Khalegi et al. [43] used a similar approach but with a strong PUF with a large CRP space. In order to make the use of a strong PUF feasible, the designer’s characterization of the PUF was limited to just a subset of the input set. For example, if the input of the PUF is $p$ bits long, then only $n$ bits were used for characterization by the designer, where $n << p$. However, an attacker would have to still characterize the entire $p$-bit input PUF. An obfuscator circuit was used to select the $n$ input bits. The added complexity of an obfuscator circuit and selection bits added to the overhead. Moreover, the approach does not scale well because as $n$ grows, the complexity of the LUT grows exponentially. Thus, one cannot obfuscate large circuits, but only small subsections of the circuit. Furthermore, another limitation of this approach is that it does not account for collision of PUF outputs—i.e., with small $n$ it is likely that different inputs could produce colliding PUF outputs, which means it would be impossible to realize a LUT that will satisfy the design functionality. To overcome those limitations, we similarly propose a hardware obfuscation method that does not require the full characterization of a strong PUF and has significantly less overhead.

Alkabani et al. [155] used a strong PUF (or RUB as it is called in the article) to generate bits that could then be transformed into a code to unlock the FSM-based locking scheme. Specifically, a specific input external key was applied as a challenge to the PUF response and XOR’d with a separate external key to generate the appropriate internal key that unlocks the FSM. Note that we refer to “internal key” as the internal bits that unlock the locked circuit, while “external key” is the set of bits that are applied externally by a user or from stored memory. Our approach also uses a strong PUF to generate responses based on
an input challenge. However, the distinction is that we allow the PUF response to directly create the internal key (or sub-key). In other words, we apply multiple challenges where each response forms part of the internal key. The external key is the set of challenges that are used to create the internal key. This simplifies the circuitry since the PUF output does not need to be as large as the internal key and moreover removes the need for any XOR circuitry.

5.3 Strong PUF-Based Key Generation

The main idea of the proposed methodology is to eliminate the use of a LUT for obfuscation and use one of the many existing key-based hardware obfuscation methods [156]. These methods have much lower overheads than LUTs for large \( n \). The PUF-based obfuscation architecture is shown in Figure 5.2. The key will be directly generated from a strong PUF given the appropriate challenge. The key can then be provided to any key-based obfuscation method. We assume a PUF with \( p \) inputs and \( n \) outputs where \( p >> n \). One can use an obfuscator circuit to limit the input set, but as we will show, that is not necessary for large \( n \). Large \( n \) can be problematic in terms of characterization, but for our approach a full characterization is not necessary.

Assume that the PUF generates an \( n \)-bit response given a \( p \)-bit challenge. As the designer, instead of characterizing the entire \( p \)-bit input space of the PUF, we only need to find one PUF input that matches the desired key of the locked circuit. Assuming a truly random PUF, on average, that will take \( 2^n \) brute force tries, rather than the \( 2^p \) challenges needed to characterize the entire PUF. As \( n \) gets large, however, even that can become difficult. Therefore, instead of using a \( n \)-bit PUF, we use only \( n' \) bits of the PUF output, where \( n' < n \). In order to generate the key, we issue \( S = \frac{n}{n'} \) challenges, where each response is stored in a \( n' \)-bit sub-key register. The \( S \) sub-keys together form the \( n \)-bit key \( K \):

\[
K = PUF(C_1)|PUF(C_2)|...|PUF(C_S) \quad (5.1)
\]

An attacker would need to apply \( S \) \( p \)-bit challenges in the correct order to find the right key to unlock the device—in other words they would need to guess \( 2^{Sp} \) challenges.
The flow diagram for PUF-based obfuscation is shown in Figure 5.3. Our goal is to generate a specific obfuscation key (K) from a set of challenge bits (C) that is unique to a specific chip as shown in Figure 5.3a. Internally, the key generation block consists of a PUF and sub-key registers as described earlier (Figure 5.3b). The sub-keys which will be used for the final key which will be configured by the designer during the post-fabrication activation shown in Figure 5.3c. To configure the sub-keys, the designer obtains the CRPs from the PUF during the activation process using the characterization channels Figure 5.3c. After characterization, the designer should remove the channels (Figure 5.3d) so that the PUF cannot be probed after market. Techniques to remove the characterization channels include laser burning the accessing wires and burning the supporting fuses, amongst other approaches.

![Figure 5.2: Overview of PUF-based logic encryption.](image)

Figure 5.2: Overview of PUF-based logic encryption.
Figure 5.3: Design flow for PUF-based obfuscation. (a) The key is generated for a part of a design using \( p \) inputs and \( n \) outputs. (b) The key generation process uses a strong PUF to create sub-keys which comprise the master key. This chip will be only functional when the designer activates it. (c) The designer configures the sub-keys based on challenge-response pairs (CRPs) from the PUF and the final key from the sub-keys. (d) The activated chip is in the open market.

### 5.3.1 Challenge Selection Process

Before a chip can be activated, the designer must select the challenges required to generate the PUF responses/sub-keys that can be combined to create the obfuscation key. As mentioned above, we have divided the \( n \)-bit key into \( S \ n' \)-bit sub-keys as shown in more detail in Figure 5.4. The sub-keys will be visible to the designer and manufacturer but not to the end user. In order to find a challenge that generates a particular sub-key, we need to try, on average, \( 2^{n'} \) challenges. Since we have to find \( S \) sub-keys, we need to find \( S \) challenges (assuming no repetitions in the sub-keys).
How many challenges do we need to try to find these $S$ sub-keys? Or, put another way, given that we will attempt $N$ challenges, what is the probability we find the $S$ sub-keys? The $N$ $p$-bit challenges generate $N$ $n'$-bit responses. Assume that $N - i$ of the responses do not contain any of the required $S$ sub-keys, or in other words, $i$ responses contain all of the $S$ sub-keys and only the $S$ sub-keys (with repetition). There are $\binom{N}{i}$ possible positions for those $i$ responses. Those $i$ responses can be partitioned into $S$ buckets in $\left\{ \frac{i}{S} \right\}$ $S!$ ways, where $\left\{ \frac{i}{S} \right\}$ is the Stirling number of the second kind:

$$\left\{ \frac{i}{S} \right\} = \frac{1}{S!} \sum_{j=0}^{S} (-1)^{S-j} \binom{S}{j} j^{i}. \tag{5.2}$$

The remaining $N - i$ responses have $(2^{n'} - S)^{N-i}$ possibilities. Given that there are in total $(2^{n'})^N$ possible responses to the $N$ challenges, we can put it all together to arrive at the probability that we find the $S$ sub-keys:

$$\sum_{i=S}^{N} \binom{N}{i} \left\{ \frac{i}{S} \right\} S! (2^{n'} - S)^{N-i} \left(2^{n'}\right)^N. \tag{5.3}$$

Using a similar analysis to the coupon collector problem, the expected value for $N$ can be calculated as:
\[ E[N] = \sum_{j=1}^{S} \frac{2^{n'}}{j}. \] (5.4)

For example, when \( n = 128 \), \( n' = 8 \), and \( N = 4000 \), the probability that you will get all 16 sub-keys is 0.999997. The expected number of challenges we need to get the 16 sub-keys is 865. When \( n = 256 \) and \( n' = 16 \), the expected number of challenges to find the 16 sub-keys we need is 221,559—not an unreasonable number. As \( n' \) grows, the expected value grows exponentially. For \( n = 256 \) and \( n' = 32 \), the expected value is \( 11.6 \times 10^9 \). At 10 ns per challenge, it would take roughly two minutes to find the required sub-keys. A larger \( n' \) provides security against attackers who have advanced probing capabilities and have visibility to the PUF responses and can thus limit their challenge tries to \( S2^{n'} \) rather than \( 2^{Sp} \) for those that don’t. However, if that is not a concern, a smaller \( n' \) is more practical.

### 5.3.2 Chip Activation

Chip activation could be accomplished in a number of ways. In one approach, once the challenges have been selected by the designer, they can be programmed into one-time-programmable ROMs and fused in permanently. This allows the designer to control the activation process. Fuses, however, can be reverse engineered, so care must be taken to have these sub-keys programmed in such a way that cannot be recovered. Alternatively, end-user customers can activate the chip directly with activation challenges provided by the designer. These challenges would be unique to each chip, but since the challenges are not permanently programmed into the chip, they would need to be reapplied on every powerup. This approach allows users to effectively disable a chip until some trusted hardware or software has enabled the chip with the appropriate challenges. There may be scenarios where chips are shipped directly from manufacturer to customer. In this case, the PUF characterization channel will need to be left open. The customer could then apply several challenges and get the associated responses for the key. These could be sent to the designer, who can select the specific challenges that will generate the activation key. After activation, an automated mechanism can disable the characterization channel so that those channels can not be used in future. This approach opens up more attack possibilities—for example,
end-users can now use machine learning attacks to characterize the PUF, and with known “good” responses, they can possibly recover the key. Therefore, this last activation method should be used only with trusted customers.

5.4 Security Analysis

5.4.1 Manufacturing Attacks

An untrusted manufacturer has access to the PUF input/output (I/O) channels, and could find the challenges required to generate the sub-keys in the same way as the designer and not require all $2^{Sp}$ challenges. The manufacturer can then unlock overproduced chips assuming that it can get access to a leaked obfuscation key. If the manufacturer has access to the locked netlist, Boolean satisfiability problem (SAT)-based attacks could be used to recover the sub-keys. Once the key has been retrieved, the manufacturer could then unlock chips, or even modify the design to remove the PUF completely and apply the retrieved key directly. Therefore, the underlying obfuscation method must be SAT-resistant. Since our proposed technique will work with any key-based obfuscation technique, as more promising anti-SAT obfuscation technologies are developed [156], our technique will work as well.

5.4.2 In the Field Attacks

We assume, for the purposes of this discussion, that the in-the-field chips no longer have the PUF characterization channels open. As mentioned before, the activation method that leaves PUF characterization channels open to the end-user is only appropriate with fully trusted customers. Unlike other key-based hardware obfuscation approaches, in our mechanism, the raw key is never applied directly to the circuit. A set of challenges will only unlock a specific chip and will not reveal any information about the underlying obfuscation key. Thus, the chances of a leaked obfuscation key are greatly reduced. An attacker must try $2^{Sp}$ challenges to guess the key and those set of challenges, if guessed correctly, would only work on that particular chip, so brute force attacks are not practical.

It is possible that advanced probing attacks could reopen the PUF characterization channels and allow an attacker to generate sub-keys easily. As mentioned above, this only
works if the underlying obfuscation key is known. Even if the key is known, the set of challenges works only on one chip and it is not practical to reopen the PUF channels on all chips and perform the attack at scale. One could potentially use machine-learning attacks that can fully characterize the PUF with a mathematical model [158, 159]. Although these attacks can be used to model a large variety of strong PUFs with high accuracy and precision in a reasonable time, there are a number of optimizations that have been proposed that increase the modeling time significantly such that it is impractical for an attacker to perform these attacks at scale. It should be noted that the same probing techniques could retrieve the key from LUT-based schemes as well.

5.5 Performance Analysis

5.5.1 Hardware Complexity

For our obfuscation key generation, we will need a strong PUF with $p$ inputs and $n'$ outputs. Most strong PUF implementations have $O(p + n')$ complexity. Other than the PUF, the primary costs to our obfuscation key generation are the sub-key registers and some simple control logic to load the sub-key registers appropriately. We need $n$ flip-flops for the sub-keys and control logic is roughly $O(\log S)$. Most key-based obfuscation schemes are $O(n)$ with respect to overhead. Compared to a LUT-based scheme, which is $O(2^n)$ in size complexity, our key generation plus obfuscation has significantly less overhead.

5.5.2 Uniqueness and Reliability of the PUF

It is possible that some input challenges could produce unstable output for the PUF at different operating conditions such as temperature variations, and voltage fluctuations. Therefore, to improve the reliability, error correcting codes (ECC) could be used at the cost of some additional hardware. Alternatively, we could use different challenges that have been evaluated to provide more reliable responses. Since there are multiple challenges that can produce the same response, we could apply several of these redundant challenges to improve the reliability and eliminate the ECC. Note that we have proposed a smaller subset of responses which doesn’t reduce the entropy of the PUF as the attacker need to
find the final $n$-bit key after the chip activation.

### 5.6 Evaluation

We have evaluated the effectiveness of our proposed technique by implementing it on an fine-weld programmable gate arrays (FPGA) and calculating the hardware overhead. Note that in this chapter, we do not report the overhead of the PUF implementation on the FPGA. The overhead of PUF implementations on FPGAs are well reported \[160, 161, 162\], and our approach does not depend on any specific strong PUF implementation. Furthermore, if a PUF is already used in a circuit for hardware security/authentication, the same PUF could be used for obfuscation key generation as well. We have performed the experiments on a number of ISCAS85-based obfuscation benchmarks \[159\]. Again, our approach will work with any key-based obfuscation method. For the purposes of these experiments, we have used logic cone size-based obfuscation \[163\] along with the key generated from the strong PUF. In the logic cone size-based obfuscation method, the locked logic is placed in the largest logic cones so that it can impact more signals. A weighted normalized metric for all gates is measured and compared to find the position of the largest logic cone in the module under consideration. Both the fanin and fanout cones of a gate are considered as the metric. Therefore, gates with a higher value of this metric will have higher fanin and fanout and will be chosen as a locking gate in the inputs.

The benchmarks have been synthesized using Xilinx Vivado. Table 5.1 shows the characteristics of the ISCAS85 benchmarks under evaluation and the FPGA utilization as reported by Vivado. We have illustrated comprehensive area overhead evaluations on the ISCAS85 benchmarks suite in Table 5.2. The benchmark circuit name is shown in the first column. The second column demonstrates the different key sizes used for obfuscation. Columns 3 and 4 show the area overhead in terms of LUTs due to obfuscation and the overhead that key generation adds to obfuscation after implementation on FPGA. Overheads in percentages have been shown in the last two columns. The area overhead in terms of LUT utilization of the selected benchmarks after obfuscation is shown in Figure 5.5. Figure 5.5 shows that for the larger circuits (C7552) we can achieve the same level of
obfuscation as compared to a smaller circuit (C880) with a lower level of hardware overhead. Also, if the key size is increased, the area overhead is increased for the same circuit because of additional key inputs. Therefore, constraints offer a choice between area and security. Figure 5.6 shows the area overhead for the key generation compared to the base obfuscation method [163] with different key sizes and benchmark circuits. It is noted that the area overhead for the key generation is typically less than 4% for different benchmarks and key sizes.

![Area Overhead Chart](chart)

Figure 5.5: Overheads for key generation compared with base obfuscation.

Table 5.1: ISCAS 85 circuit characteristics

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Inputs</th>
<th>Outputs</th>
<th>LUTs</th>
<th>Slices</th>
</tr>
</thead>
<tbody>
<tr>
<td>C880</td>
<td>60</td>
<td>26</td>
<td>66</td>
<td>25</td>
</tr>
<tr>
<td>C3540</td>
<td>50</td>
<td>22</td>
<td>249</td>
<td>67</td>
</tr>
<tr>
<td>C5315</td>
<td>178</td>
<td>123</td>
<td>267</td>
<td>89</td>
</tr>
<tr>
<td>C7552</td>
<td>207</td>
<td>108</td>
<td>291</td>
<td>93</td>
</tr>
</tbody>
</table>
Table 5.2: Area overhead for obfuscation with key generation for different benchmarks suite with varying key size

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Key size</th>
<th>Obfuscated area (LUT)</th>
<th>Obfuscated area with key gen (LUT)</th>
<th>Area OH (%)</th>
<th>OH (%) for key gen</th>
</tr>
</thead>
<tbody>
<tr>
<td>C880</td>
<td>32</td>
<td>84</td>
<td>85</td>
<td>27.27</td>
<td>1.19</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>94</td>
<td>103</td>
<td>42.42</td>
<td>9.57</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>105</td>
<td>109</td>
<td>59.09</td>
<td>3.81</td>
</tr>
<tr>
<td></td>
<td>256</td>
<td>138</td>
<td>142</td>
<td>109.09</td>
<td>2.90</td>
</tr>
<tr>
<td>C3540</td>
<td>32</td>
<td>255</td>
<td>256</td>
<td>2.41</td>
<td>0.39</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>257</td>
<td>261</td>
<td>3.21</td>
<td>1.56</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>309</td>
<td>316</td>
<td>24.10</td>
<td>2.27</td>
</tr>
<tr>
<td></td>
<td>256</td>
<td>343</td>
<td>356</td>
<td>37.75</td>
<td>3.79</td>
</tr>
<tr>
<td>C5315</td>
<td>32</td>
<td>290</td>
<td>296</td>
<td>8.61</td>
<td>2.07</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>307</td>
<td>311</td>
<td>14.98</td>
<td>1.30</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>342</td>
<td>351</td>
<td>28.09</td>
<td>2.63</td>
</tr>
<tr>
<td></td>
<td>256</td>
<td>452</td>
<td>457</td>
<td>69.09</td>
<td>1.11</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>316</td>
<td>320</td>
<td>8.59</td>
<td>1.27</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>347</td>
<td>348</td>
<td>19.24</td>
<td>0.29</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>359</td>
<td>370</td>
<td>23.37</td>
<td>3.06</td>
</tr>
<tr>
<td></td>
<td>256</td>
<td>463</td>
<td>468</td>
<td>59.11</td>
<td>1.08</td>
</tr>
</tbody>
</table>

Figure 5.6: Overheads for key generation compared with base obfuscation.

Depending on the internal design and architecture, a PUF can take multiple clock cycles to evaluate the response. In Table 5.3 we show the impact of the evaluation time on the overhead. As the clock cycles increase, the number of registers increase logarithmically
primarily to keep timing state. We also see the difference in utilization for a 16-bit PUF \((n' = 16)\) compared to a 32-bit PUF \((n' = 32)\). Consider the 1-clock cycle 64-bit key case. For a 16-bit PUF, we need 67 registers total—64 registers for the key, two registers to keep track of the four sub-keys, and one more register for control logic state. As we increase \(n'\), the key generation cost goes down slightly because \(S\) goes down. However, it comes at the expense of a larger PUF and more effort to find the challenges (Equation (5.4)).

### 5.7 Conclusions

We have proposed a strong PUF based obfuscation method so that the functionality of the logic is completely hidden. The approach is evaluated on a number of different ISCAS 85 benchmark circuits. Compared to LUT-based obfuscation, our approach uses significantly less hardware overhead to provide obfuscation of much larger circuit blocks with similar security guarantees. Compared to key-based logic locking obfuscation, our PUF methodology provides several advantages for very little overhead. Since the key is never applied directly, it provides significant protection against key leakage. In addition, by using a PUF, the activation sequence for each chip is unique, limiting the chance that a key can be leaked. We have also presented a mathematical model to help estimate the effort required to find the sub-keys to make a final key for obfuscation. This effort calculation can be used to choose an appropriate \(n'\) and \(S\) value based on security needs. Overall, our proposed method is a viable solution against IC piracy/overproduction through the use of strong PUF based key generation for obfuscation.
Table 5.3: Area (LUT) and slice registers for different PUF bits for different clock cycles.

<table>
<thead>
<tr>
<th>Key size (PUF delay)</th>
<th>4-bit PUF</th>
<th>16-bit PUF</th>
<th>32-bit PUF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>#LUTs</td>
<td>#registers</td>
<td>#LUTs</td>
</tr>
<tr>
<td>32 bit (1 clock cycle)</td>
<td>3</td>
<td>36</td>
<td>1</td>
</tr>
<tr>
<td>64 bit (1 clock cycle)</td>
<td>4</td>
<td>69</td>
<td>2</td>
</tr>
<tr>
<td>128 bit (1 clock cycle)</td>
<td>6</td>
<td>134</td>
<td>2</td>
</tr>
<tr>
<td>256 bit (1 clock cycle)</td>
<td>7</td>
<td>263</td>
<td>4</td>
</tr>
<tr>
<td>32 bit (2 clock cycle)</td>
<td>4</td>
<td>37</td>
<td>2</td>
</tr>
<tr>
<td>64 bit (2 clock cycle)</td>
<td>5</td>
<td>70</td>
<td>3</td>
</tr>
<tr>
<td>128 bit (2 clock cycle)</td>
<td>7</td>
<td>135</td>
<td>3</td>
</tr>
<tr>
<td>256 bit (2 clock cycle)</td>
<td>8</td>
<td>264</td>
<td>5</td>
</tr>
<tr>
<td>32 bit (4 clock cycle)</td>
<td>4</td>
<td>38</td>
<td>2</td>
</tr>
<tr>
<td>64 bit (4 clock cycle)</td>
<td>5</td>
<td>71</td>
<td>3</td>
</tr>
<tr>
<td>128 bit (4 clock cycle)</td>
<td>7</td>
<td>136</td>
<td>3</td>
</tr>
<tr>
<td>256 bit (4 clock cycle)</td>
<td>8</td>
<td>265</td>
<td>5</td>
</tr>
<tr>
<td>32 bit (8 clock cycle)</td>
<td>5</td>
<td>39</td>
<td>3</td>
</tr>
<tr>
<td>64 bit (8 clock cycle)</td>
<td>6</td>
<td>72</td>
<td>4</td>
</tr>
<tr>
<td>128 bit (8 clock cycle)</td>
<td>8</td>
<td>137</td>
<td>4</td>
</tr>
<tr>
<td>256 bit (8 clock cycle)</td>
<td>9</td>
<td>266</td>
<td>6</td>
</tr>
</tbody>
</table>
Part IV

Reliable Session Key Generation
Chapter 6

Session Key generation using strong PUF modeling

6.1 Introduction

Networks of consumer electronics and embedded systems are involved in most parts of our daily life. Therefore, it is essential that these systems ensure data confidentiality, integrity and privacy as well as provide mechanisms to trust device identity and authenticity. The authenticity of the electronic devices is an important factor because it confirms and strengthens the safety of the home electronics network and gives access to the user to control the network securely. However, most consumer home electronics services rarely have mechanisms for client-side authentication such as certificates or public keys. As a result, communications could be attacked in a number of ways - e.g. man-in-the-middle attack, denial of service attacks, static key leakage, etc. Thus, the credibility and authenticity during communications need to be guaranteed by the consumer devices which are connected to the home network. Without strict client authentication, an unauthorized device could try to access a consumer electronics network with sensors, alarms, smartwatch, IoT, etc. for malicious purposes. For example, in a smart home system, a malicious smoke sensor could insert false messages into a home security control network and set off the alarm and automated activation of water from a sprinkler system. Furthermore, in addition to
authentication, secure and confidential communication between devices is crucial to ensure privacy. Security and privacy vulnerabilities issues from these devices could be severe as an attacker could collect sensitive data and information from the network system [164, 165].

![Figure 6.1](image.png)

Figure 6.1: Examples of consumer electronic (CE) products which are classified as (a) home CE, (b) medical CE and (c) personal CE.

Consumer electronic networks have found their way into a wide range of applications domains [164]. Example devices in these application areas include medical (pacemakers, heart-rate monitors, and insulin pumps), home (sensor, alarm, Google Home, smart thermostat), and personal (smartphones, smartwatch, portable music players, tablets, and laptops). (See Fig 6.1). In this chapter, we have focused on consumer electronics networks that need authenticated communication with a central server. Fig. 6.2 shows a consumer home security network with the proposed authentication framework. The server manages the consumer devices - for example, a home automation controller managing sensors, alarms, cellular phones, smart TVs, IoT, etc. Those devices need to communicate with the server with secure encryption.

6.1.1 Transport Layer Security (TLS)

An obvious choice to provide security for consumer electronics and embedded systems communications is the Transport Layer Security (TLS) protocol which forms the backbone of most secure Internet communications. TLS is a standard encryption protocol that provides privacy and security between communicating users during web browsing, email delivery, instant messaging, and voice over IP (VoIP) [166]. TLS guarantees data confidentiality and authenticity between the server and client during communication by ensuring that no third party or eavesdropper can eavesdrop on or fake a message. The TLS protocol starts with a handshaking procedure by using an asymmetric cipher to establish the secure
connection. Public-key based certificates are used for the authentication of the server and the communicating client. Without authentication, an unauthorized client could easily impersonating a legitimate client. During the initial handshaking, TLS establishes a symmetric encryption key for the communication session using public keys or Diffie-Hellman key exchange. After this initial handshaking, the communication is encrypted with the session specific symmetric key. However, if the attacker is able to steal this session key, they could extract the data and potentially all prior data communications. The connection will continue communication with the current key for encryption and decryption until the session ends.

Figure 6.2: A platform of home electronics communication network with authentication framework.

Although TLS enables computer systems to establish secure connections with session keys, TLS is not practical for embedded systems. Low-end processors in most embedded systems do not have the computation capabilities to efficiently implement the public key cryptography necessary for TLS authentication or the Diffie-Hellman key exchange pro-
tocol. Moreover, even if public key cryptography was computationally feasible, typically only server-side authentication can be implemented. Client-side authentication requires the deployment of private keys in the embedded system, making the system vulnerable to memory probing that can recover these stored private keys [16, 47, 48, 167, 24]. If the key is static and shared among all devices, breaking the key will allow fake devices to present themselves as authentic. Moreover, if the key is used to establish the session key, a broken key could expose all past and future communication. Therefore, recent versions of the TLS protocol have recommended the use of ephemeral keys to ensure forward secrecy. To communicate with each other, a session specific symmetric key will ensure the encryption of all messages for only a particular time frame. Furthermore, forward secrecy will protect past communications or session against future attacks of the secret key. If a unique session key is generated for each user or client, one compromising session will not impact other sessions data communication other than for this session which was encrypted by that specific key.

6.1.2 Physical Unclonable Functions

One possible mechanism to generate private keys without storing the keys is to use physical unclonable functions (PUFs), a promising security primitive that can be used for secret key storage, identification, authentication and cryptography [30, 168, 161, 169] (See Fig. 6.3). PUFs depend on manufacturing variations to present a unique fingerprint (or set of fingerprints) in response to a set of challenge inputs. PUFs can be categorized as strong PUFs or weak PUFs (also called Physical Obfuscated Keys or POKs [170]). The distinguishing feature is that strong PUFs have a large enough challenge space such that enumerating all the challenges is infeasible while weak PUFs or POKs have just one possible challenge or limited set of challenges. Strong PUFs are typically used for authentication, while POKs can be used to generate private keys. However, current weak PUF based key generation methods are static which means the keys do not change over time, and thus is not suitable as an ephemeral session key, and the limited challenge/response pair space restricts its authentication capability. In this work, we propose a unique session key generation method for embedded devices by creating a strong PUF circuit model that can be evaluated by the server [171]. Furthermore, the client or device can be authenticated as
well because the client/device specific session keys will work only for specific PUF devices which have been parameterized and have a saved machine learning model in the server site.

![PUF based device](image)

Figure 6.3: Secure key generation of devices using physical unclonable functions (PUFs).

### 6.2 Proposed Approach

As mentioned above, POKs have been used for key generation but can not be used to generate ephemeral session keys. In this chapter, we propose a session key generation method using strong PUFs without the requirement of secure memory, expensive tamper-resistant hardware, or a PUF access channel. The core of the approach is twofold: (1) Use a model of the PUF to simulate the behavior of the PUF without having direct access to PUF channels (2) Refresh keys periodically using a response challenge feedback loop.

As with most PUF protocols, we have two phases: enrollment, when the PUF model is constructed, and key reconstruction, when the PUF is activated to create session keys.

#### 6.2.1 Enrollment

In the proposed method, the device will be provided with a unique serial number (See Fig. 6.4). Before the device is distributed to end users, the PUF will be characterized through open channels that allow access to the PUF input and output. Machine learning algorithms have been used to attack strong PUFs in order to develop a mathematical model of the PUF \[158, 172, 173\]. In this work, we use these same type of algorithms to develop a model that, instead of being used to attack the PUF, will be used by a server to simulate the PUF. The parameters/features of the model PUF \( PUF_M \) are derived and stored in
the server. Each database entry in the server consists of a device serial number along with associated PUF model parameters. The space required to store all this data will be still smaller than saving many challenge/response pairs in the server because of the linear number of components. After the PUF model has been characterized, any open channels providing access to the PUF will be removed or fused out.

![Diagram of Enrollment phase for session key generation.]

Figure 6.4: Enrollment phase for session key generation.

6.2.2 Key Reconstruction

Before starting a communication session, both the device and server must reconstruct a common key using the PUF and simulated PUF respectively. The device first presents the server with its unique serial number, and the server will retrieve the database entry associated with the provided serial number. The first time that a key is created, the device internally presents the serial number as a challenge to the device PUF. The response can be used to generate the session key through some function $G$, where $G$ could be the identity function.
function or a hash function if needed to generate an appropriate length key with uniform distribution. The server can retrieve the PUF model $PUF_M$ associated with the device’s serial ID and recreate the same session key using the simulated PUF model (See Fig. 6.5).

![Key generation and Encryption diagram]

Figure 6.5: Session key generation protocol and sending of ciphertext $M_C$ which is the encryption of plaintext $M_P$ using key $K$ derived from PUF or model.

### 6.2.3 Key Refresh

The key reconstruction mechanism, as described, will only allow the creation of a single static key. However, in order to enable ephemeral keys, we need a mechanism to change the keys over time. Previous work has introduced the notion of reconfigurable PUFs which change a strong PUF’s functional behavior either logically or physically [174, 175, 176, 177]. However, these approaches require challenges to be applied to an external PUF interface, thus making the challenges (and responses) visible to potential adversaries who can use this information to generate their own model of the PUF. In our approach, as mentioned above, we remove the PUF access channels so adversaries are not able to apply challenges directly in order to model the PUF. In order to refresh the keys within a device, new keys are generated using a dynamic key generation process as shown in Fig. 6.6. As before, the
key is created from the response of the PUF using the function \( G \). The host server can
determine when to refresh the key or the refresh can be triggered by an internal counter
or timer. When a key refresh is enabled, the last PUF response is fed back to the input
of the PUF through a function \( F \) to dynamically update the challenge. The new challenge
will generate a new PUF response which will create an updated key that can be used for
new session encryption. The refresh mechanism is defined with the following equations:

\[
\begin{align*}
C_i & \leftarrow F(R_{i-1}) \\
R_i & \leftarrow PUF(C_i) \\
K_i & \leftarrow G(R_i)
\end{align*}
\]

Figure 6.6: Block diagram of proposed dynamic key generation.

The server side will follow a similar process to refresh its key so that it corresponds
to the key on the client side. Since both the server and client can deterministically arrive
at the same key, there is no need for a Diffie-Hellman or public key based key exchange
protocol.

6.2.4 Error Correction

The PUF output needs to be stable under various supply voltage and temperature condi-
tions [178]. The output value of the PUF device could vary due to several contributing
factors including changes in supply voltage or temperature conditions. These variations
could alter the output parameters of the PUF devices such as threshold voltages, leakage current, gate voltage, delay, timing and effectively change the output bits of the PUF. In spite of these external variations, the PUF output still needs to remain stable.

In addition, the PUF model may have slight inaccuracies that may cause certain bits in the PUF output to be incorrectly predicted. Thus, both the PUF and model responses, $R'_P$ and $R'_M$, may have errors. The presence of noise for each key means that the keys need to be corrected in both the device and server side with the use of helper data (Figures 6.7 and 6.8). There are a number of error correcting codes that could be used for this purpose, such as Hamming, BCH or Reed Solomon. The resultant helper data from the code will need to be stored in the server database, and provided to the device at initialization and whenever the session key is refreshed. Depending on how much helper data can be stored, that will limit the number of times the session key can be refreshed before the process starts again. The cycle can be extended by using a nonce as an input to the $G$ function as shown in Fig. 6.7.

![Block diagram of proposed dynamic key generation with error correction](image)

6.3 Discussion

The proposed protocol removes the need for a public key or certificate based authentication and instead uses a PUF-based identity. The system depends on the fact that the server keeps a database of all possible PUF models. This database is a potential vulnerability point as it exposes the equivalent of a private key for all possible client devices in a particular
Figure 6.8: Session key generation protocol with error correction.

ecosystem. Since the database is potentially high-value, an attacker may be willing to
dedicate significant resources to use aggressive invasive methods to probe the server. Thus,
the server may need exceptional anti-tamper features. In a practical system, where a home
concentrator device in a security system may be the "server", the concentrator may not
keep the database itself, but may instead communicate with a dedicated cloud service
to retrieve PUF models. In such a case, the database is less vulnerable, but individual
client device PUF data may be exposed within the local server. If an attacker were to
even retrieve a single device PUF model, they could use that to impersonate a real device.
However, the combination of the refreshing keys and the nonce, would make it difficult
for an attacker to keep in sync with the real device. Moreover, one could use monitoring
mechanisms to detect if there are multiple devices with the same ID within the ecosystem.

The approach is similar to that used by the SlenderPUF protocol for device authenti-
cation [179] or the Public PUF (PPUF) public key cryptography mechanism [180]. Both
SlenderPUF and PPUF use PUF models at the server side to help authenticate a client
device. However, SlenderPUF does not address the generation of session keys. PPUF is
most similar to our approach and they present a simple key exchange protocol that can
be used to generate session keys. PPUFs depend on large timing differences between the simulation and actual PUF to distinguish between private (actual PUF) and public (simulation PUF) keys. However, it is impossible to guarantee that the simulation models will remain computationally difficult as compute resources become cheaper and more powerful. Instead, in our approach, we assume that both the actual and model PUFs are secure and are not shared.

We have, to this point, only discussed client device side authentication. Since we assume that only the server would have access to the device PUF model, if it can establish a common key and is able to decrypt messages, we can be assured that it is an authentic server. If the device PUF model leaks, then this assumption is no longer true.

The devices do not store any secure state in non-volatile memory or fuses that could be probed by an adversary. However, an attacker could potentially decapsulate a device and probe internal buses to activate the PUF and retrieve challenge response pairs in order to construct a PUF model. Such techniques are extremely difficult to do while keeping the device powered and we assume that this is beyond the capability of most adversaries or not worth the effort. As mentioned above, even if an attacker were able to model a single device’s PUF, it could not be replicated at scale because monitoring mechanisms should be able to detect that the same ID is being used by copied devices.

If the helper data can not correct errors in the hardware or model PUF, then the server and client will construct different keys and will not be able to communicate. As a result, we assume the error correction is strong enough to correct any errors in the PUF or model with sufficiently high probability.

6.4 Conclusions

As the consumer electronic (CE) market has grown with smart embedded devices, we have addressed symmetric key generation for encryption of the messages of those electronics devices using a low-cost method for customer network frameworks. Our method not only encrypts the data with an ephemeral key but also is able to authenticate the device (client) to a centralized server within a consumer electronic network. In this chapter, we have
proposed a dynamic symmetric session key generation method using strong PUFs and a server-side simulated model of the PUF. With our proposed method, consumer electronic devices will be able to communicate in the network with dynamic session keys which will make the message and information secure and trustworthy. The proposed PUF based session key-based encryption will provide a trustworthy embedded system communication platform which will provide a secure exchange and access of sensitive data. The market of consumer embedded electronic devices will continue to expand; therefore, vendors will need a low-cost secure technique for data encryption as well as authentication of those devices. Trustworthiness of the consumer electronic product requires secure data communication as well as the prevention of any loss of credentials and information of that device. Our proposed session key solution will help to secure the encrypted data during the communication in a lightweight and easily adaptable hardware framework which could be suitable for any form of consumer devices.
Chapter 7

An Efficient Algorithm for Extracting reliable Key from Noisy Data

7.1 Introduction

Many security protocols are based on encryption algorithms, biometrics, and Physical Unclonable Functions (PUFs) [181]. One of the major properties of those security applications that they are sensitive to small variations in their inputs. Therefore, those security primitives cannot be applied directly when the input data are noisy. In the case of biometric authentication, privacy could be stored as a cryptographic key or has to be hashed. However, the biometric data are noise inherent which prevents directly applying a hash function or serve as a cryptographic key. A PUF is a random function which extracts a unique signature from each production unit [30] [147]. A PUF is a multiple-input multiple-output function and because of random nature it is to predict outputs of the PUF. Therefore, PUF could be used for a variety of purposes, for example, key generation, authentication, and anticounterfeiting. However, PUF output data is also noisy, therefore additional processing need be performed to remove the noise without compromising security. Two schemes have been proposed for the error correction: secure sketches and fuzzy extractor. It is assumed
that error correction for those schemes would not leak any confidential information. Those primitives have been proposed in \[182, 183, 184\]. Those schemes are shown in Fig. 7.1.

In the secure sketch (SS), original data \(W\) would be recovered using public information \(S\). The main idea is that public information \(S\) will minimally leak some information of \(W\). A fuzzy extractor derives a secret \(R\) with public information \(P\), where it is assumed that \(P\) does not leak any information. Furthermore, an FE could be derived from a SS using the Universal Hash Functions (UHFs) \[184\]. This also involves a large entropy loss \[185\].

![Secure Sketch and Fuzzy Extractor](image)

Figure 7.1: Secure sketch and fuzzy extractor.

The challenge in making a cryptographic key using biometric or PUF data is that the key cannot be reproduced exactly the same \[186\]. The noise could be introduced inevitably during the data acquisition and processing from biometric and PUF samples. Several mechanisms have been discussed in the literature on how to extract a reliable cryptographic key from such noisy data. A few techniques propose to correct the noise in the data by using the public information \(S\) which has been derived from original biometric data \(W\). Those techniques are fuzzy commitment \[183\], fuzzy vault \[187\], helper data \[188\], and secure sketch and extractor \[182\]. Two components are defined in a secure sketch protocol. The first component is decoder which is used for sketch generation algorithm. It converts biometric or PUF output data \(W\) into a sketch \(s\). A decoder is used during the second algorithm which reconstructs \(W\) from the biometric or PUF data \(W'\) and the sketch \(s\). The \(W\) and \(W'\) have to be sufficiently closer for reconstruction. It is assumed that the sketch \(s\) would not reveal too much information, however, Dodis et al. \[182\] provided that there will be an entropy loss which gives the measurement that sketch \(s\) gives to an adversary.
for guessing the input data $W$, provided that $W$ is discrete. Furthermore, entropy loss gives the worst-case bound for all distribution of $W$. There are several difficulties such as applying error-correcting code and entropy loss for many known secure sketch techniques described in [189]. In this chapter, we study how to extract secure key with error-correcting capability for minimal entropy loss from PUF data.

### 7.2 Related Works

The secure sketch depends on the representation of the PUF and biometric data. Furthermore, the construction of the data also measures the underlying distance function. The techniques described in the literature assume that noisy data are presented as points in some metric space. The fuzzy commitment method [183] is based on error-correcting codes which consider binary strings and the similarity is determined by Hamming distance. The fuzzy vault scheme [187] corrects error using polynomial interpolation and the method considers a set of elements in the finite field with a set difference as a distance function. The fuzzy extractor described by Dodis et al. in [182] applies a hash function after the original data $W$ is reconstructed from noise affected data. The secure sketch [182] is analyzed by three metrics and those are set difference, edit distance and Hamming distance. The authors described a secure sketch in [190] by finding the similarity measure used for fingerprints. This method doesn’t define a metric space and each template is consists of set points in 2-D space.

There are other approaches have been proposed to extract keys from biometric data which have been represented differently than the abovementioned methods. Those approaches are to extract biometric templates which include face biometrics [191], fingerprints [192], iris patterns [193], voice features [194] and handwritten online signatures [195]. However, those techniques are not efficient for security compared to cryptographic techniques. Those approaches are analyzed by entropy loss of the data and effort and time needed by an attacker using a brute-force attack.

The authors in [196] showed that a secure sketch technique could be insecure if multiple sketches have been derived for the same data. The authors analyzed the security of secure
sketches for general attacker models in [197]. They also proposed mutual authentication
techniques. Mutual information also proposed in [184] against the attacker for biometric
data. They have considered zero mean of Gaussian random vectors for biometric templates
as biometric templates. A similar approach has been proposed in [188] when verifier could
be trusted and distribution of the original data is known. In [194], the authors have shown
some practical results of those security measures for authentication.

Bosch et al. implemented a fuzzy extractor in [198] to generate cryptographic keys from
a noisy response with error-correction using Physical Unclonable Functions (PUFs). They
also reported hardware resource utilization when helper data algorithms are implemented
in FPGA for PUFs. The authors compared various linear codes constructions in FPGA,
for example, Reed-Muller and Golay codes with advantages and shortcomings. However,
they didn’t report entropy loss in [198] of the data for implementing Fuzzy extractor.

### 7.3 Entropy Loss in Secure Sketch

Secure sketch and entropy loss are defined in [182]. Let define $S \subseteq M \times M$ with a similarity
relation, where $M$ is a finite set of points. If $(W, W') \in S$, then we could say that pair
$(W, W')$ is similar or $W'$ is similar to $W$. Furthermore, if we define a secure sketch with
$(M, S, Enc, Dec)$, where $Enc : M \rightarrow \{0, 1\}^*$ is an encoder and $Dec : M \rightarrow \{0, 1\}^* \rightarrow M$ is
a decoder, where $(W, W') \in M$. Then it could be shown that $Dec(W', Enc(W)) = W$, if
$(W, W') \in S$. Now, if the string $P = Enc(X)$ is the sketch, then it would be made public.
Then, the scheme is $L$-secure for all random variables $W$ over $M$ and the entropy loss of the
secure sketch (SS) $P$ will be at most $L$. The entropy loss is, $H_\infty(W) - \tilde{H}_\infty(W)(Enc(W)) \leq L$.

![Figure 7.2: Error-correcting code for the proposed PUF method.](image)
7.4 Methodology

![Generation Matrix of the (15,11) Hamming code.](image)

<table>
<thead>
<tr>
<th>D3</th>
<th>D5</th>
<th>D9</th>
<th>D10</th>
<th>D11</th>
<th>D12</th>
<th>D13</th>
<th>D14</th>
<th>D15</th>
<th>P1</th>
<th>P2</th>
<th>P4</th>
<th>P8</th>
</tr>
</thead>
<tbody>
<tr>
<td>D3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 7.3: Generation Matrix of the (15,11) Hamming code.

![Probability of getting the correct codewords versus PUF output responses for (3,1) Hamming code.](image)

Figure 7.4: Probability of getting the correct codewords versus PUF output responses for (3,1) Hamming code.

A \([n, k, d]\) code is \(n\) bits long, of which \(k\) bits are data bits, \((n - k)\) bits are ECC bits and all codewords are minimum distance \(d\) bits apart - meaning we can correct \((d - 1)/2\) errors (See Fig. 7.2). If the PUF generates an \(n\)-bit response, that means of the \(2^n\) possible responses only \(2^k\) are valid codewords. Thus, the probability of getting a valid codeword is \(2^k/2^n\) which may be quite small. Instead, if the PUF generates an \(r\)-bit response where \(r > n\), we may be able to select \(n\) bits from the \(r\) bits such that it is a valid code. The larger that \(r\) is, the easier it is to find the correct \(n\) bits. We will need helper data to
identify those $n$ bits.

For example, the probability of getting the valid codeword of the perfect Golay code $[23, 12, 7] \text{ is } 2^{12}/2^{23} = 1/2048$. Therefore, to increase the probability of getting the perfect code, we could increase the PUF response and helper data will be used to get the ECC bits.

Now, if we want to get a valid codeword using a Hamming code, we will need a generator matrix. An example of $(15,11)$ generator matrix is shown in Fig. 7.3. Now, using the generator matrix and PUF responses, the probability of getting the correct codewords for
different Hamming codes is shown in Fig. 7.4, Fig. 7.5 and Fig. 7.6 respectively. From these figures, we can see that as the number of PUF response bits increases, the probability of getting the correct codeword also increases.

7.5 Conclusion

Physical Unclonable Functions (PUFs) are used for a variety of security-related applications. However, to generate cryptographic keys an efficient helper data algorithm needed to extract entropy from noisy and non-uniform random PUF responses. In this work, we have proposed a probability-based algorithm to extract codeword from noisy data and we have shown the codeword extraction solution for different Hamming codes.
Part V

Anti-RE of Flash Memory
Chapter 8

Anti-RE Techniques for Flash Memories against Backside SCM Probing

8.1 Introduction

Integrated circuits (IC’s) are used for a variety of applications, such as smart cards, RFID tags, mobile phones, Pay-TV chips, cash machines and military weapon systems. These chips often contain security keys to protect personal and confidential information [42] [82]. To protect integrated circuits (IC’s) from tampering, companies, researchers, and aerospace and defense (A&D) industry are aiming to secure confidential data stored in the integrated circuit [40] [39]. An example of tampering attack consists decapsulation of the IC and etching the die from the front side to access the metal layers to measure the signals using focused ion beam (FIB) [167] [199].

To prevent tampering attempts, various methods have been proposed before. Tamper protection materials and sensors could be used against theft or probing [62]. To defeat tampering, ceramics, bricks or steel could be used to separate the top layer of the IC. Single chip coatings might be used to foil pico-probing attempts. Brittle packages, polished packages, or aluminum packages could be exploited to protect electronic devices. Other
packages of interest could be bleeding paint, holographic and tamper responding tapes and labels. To protect an electronic device, different sensors could be used, for example, optical sensors, voltage sensors, radiation sensors, probe sensors, motion sensors, and top metallization sensor meshes [62]. Note that, attackers could attack the IC from front-side as well as from backside. The side of the semiconductor device where the circuitry is provided is defined as "frontside" of the semiconductor device [120] (See Fig. 8.1(a) and 8.1(b)). The opposite side of the semiconductor device to the frontside is defined as "backside". The "frontside" and "backside" of the semiconductor device are also referred to as the "first side of the substrate" and "second side of the substrate" respectively [120]. Existing protection schemes are unable to protect backside attacks of the IC by Focused Ion Beam (FIB) [200] or Scanning Capacitance Microscopy (SCM) probing [47].

Although EEPROM and Flash memory have the same frontside and backside structures discussed above, EEPROM/Flash memory technology had long been considered immune to advanced tampering techniques. For example, attackers can not detect the memory contents by non-destructive X-Ray technology, because EEPROM/Flash logic states are represented by the presence/absence of electrons in a floating gate. Thus, they are not visible by simple imaging techniques as would circuit structure be revealed by the physical and geometric appearance [24]. Furthermore, traditional front-side destructive analysis using scanning electron microscope (SEM) and transmission electron microscopes (TEM) will disturb the electron distribution in the floating gate (FG) [47].

However, more recently, Scanning Kelvin Probe Microscopy (SKPM) and Scanning Capacitance Microscopy (SCM) procedures have been proposed in [47] [16] [48] to extract the information of the EEPROM and Flash memory accurately. SKPM and SCM are performed from the backside of the IC to prevent the charge distribution on the floating gate from being disturbed. As these procedures are performed from the backside of the memory, the entirety of the bulk silicon on the backside must be removed. SKPM probing is done by measuring the floating gate potential through the tunnel oxide layer. The thickness of the tunnel oxide layer is about 10 nm, so that it isolates the floating gate from the transistor channel. To avoid charging and discharging of the floating gate, the backside removal must leave 10 nm undamaged tunnel oxide layer. After that, a DC voltage can be
applied to the SKPM probe tip to read the bit value [16].

SCM probing measures the capacitance variations with the sample [47]. A high-sensitivity capacitance sensor is equipped on the SCM. The capacitance of the sample is measured by the tip in the contact mode. And then, capacitance variations between the sample and equipped capacitance sensor is measured. As holes are paired in the transistor channel with the electrons of the floating gate, the capacitance sensor can measure
the memory contents by probing the hole concentration in the channel. In this case, it is necessary to keep a silicon thickness of 50-300 nm after backside delayering to leave an undamaged transistor channel. Logic states can then be read through the backside by probing the carrier (hole) concentration in the transistor channel. Because of technology scaling, stored electrons in the floating gate for 90 nm-node NAND flash have been reduced to less than 1000 electrons \cite{16}. At that scale, SKPM methods can not differentiate two logic states accurately, but the SCM method will still performs well. Furthermore, the resolution of the SCM is 15 nm, so it is possible to probe for 45 nm or lower technology nodes \cite{47}.

Therefore, we will consider only SCM probing for tampering by attackers, and how to successfully resist tampering attempts from SCM probing. We have proposed two efficient schemes against backside SCM probing. The first scheme protects from backside probing by connecting a metal with the floating gate of the transistor. This protective metal will go down through one side of the transistor channel from the floating gate, and then the extended metal at the bottom stays parallel with the transistor channel. Therefore, if attackers are trying to probe with SCM, then the tip will come in contact with the added metal layer allowing any stored charge to discharge from floating gate. The second scheme prevents SCM probing by adding a $p^+$ layer just below the transistor channel. This added $p^+$ layer will prevent SCM probing by disturbing the capacitance measurement via SCM tip. The advantage of our proposed tamper-protected methods is that these schemes can prevent tampering or reading of the memory even when the chip is powered off.

The rest of the chapter is constructed as follows: Section 8.2 will provide related work of IC anti-tampering methods. Next, in section 8.3, we will present our proposed anti-tampering schemes in detail.

### 8.2 Related Work

To avoid disturbing the existent floating gate charges, attackers prefer tampering from the backside of the EEPROM/Flash memory. Therefore, the most effective anti-tampering methods should protect memory from backside attacks. Different detection schemes have
been proposed against backside tampering attacks. These existent backside anti-tampering techniques can be categorized into two groups [24]: circuit parameter sensing and light sensing.

**Circuit Parameter Sensing**

If the attacker polishes the backside of the memory, it will thin the bulk silicon. A capacitor is formed by burying two parallel plates in the bulk silicon of the IC backside [118]. When the attacker delayers the IC from the backside, the capacitive sensor will detect the capacitance reduction. The erase operation will be activated in the EEPROM/Flash memory, when the capacitance reaches below a pre-defined threshold. [201] proposed a method to compare substrate resistance variation between two adjacent regions for protecting an integrated circuit against backside attacks. In this method, the system compares current flowing between the two adjacent regions with a reference threshold. Similarly, resistance variation is measured in [120] to compare with certain reference resistance threshold to detect backside tampering.

**Light Sensing**

An optical monitor system against backside tampering was proposed in [121]. In this technique, at least one light-emitting and light-sensing pair of devices are placed on the front-side of the IC and a light reflection module is placed on the backside of the IC. The working principle of this optical monitor system is that light-emitting device emits light which penetrates through the silicon bulk, and then the light is reflected back by the light reflection module, and finally, the light is collected by the light sensing device. If delayering has been performed by the attacker, the light distribution will be changed and can be detected at the light-sensing device. This detection can initiate a process to self-erase the memory contents of the EEPROM/Flash. Similarly, another optical anti-tampering method was proposed in [202]. In this technique, the light source and light sensors are equipped inside the IC. Light is emitted from the light source, and then the light interacts with the dispersed particles, and then finally sensed by light sensors. This optical method will form a characteristic interference pattern and will produce cryptographic key. Note,
that it is required to encapsulate the IC so that light can not enter inside the IC. Therefore, if the package is perforated by the attacker, then light will escape from inside which will change the interference pattern, and tampering will be detected.

The drawbacks of the circuit parameter sensing and light sensing anti-tampering methods are associated with power consumption. These methods could protect the IC in the active state, but the protection scheme could be foiled if the chip is in the "OFF" state. When the chip is in "OFF" state, sensors and control circuits are ineffective against tampering. Furthermore, optical sensing procedure has more complex structure which will increase the manufacturing cost. Additionally, light emitting and light sensing devices will need continuous power consumption. Since flash devices are obviously non-volatile, its data is retained even without power, meaning it is possible to retrieve the data when there is no power. Therefore, it is essential that the anti-tampering techniques operate without power.

8.3 Backside Protection Schemes

Firmware and netlist information could be stored through read-only memory (ROM), electrically erasable programmable ROM (EEPROM), or Flash memory. The benefit of EEPROM/Flash is that it has the capability to reprogram the memory contents. One-bit EEPROM cell is shown in Fig. 8.2(a). This cell is comprised of floating gate transistor (FGT) and select transistor (ST) [21]. Furthermore, floating gate transistor is composed with two gates - control gate (CG) and floating gate (FG). The bit value information is stored in the FGT by presence or absence of the electrons. Because FG is isolated electrically, it could keep the electrons when power is off. Similarly, flash memory cell is shown in Fig. 8.2(b). Flash cell has almost same structure except it doesn’t have the ST. ST allows EEPROM to be byte addressable.

Given that in the EEPROM/Flash, stored data are related with absence or presence of a certain electron level in FGs. The memory cell is considered as "0" state (blocked) or "1" state (conductive) based on electron levels in FGs. The attacker will try to read (measure) those stored charges(electrons) by backside SCM probing.
Following conventions are used when electrons are stored at the FG as Flash data contents [16]:

- When a cell has $10^3$ to $10^5$ electrons stored at the FG, it has a high threshold voltage, for example, VT1 and considered as "0" logic state. But, if there are no electrons at the FGs, it will have a low threshold voltage, for example VT2, and corresponds to "1" logic state.

- If a read voltage is applied to the CG between VT1 and VT2, "0" logic state the cell remains off (blocked), but "1" logic state cell will be conductive.

As charges are stored in the FGs, sample preparation consist of backside delayering of the die to come as near as possible of the tunnel oxide layer without removing the charges. SCM is performed by probing the "mirror charge" (holes) in the transistor channels which are coupled of the one stored onto FGs (electrons). To leave undamaged transistor channel, the SCM principle required to keep a backside silicon thickness of 50 - 300 nm.
Fig. 8.3 shows the measured bit information when a flash is read by SCM procedure. Note that flash data are related with electron concentration of the FG which make a cell conductive or blocked. The SCM signal depicts that the charged FG (associated with blocked state "0") has a darker signal (in black circles), which has a mirror high density of holes in the transistor channel. But, for the conductive state "1", there will be low density of holes in the transistor channel, so that the signal coming from transistor channel will be brighter (in white circles).

As from previous discussion, to probe with SCM, attackers need to delayer from IC backside to avoid removing the charges from FG. Thus, the most efficient anti-tampering method could be to foil backside attacks. In the following, we have proposed two efficient backside attack protection schemes. The benefits of our proposed backside protection schemes are that a chip doesn’t need be in "active state", i.e., the chip could be protected without power with those schemes.
8.3.1 Scheme 1: Metal Connection from Floating Gate

Our proposed protection scheme 1 for flash cell is shown in Fig. 8.4. A metal layer is connected with the floating gate of the transistor (See Fig. 8.4), going down parallel to the side of the transistor channel and then come back lower to transistor channel. Note that, as SCM procedure requires backside silicon thickness 50 nm to 300 nm to probe transistor channel, the anti-probing metal need to go down upto 300 nm. So, if an attacker tries to probe with SCM, the anti-probing metal come in contact with electrical probe and charge will vanish from floating gate via anti-probing metal.

![Figure 8.4: Protection scheme 1.](image)

It is worth mentioning that the anti-probing metal line which is perpendicular to the backside silicon was challenging to achieve previously. But [119] proposed through-silicon via (TSV) technique which will make fabrication much easier.

In Fig. 8.5 and Fig. 8.6 layout of a NAND flash and NOR flash are shown respectively. Similarly, layout with protection for NAND flash and NOR flash are shown in Fig. 8.7 and Fig. 8.8 respectively. We have depicted cross sectional and top view image with protective metal layer in Fig. 8.9 and Fig. 8.10 respectively. Note that it is required to add SiO$_2$ of the outside of the protective poly layer, so that SiO$_2$ will resist charge to vanish from
As we are adding extra metal layers, we have to validate how much area will be increased for this technique. Cadence DRC verification for 45 nm node has shown in Fig. 8.11. For this node, minimum width of poly is 50 nm and spacing between poly is 140 nm [203]. We have proposed to add poly metal line which will use as protective metal line, so it will perfectly fit between two diffusion layers (See Fig. 8.7 and Fig. 8.11). So area will not increase after adding those protective poly metal lines, as the minimum spacing between two diffusion layers is 80 nm for 45 nm technology node [203].

To make our scheme more efficient, we could make zigzag metal protection method which has shown in Fig. 8.12. As, now the spacing will be checked by DRC between protective metal lines diagonally, the area will not be increased after placing the protective metal lines in that way (See Fig. 8.12).
8.3.2 Scheme 2: Adding p+ Layers

As mentioned before, the SCM procedure requires backside probing to measure the capacitance to the floating gate through the transistor channel. Our first scheme adds a discharge path from the floating gate to prevent this probing. However, the technique adds some area overhead to the flash layout. As an alternative scheme we suggest simply adding a P+ layer below the transistors as a barrier (See Fig. 8.13). The distance of the added P+ region from inversion layer is approximately 20 nm. So, when attackers will try to measure capacitance variation through SCM probing, the added P+ layer will act as a shield and disturb the capacitance measurement. As our proposed second scheme only requires adding P+ layer, it will not increase area of the flash cell. The only way to defeat this mechanism is to not only remove the bulk silicon but also the p+ layer as well. This would require delayering accuracy to within 20 nm of the channel and may be difficult to do consistently.
If newer SCM techniques allow consistent probing to within 20 nm of the channel, then we can fall back on our metal connection scheme.
Thus, the two schemes offer a choice between the lower area cost with the p+ layer and the higher resistance against high-resolution future probing attacks with the metal connection scheme. Table 8.1 shows a summary of the advantages and disadvantages.
Table 8.1: Design and Manufacturing Costs and Implementation Challenges of Anti-Tampering Techniques, where Very High = Most and Very Low = Least

<table>
<thead>
<tr>
<th>Anti-tampering Techniques</th>
<th>Fabrication Cost</th>
<th>Area</th>
<th>Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scheme 1: Metal connection from floating gate</td>
<td>Moderate</td>
<td>Low</td>
<td>Very High</td>
</tr>
<tr>
<td>Scheme 2: Adding p+ Layers</td>
<td>Low</td>
<td>Very low</td>
<td>High</td>
</tr>
</tbody>
</table>

of the schemes. The classifications from very low to very high are rough assessments of the feasibility of the anti-tampering schemes. Fabrication cost reflects the cost from a fabrication perspective and reflects yield loss as well. Note that both techniques complicate the fabrication process as they require etching beyond the transistor wells. The metal connection scheme is slightly more complicated because of the vertical connection from the floating gate. Area assesses the impact on overall area of the memory design and reliability.

Figure 8.11: Cadence layout verification for NAND flash with protection for 45 nm node.
is a measure of the scheme’s resistance to attack. It is worth mentioning again that the added metal lines and P+ region protect the flash in the unpowered state, and the circuit will not draw additional power.
8.4 Conclusion

In this work, we have presented two new anti-tampering methods to detect backside attacks of Flash memories using scanning capacitance microscopy (SCM). The proposed anti-tampering technique has been verified in a 45 nm CMOS process. Our proposed techniques can protect memory when it is in an unpowered state, so the memory will not dissipate additional power. Our proposed anti-tampering schemes can provide low-cost, low-area, low-power and efficient mechanism to protect memory with keep in mind all constraints.
Part VI

Conclusions and Future Directions
Chapter 9

Conclusion and Future Work

This thesis has focused on reverse engineering, piracy and counterfeiting of chip to system level and countermeasures against those attacks: LoPUF (authentication of printed circuit board and ICs based on resistor and capacitor variations), key generation for hardware obfuscation, reliable session key generation using strong PUF modeling, and anti-RE techniques against backside SCM probing.

It is very hard to defeat reverse engineers in their attacking attempts, but a complex and protective anti-RE system could be devised that would be so time-consuming, difficult, and expensive that it could deter most forms of RE. In the meantime, if the reverse engineering of adversaries is successful, the technology could be superseded by its next-generation version [57]. There are several anti-RE hardware obfuscation, and key generation technologies discussed in Sections II, III, IV and V but it is worth mentioning that some of these techniques could be vulnerable to RE attacks. For example, camouflage tries to make imaging, etc. difficult. Therefore, if someone delayers the chip, board, or system destructively, they could find the full functionality. Obfuscation doesn’t prevent imaging but makes the functionality of the design ambiguous or locked. After conducting a destructive analysis, a reverse engineer could extract high-level netlists so that they could find the functionality [41]. There are other, chip-level techniques like hardware metering, EPIC, and reconfigurable logic barriers that could be used as anti-RE techniques, but most of these methods are used for anti-piracy. Metering, reconfigurable logic barriers, and EPIC have limited uses for anti-RE because, if someone can extract the key by applying backside
attacks, RE is trivial. Also, applying anti-RE techniques to the board-level can be very challenging because the board is much more vulnerable to RE due to its simple laminated structure.

Also, most of the existing anti-RE techniques are merely deployed against RE attacks from the front side of the system, leaving the system more vulnerable to the backside attacks \[204\]. As discussed in chapter \[8\] it is much easier and quicker to decapsulate the chip from the backside to expose the memory contents. New dedicated countermeasures against backside attacks are very much in demand in the future from the system design perspective. Though some of the anti-RE techniques used against EEPROMs/Flashs reverse engineering are deployed particularly for the backside attacks, most of them rely largely on the embedded power supply (such as embedded battery) to sense the invasion from the backside. From the attackers’ perspective, these countermeasures can be broken without too much difficulty, since the active sensing mechanisms are easily bypassed once the power supply is recognized and isolated by the attackers.

Although a large amount of recent research and new developments have appeared on anti-RE techniques for decades, there still exist many open problems that will need additional research in the future. In the following section, we will list some important issues that still need to be addressed.

1. Currently, measuring quantitatively how strong the anti-RE techniques are still remains an open challenge. Lack of metrics for evaluating the efficiency of anti-REs will delay their mass deployment in the industrial sector. By viewing side-channel attacks as a communication problem, a good measuring example in \[12\] \[205\] demonstrates that both the leaked information obtained by side-channel attacks and the effect of the adopted countermeasures can be measured from the information theory viewpoint.

2. In practice, anti-REs will inevitably cause other issues, such as reliability, power consumption and area overhead. The tradeoffs between reverse engineering resistance, reliability, power consumption, and area overhead should be thoroughly investigated before applying the anti-REs in different electrical systems.
3. Currently, most anti-RE techniques are proposed independently. Integrating two or more anti-RE techniques in the same design can definitely improve the hardware security against reverse engineering. Take the active sensing mechanisms of anti-RE techniques for EEPROMs/Flash for example: the power supply can be hidden by the existing camouflage techniques to further increase the complexity of reverse engineering. Additionally, the cost of reverse engineering will be increased if BGA packages are used with multilayer PCBs [10]. In this way, the interconnections will be hard to observe, and BGA pins on chip will be impossible to access for analysis. It is worth mentioning that desoldering and the decapsulation of BGA packages are harder to break than plastic packages. Also, interconnection obfuscation could be applied by introducing dummy ICs in the PCBs [206]. This technique will scramble the traces of the board, so RE could not discover the exact design of the PCB. The problem of how to optimally combine different techniques, however, still remains an open issue.

4. As far as we know, most of the current anti-RE techniques basically provide security features attached to the original designs, which do not consider anti-RE capability. In the long run, electrical systems are in urgent need of the research and development of new approaches with inherent resistance to reverse engineering. For example, as the technology of 3D IC matures, it is believed that the 3D structure will possess the inherent potential to resist reverse engineering because the die in 3D IC structures is less observable compared with traditional IC structures.

5. To protect against noninvasive attacks, some dummy metals or ceramic powders could be used inside the internal structure of the chip or between the board layers without changing its functionality. These materials highly attenuate the X-ray and create artifacts in the reconstructed images of the tomography. Therefore, the reverse engineer cannot extract the desired information about the layout after X-ray imaging.

Anti-Reverse engineering and security solutions for electronics hardware and embedded systems become an integral part of our daily life, therefore it is important to improve security measures as well as performance, cost, performance, reliability, and area requirements.
Hence, designers need to consider all aspects of those requirements and find a viable solution between cost and performance. To address all issues, Physical unclonable functions (PUFs) have become an important security primitive with a lot of advantages. In this thesis, we have developed several novel designs and new architectures for hardware-based authentication and key generation using PUFs. The major contributions of our thesis will be summarized in this chapter.

9.1 Authentication of Printed Circuit Boards

Physical Unclonable Functions (PUFs) are probabilistic circuit primitives that extract randomness from the physical characteristics of a device. PUFs are easy and simple to implement and its random nature makes its behavior hard to predict and model. Most existing PUF designs are based on variation at the chip level and can not be implemented in a printed circuit board (PCB). Therefore, these PUFs cannot be used to protect against counterfeit PCBs in a distributed supply chain. In Chapter 3, we have proposed a novel PUF design based on resistor and capacitor variations for low pass filters (LoPUF). We demonstrate the setup in a protoboard for different resistor-capacitor pairs (RC pairs) for reliable low pass filter PUF. Because of process variations, the voltage will be different at the same cut-off frequency for our proposed PUF. Finally, the output of the filter is connected to an inverter to measure the pulse width and best suitable pulses are used for ID generation based on our algorithm.

9.2 LoPUF: Authentication of IC’s based on RC variations

Because of the outsourcing of chip fabrication, piracy, reverse engineering, and counterfeiting are major concerns for the government as well for the industry. To prevent those issues, Physical Unclonable Functions (PUFs) based methods have been proposed for chip authentication. PUF extracts randomness of the manufacturing variations of a device that is probabilistic in nature. PUF implementation is easy and simple, however, because of random characteristic makes its response hard to predict and model. Most existing PUF techniques such as that arbiter, ring oscillator, and lightweight secure PUFs will be hard
to implement inside the chip because of large area overhead which will not be a feasible solution for IC authentication. Hence, instead of using external sources for extracting variability, we have proposed a novel low pass filter PUF design in Chapter 4 that is based on the internal resistor and capacitor variations of an IC. We have simulated the setup in Cadence virtuoso for a 45 nm technology node. The resultant output voltage of our proposed PUF will be different at the same cut-off frequency because of the manufacturing variations. Finally, the output of the low pass filter PUF is connected with an inverter to make the signal digital and after that, a digital counter is connected with the inverter to measure the pulse width that can be used for the key generation to authenticate the chip.

9.3 Key Generation for Hardware Obfuscation using Strong PUF

As a result of the increased use of contract foundries, IP theft, excess production and reverse engineering are major concerns for the electronics and defense industries. Hardware obfuscation and IP locking can be used to make a design secure by replacing a part of the circuit with a key-locked module. In order to ensure each chip has unique keys, previous work has proposed using physical unclonable functions (PUF) to lock the circuit. However, these designs are area intensive. In Chapter 5 we have proposed a strong PUF-based hardware obfuscation scheme to uniquely lock each chip.

9.4 Session Key Generation using Strong PUF Modeling

Consumer electronics and embedded systems must ensure trusted authentication and secure communications. Communication session encryption keys are negotiated in a key agreement protocol enabled by a public key/private key pair for authentication. However, embedded systems are limited in their capability to implement public key encryption and client-side authentication. In Chapter 6 we introduce an authentication and session key generation mechanism using Physical Unclonable Functions (PUFs) that extracts randomness from device physical characteristics. The method uses PUF models to help with key
exchange and creates periodically refreshable. The approach mitigates tampering attacks as no key is stored on device.

9.5 An Efficient Algorithm for Extracting Reliable Keys from Noisy Data

In Chapter 7, we have presented an efficient algorithm for error-correcting from the noisy data. This algorithm could be used to extract cryptographic keys from the noisy response of a Physical Unclonable Function (PUF) and biometric data. PUF is an attractive solution for authentication and secure key generation. Our proposed solution will be an effective method that could turn noisy PUF data into reliable and secure keys usable for any cryptographic application.

9.6 Anti-RE Techniques for Flash Memories against Backside SCM Probing

Preventing unauthorized reading or tampering of non-volatile memories is often a crucial issue for critical systems in order to protect sensitive data such as cryptographic keys or software. Flash memories, even when embedded within other devices, are known to susceptible to advanced attacks that probe internal device topologies. Developing efficient and low cost anti-tampering technologies is an important issue in highly secure systems. In chapter 8, we have presented possible backside attacks by scanning capacitance microscopy (SCM) of the Flash memories and anti-tampering methods to mitigate them. The proposed anti-tampering schemes will not dissipate additional power, as they are fully operational even in an unpowered state. The design rules have been verified using Cadence 45 nm CMOS technology. Our proposed solution will exhibit low power consumption, low cost and low area design to protect Flash memories from advanced attackers who have advanced SCM technology.
9.7 Future Work

This thesis work has demonstrated potential anti-RE techniques and authentication using PUFs, still, there are many open opportunities possible to extend those works. Several potential lines of research that have been raised from our thesis work. We will discuss those directions in this section. A few ideas that could be explored are discussed below:

A potential area of investigation is to explore trusted authentication and secure communications for consumer electronics, medical devices and embedded systems using machine learning algorithms. Communication session encryption keys are negotiated in a key agreement protocol enabled by a public key/private key pair for authentication. However, embedded systems are limited in their capability to implement public-key encryption and client-side authentication. Future research could build on the PUF-based authentication and session key generation mechanism we have introduced in Chapter 6 to help with key exchange and create periodically refreshable keys. The approach will mitigate tampering attacks as no key will be stored on the device. An implementation of our secure key method in hardware, for example, in FPGA, with Cadence verification for the industrial and bio-medical chips could validate these techniques.

Further exploration of hardware security countermeasures on electronics devices, IoT, network infrastructure and medical instruments is critical because modern self-driving cars, drones, mobile and electronics devices, and medical devices such as ECG, heart rate, and insulin monitors are used to assist our daily lives. Rogue attacks by cyber-criminals and hackers on those sensitive domains and critical infrastructures will be disastrous. Non-linearity behaviors on Physical Unclonable Functions (PUF) could be used to make a secure key for those devices. Furthermore, Intellectual Property (IP) theft, overproduction, counterfeiting, and reverse engineering can be investigated with this hardware security research and how to find a remedy on those attacks.

Also, the RC based PUFs we have presented in Chapters 3 and 4 should be implemented and fabricated to validate our authentication method in PCB and ICs as well as for the Internet of Things and embedded systems. That will help us to find the performance of our methods in real situations.
9.8 Summary of Conclusions

Reverse engineering (RE) of electronic chips and systems can be used with honest and dishonest intentions. To inhibit reverse engineering (RE) for those with dishonest intentions (e.g., piracy and counterfeiting), it is important that the community is aware of the state-of-the-art capabilities available to attackers today, and should use efficient and reliable design to prevent those dishonest attacks. In this dissertation, we have devoted to developing and implementing a series of methods and designs based on Physical Unclonable Functions (PUFs) for integrated circuits and electronics hardware, in order to protect against dishonest or malicious attacks, for example, reverse engineering, piracy, and counterfeiting. Our proposed PUF-based security primitives provide secure, efficient, reliable, lightweight, and low-cost hardware platforms for the integrated circuits, IoT, consumer electronics, medical devices, and electronics hardware.
Bibliography

plementation into mechanical components,” Research in Engineering Design, vol. 22,
no. 4, pp. 245–261, 2011.

wiki/CMOS

edu/users/06/adem/engin/e77vlsi/lab3/

03-1bw.html.

dual-in-line-package


circuit camouflaging,” in Proceedings of the 2013 ACM SIGSAC conference on Com-

gencontent.tsp?templateId=5909&navigationId=12626&contentId=153966


sis,” University of Cambridge Computer Laboratory, Tech. Rep. UCAM-CL-TR-630,
2005.


