SECURING ANALOG INTELLECTUAL PROPERTIES AND CLOUD FPGAS

A Dissertation

by

NITHYASHANKARI GUMMIDIPOONDI JAYASANKARAN

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Jiang Hu
Co-Chair of Committee,	Jeyavijayan (JV) Rajendran
Committee Members,	Samuel Palermo
	Duncan M. (Hank) Walker
Head of Department,	Miroslav M. Begovic

December 2021

Major Subject: Computer Engineering

ABSTRACT

There are a plethora of hardware security threats depending on the source from which they arise. One of the sources of threats is the globalization of the integrated circuits (IC) supply-chain. With the outsourcing of IC fabrication, the semiconductor industry faces several challenging security threats. These threats include but are not limited to intellectual property (IP) piracy, reverse engineering, and overproduction. Another source of threat is resource sharing, where more than one user accesses a single resource. If the cloud field-programmable gate array (FPGA) servers allow more than one user to access a single FPGA, data leakage can occur between the users. Though there are various other sources for hardware security threats, in this dissertation, we focus on the threats based on the sources mentioned above.

In the first work, we propose a defense technique to secure analog and mixed-signal (AMS) circuits against overproduction. We perform "logic-locking" on the digital section of the AMS circuits. The idea is to make the analog design intentionally suffer from the effect of process variations that impedes the operation of the circuit. Our results show that, only on applying the correct key, the analog circuit performs as desired.

For the second work, we propose an attack technique to evaluate the resilience offered by the existing defenses that secure analog-only and AMS circuits with the help of satisfiability modulo theories (SMT) and Boolean satisfiability (SAT). We demonstrate our attack on five analog locking techniques and three AMS locking techniques. The attack is demonstrated on commonly used circuits, such as bandpass filter (BPF), LC oscillator, triangular waveform generator (TWG), and quadrature oscillator.

For the third and final work, we investigate the impact of primitive-level placement on power-side channel (PSC) attack on cloud FPGAs and the defenses that thwart them. Also, the impact of additional logic that resides along with the advanced encryption standard (AES) on the correlation power analysis (CPA) attack is studied. Our results showcase that the AES along with filters and/or processors, is sufficient to provide better security compared to the existing defenses.

DEDICATION

To my mother, my father, my husband, and all my well-wishers.

ACKNOWLEDGMENTS

This work would not have been possible without the support and help of many with whom I am closely associated. First and foremost, I would like to share my heartfelt thanks and gratitude to my advisors Dr. Jiang Hu and Dr. Jeyavijayan Rajendran, for their constant support and guidance throughout my degree. Their support has been imperative, without which this dissertation would not have been possible at all. I am also fortunate to get taught by both of them on several nuances in choosing a research problem, executing research, and writing good journals. They also gave me regular valuable feedback helping me grow professionally and instill self-confidence. I am indebted for my entire life for the opportunities they gave me during my study here.

I want to extend my sincere thanks to the late Prof. Edgar Sánchez-Sinencio for the fruitful design discussions and valuable suggestions he gave for this work. I am grateful to my thesis committee members Prof. Duncan M. (Hank) Walker and Prof. Samuel Palermo, for their helpful suggestions and comments on my research and dissertation.

I was fortunate to have a great set of people to work with during my five years. I sincerely thank Dr. Adriana Sanabria-Borbón, Hao Guo, and Dr. Satwik Patnaik for their collaborations in this work and research discussions. Special thanks to Dr. Jiafan Wang, my senior, for his support and valuable suggestions. I would also like to thank Dr. Muhammed Yasin and labmates Rahul Kande, Vasudev Gohil, and Zhaokun Han for all the research discussions I had with them during my stint here. My heartfelt gratitude to my seniors, friends-cum-roommates, and friends in College Station. They have always been by my side during my highest and, most importantly, the lowest point here.

Finally, I am grateful to my mother Paarvathi, father Jayasankaran, and my family members for supporting, trusting, and loving me. They were available to provide guidance any time of the day or night without any expectations. A special thanks to my in-law family, who provided constant support and love. Saving the best for last, I want to thank my husband, Deva for being supportive and caring. He has been more of a friend to me, making the Ph.D. journey more enjoyable.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor Jiang Hu, advisor, of Department of Electrical and Computer Engineering and Department of Computer Science and Engineering, Professor Jeyavijayan (JV) Rajendran, co-advisor, and Professor Samuel Palermo of Department of Electrical and Computer Engineering, and Professor Duncan. M (Hank) Walker of Department of Computer Science and Engineering.

In Chapter 2, the analog simulations to determine the attack results shown in Fig. 2.11 was executed by Dr. Adriana Sanabria-Borbón. She also executed the analog simulations to determine the attack results illustrated in Fig. 3.10 in Chapter 3. All other work conducted for the dissertation was completed by the student independently.

Funding Sources

The research done in this dissertation is funded by National Science Foundation (CNS-1618824, CNS-1828840, CCF-1815583, STARSS-1618797, and SATC CAREER-1822848), Semiconductor Research Corporation (2016-T3S-2688 and 2016-T3S-2689), and Intel. Any opinions, findings, conclusions, and recommendations shared in this dissertation are those of the author and do not necessarily reflect the views of this work's funding sources.

NOMENCLATURE

IC	Integrated Circuits
IP	Intellectual Property
RE	Reverse Engineering
DfTr	Design-for-Trust
AMS	Analog and Mixed-Signal
BPF	Bandpass Filter
LNA	Low-Noise Amplifier
LDO	Low-DropOut voltage regulator
TWG	Triangular Waveform Generator
OTA	Operational Transconductance Amplifier
CS	Common-Source
ADC	Analog to Digital Converter
BIST	Build-In Self-Test
R	Resistor
C	Capacitor
CCM	Configurable Current Mirror
PSR	Power Supply Rejection
ANN	Analog Neural Network
NBTI	Negative Bias Temperature Instability
HCI	Hot Carrier Injection
LUT	Look-Up Table
SMT	Satisfiability Modulo Theories

SAT	Boolean SATisfiability
SFLL	Stripped-Functionality Logic Locking
RLL	Random Logic Locking
FSC	Functionality-Stripped Circuit
PIP	Protected Input Patterns
DIP	Distinguishing Input Pattern
HD	Hamming Distance
PDK	Process Design Kit
FPGA	Field-Programmable Gate Array
PSC	Power Side-Channel
RO	Ring Oscillator
TDC	Time-to-Digital Converter
AES	Advanced Encryption Standard
MTD	Minimum Traces to Disclosure
AWS	Amazon Web Services
CPA	Correlation Power Analysis
SPA	Simple Power Analysis
RSA	Rivest, Shamir, Adleman
LUT	Look-Up Table
CLB	Configurable Logic Block
CEP	Common Evaluation Platform
PDN	Power Distribution Network
SNR	Signal-to-Noise Ratio
VIO	Virtual Input Output
MD5	Merkle–Damgård

SHA256	Secure Hash Algorithm 256
DES3	Data Encryption Standard

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION AND MOTIVATION

The increasing cost of manufacturing of integrated circuits (IC) has forced many companies to go fabless over the years. With the outsourcing of IC fabrication in a globalized/distributed design flow, including multiple (potentially untrusted) entities, the semiconductor industry faces a number of challenging security threats. This fragility in the face of poor state-of-the-art intellectual property (IP) protection has resulted in hardware security vulnerabilities, such as IP piracy, overbuilding, side-channel attacks, counterfeiting, reverse engineering, and hardware Trojans [25]. These vulnerabilities depend on where the attacker is in the supply chain, his/her capabilities, and the resources he/she could access. Each of these vulnerabilities are explained as follows:

- **IP piracy.** The attacker steals and claims ownership of the IP. He/She can be in the foundry having access to the layout, process development kit (PDK) details, and sufficient resources to modify it based on his/her requirements and claim that to be his/her product.
- **Overproduction.** It is a subset of IP piracy, where the attacker has minimal resources sufficient to overproduce the chip but not sufficient enough to change the existing layout and pirate the design.
- **Side-channel attacks.** The attacker extracts the secret information such as a crypto key, by exploiting either the power consumption, timing, or electromagnetic emission of the hardware that implements the cryto algorithm.
- **Counterfeiting.** The attacker takes a used chip, refurbishes it and sells it as a new one.
- **Reverse engineering.** An untrusted end-user purchases a chip, de-packages it, and takes high resolution pictures of each layer of the chip using a scanning electron microscope. The pictures are then processed using an image processing software that helps in annotating the netlist.

- **Hardware Trojans.** Malicious circuits are inserted in the design, whose activation can cause either denial of service, performance degradation, or stealing of secret information such as crypto keys.

To address these issues most effectively at the hardware level [26], a number of hardware design-for-trust (DfTr) techniques, such as IC metering, watermarking, IC camouflaging, split manufacturing, and logic locking [9, 27–31] have been proposed by the researchers. Hence, the hardware security domain involves designing and evaluating defense techniques that protect the ICs from the supply chain vulnerabilities listed above.

This dissertation focuses on the following three pieces of work in the hardware security domain:

1. **AMSlock [6, 32]:** A defense technique to protect analog and mixed-signal (AMS) circuits against overproduction.

2. **Breaking analog locks [33,34]:** An attack technique to evaluate the resilience offered by the various analog-only and AMS locking techniques against supply-chain attacks. This attack uses satisfiability modulo theories (SMT) and Boolean satisfiability (SAT) formulations.

3. **Securing cloud FPGAs against PSC attack.** This work studies the impact of temperature and primitive-level placement of sensor design on the power side-channel (PSC) attack. It also studies the impact of primitive-level placement of crypto design under protection and other logic that resides along with the crypto design under protection.

1.1 Securing analog and mixed-signal (AMS) circuits

Out of the different DfTr techniques available, logic locking has received significant interest from the research community, as it can protect against a potential attacker located anywhere in the IC supply chain. In contrast, other DfTr techniques such as camouflaging or split manufacturing can protect the design only against a limited set of malicious entities. Logic locking inserts additional logic into a circuit, locking the original design with a secret key. For a given input, a

locked design produces the correct output only upon applying the correct key; otherwise, an incorrect output is produced. In addition to the original inputs, a locked circuit has *key inputs*. An on-chip tamper-proof memory drives these *key inputs* [35, 36]. While logic locking schemes are well-defined for digital designs, there is no formal approach for analog designs. In this work, we develop a logic locking scheme for AMS designs. Here, only on applying the correct key, the locked AMS design produces the desired response. Otherwise, for an incorrect key, the response deviates from the desired value. Out of the different digital logic locking techniques available, the stripped-functionality logic locking (SFLL) [9] provides provable security against SAT and removal attacks. Adding to this, it gives freedom to the designer to choose the input patterns to protect. Hence, this technique is used for locking the AMS circuits.

1.2 Evaluation of analog-only and mixed-signal (AMS) locks

There are several analog locking techniques proposed in recent past to combat supply-chain attacks [1–7, 37]. However, there exists no elaborate evaluation procedure to estimate the resilience offered by these techniques. Evaluating analog defenses requires the usage of non-Boolean variables, such as bias current and gain. Hence, in this work, we evaluate the resilience of the analog-only locks and AMS locks using SMT. For the analog-only locks, once the attacker determines the bias current range, the algorithm determines the key required to unlock the circuit. Similarly, knowing the protected input patterns (PIPs), the attacker can determine the key to unlock the AMS locks using SAT. We also propose to extend this attack to break the existing analog camouflaging technique [5].

1.3 Securing cloud FPGAs against power side-channel (PSC) attack

In this work, we address the challenges in the PSC attack and the defenses that thwart it. The two primary challenges in the existing attack techniques are (i) repeatability of the attack and (ii) the need for lower MTD. Also, the existing defense techniques use ring oscillators (ROs) that are blocked by cloud servers. Additionally, they can also crash the FPGAs in few microseconds. Hence, in this work, we study the impact of junction temperature and primitive-level placements of

the sensor design to improve the PSC attack. In addition, we also study the impact of the primitive-level placement of the crypto design under protection and the impact of additional logic residing along with the design under protection on the CPA attack.

2. TOWARDS PROVABLY-SECURE ANALOG AND MIXED-SIGNAL LOCKING AGAINST OVERPRODUCTION*

2.1 Introduction

2.1.1 Motivation

The increasing cost of manufacturing of integrated circuits (IC) has forced many companies to go fabless over the years. With the outsourcing of IC fabrication in a globalized/distributed design flow, including multiple (potentially untrusted) entities, the semiconductor industry is facing a number of challenging security threats. This fragility in the face of poor state-of-the-art intellectual property (IP) protection has resulted in hardware security vulnerabilities, such as IP piracy, overbuilding, reverse engineering, and hardware Trojans [25]. To address these issues most effectively at the hardware level [26], logic locking inserts additional logic into a circuit, locking the original design with a secret key. For a given input, a locked design produces correct output only upon applying the correct key; otherwise, an incorrect output is produced. In addition to the original inputs, a locked circuit has *key inputs*. An on-chip tamper-proof memory drives these *key inputs* [35, 36]. In the case of digital designs, the additional logic may consist of XOR gates [27, 28] or look-up tables (LUTs) [38]. The locked netlist passes through the untrusted design phases. Without the secret key (i) the design details cannot be recovered by reverse-engineering the IC, and (ii) the over-produced IC gives incorrect outputs. A locked IC has to be activated by loading the secret key onto the chip's memory.

While logic locking techniques exist for digital circuits, there is a great dearth of techniques for AMS IP protection. Moreover, analog ICs are not simple although they have less number of transistors. Even with only hundreds of transistors, analog IC design requires highly experienced designers and long time, as analog behaviors are quite complicated. Hence, it involves more cap-

ital in designing analog ICs [39]. Also, as explained in [40], analog ICs rank one in the top five counterfeited parts and cost several million dollars loss. Hence, we focused on developing a provable defense technique to secure AMS circuits. Hence, in this work, we focused on developing a provable defense technique to secure AMS circuits. Out of the different digital logic locking techniques available, the SFLL technique [9] provides provable security against SAT and removal attacks. Adding to this, it gives the freedom to the designer to choose the input patterns to protect. As this locking technique has provable security against different attacks and can successfully lock designs that have more than 100K gates, it is suitable to lock AMS design that consist of digital section (the optimizer circuit) with approximately 50K gates.

2.1.2 Problem statement

While logic locking schemes are well-defined for digital designs, there is no formal approach for analog designs. In this work, we develop a logic locking scheme for AMS designs. Here, only on applying the correct key, the locked AMS design produces the desired response. Otherwise, for an incorrect key, the response deviates from the desired value. For example, in the case of BPF, it exhibits the desired frequency response for the correct key and an incorrect frequency response for an incorrect key.

2.1.3 Prior work on analog locking

A locking technique using memristors is proposed in [3]. It uses a memristor-based voltage divider to bias the bulk terminal of transistors in a differential amplifier. Only the correct key can configure the voltage divider to provide the correct body-bias voltage. This scheme conceptually works well, but its practical applicability is quite restrictive due to its dependence on memristor, and hence, it does not apply to conventional CMOS-based AMS designs. The work [41], proposes a split manufacturing technique for RF circuits. This technique protects the circuit from an attacker in the foundry.

The work [1] demonstrates a satisfiability modulo theories (SMT)-based combinational locking. This defense mechanism ensures that each chip has a unique key. Hence, any attack to make

the chip usable by finding the key is applicable only to that chip. Though this technique has increased the effort of the attack by using SMT-based combinational locking, one disadvantage is applying an incorrect key may sometimes produce close to the desired response. However, in our work, we ensure that the circuit suffers a deterministic error for an incorrect key. Another similar work [2] obfuscates the analog circuit performance using parameter-biasing obfuscation technique. Applying the correct key sets the required transistor width in the current mirror, which in turn provides the suitable bias current for the analog circuit operation.

Similar to our work, in [42], the locked digital circuit mandates the correct key input to set one or more specifications of the analog circuit correctly. This technique is demonstrated on a $\Sigma\Delta$ analog to digital converter (ADC). In [7], both the analog and the digital sections of the AMS circuits are locked. Here, the analog section is locked using the parameter-biasing obfuscation [2], and the digital section is locked using SFLL [9]. Though this technique increases the security by locking both analog and digital sections, it did not convey how to scale up for larger key sizes to thwart brute-force attack. In [4], only on providing the unique key inputs, the trained neural network generates the necessary bias to the analog circuit. This technique cannot protect the analog design against overproduction, cloning, and lock removal attacks. Our technique, however, protects the design against overproduction and lock removal attacks.

2.1.4 Attacks on analog locking

The work in [33] evaluates the resilience offered by the existing analog protection schemes [1, 2], and [3] using SMT. SMT is a decision problem similar to the Boolean satisfiability (SAT) problem. Unlike the SAT, which can only handle Boolean variables, SMT can handle non-Boolean variables. The bias current or voltage range and the locked analog netlist are the inputs to the SMT formulation. This formulation provides the correct key required to unlock the circuit. As the existing analog locks [1–3] are broken by [33], one needs to develop a new defense technique to protect the AMS circuits. Hence, in this work we develop one such technique.

2.1.5 Attacks on digital logic locking

Recent attacks such as SFLL-hd – Unlocked [43] and FALL attack [44] break SFLL-HD^0 and SFLL-HD^h. In combinational logic-locked circuits, the output is a Boolean variable for a given input-key combination. However, in analog circuits, the output is a non-Boolean variable, such as bias current, bias voltage, and frequency response. As these attacks [43, 44] can handle only Boolean variables, it cannot break analog logic locking. Removal attack [45] identifies the protection logic and removes them, thereby extracting the original functionality of the locked circuit. However, launching this attack removes the logic-locked optimizer, which sets the correct value of the passive components in the analog circuit-under-protection. Therefore, this attack does not apply to our proposed work.

2.1.6 Challenges in AMS locking

A simple and obvious approach to lock an AMS design is to insert extra transistors, controlled by key inputs in the analog circuit. These key-transistors can be inserted at random locations in the circuit. On applying the correct key, the analog circuit provides the correct output. However, such a simple approach suffers from the following issues:

- As this includes a minimal number of key-transistors, the attacker determines the correct key by brute-forcing.
- Analog circuits have a smaller number of devices (only a few hundreds). Hence it is relatively simple to reverse engineer than digital circuits, which have millions of transistors on a single chip.
- From the reverse-engineered netlist, the attacker can find the key-transistors by tracking the key inputs and remove them, thereby obtaining the original circuit [46].
- Unlike digital circuits, which have thousands of gates in the circuit-to-be-protected, analog circuits have a few hundreds of components. Thus, one needs to select the best set of components to lock so as to trade-off between overhead and corruptibility.

2.1.7 Proposed approach

Piracy vs. overproduction. In analog designs, most of the commonly-used circuits follow standard layout techniques, such as common-centroid and interdigitization, which makes it easy to reverse engineer [47]. Also, an attacker can always recreate a design from scratch, given the circuit specification; an attacker can obtain such information from the publicly-available datasheet. These challenges make it difficult to prevent piracy attacks, where the attacker can modify the existing design, produce the mask for the modified design, and manufacture new chips. Hence, we try to prevent overproduction, where the foundry uses the same masks and produces excess chips. Our technique renders the overproduced chips non-functional, even if the attacker has access to the complete specification of the target chip.

Our technique for protecting the analog circuit is to logic-lock the digital section of the AMS circuit, as illustrated in Fig. 2.1. This digital section minimizes the effect of process variation by choosing the correct value of passive components in the analog circuit via the tuning knob settings. Analog circuits are susceptible to process variations; for instance, a filter can suffer up to 20% of variation due to the component's tolerances [48]. Many approaches have been proposed to minimize the effect of process variations [49]. In one of these approaches, the passive components of the analog circuits, such as resistors and capacitors, are set to their optimal values using tuning knobs [50]. The digital components determine the optimal values for these tuning knobs. By performing *judicious* logic locking on the digital components of such circuits, only on applying the correct key, the effect of process variations are nullified as the digital circuit works correctly,

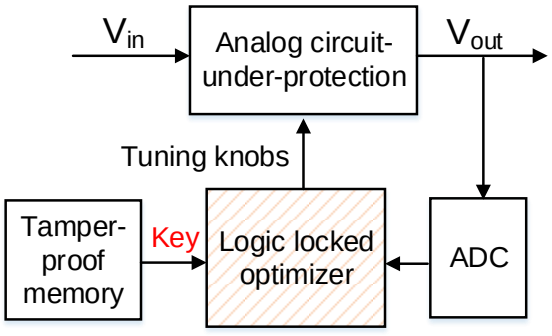

Figure 2.1: Logic locking of the AMS circuit.

and thus making the analog circuit to perform as desired. On applying an incorrect key, the digital circuit produces incorrect output, thereby setting the tuning knobs of the analog circuit to non-optimal values. This improper tuning deteriorates the performance of the analog circuit.

Our approach provides the following benefits:

1. By setting the default tuning knobs where the harmful process variation effect is high, even if the attacker removes the locked digital circuit, the resultant analog circuit has degraded functionality. This degradation is due to the presence of the harmful effect of process variations.

2. The tuning knobs are selected such that even a small amount of change in their values significantly impact the behavior of analog circuits.

3. Since we cannot protect all the input patterns of the digital circuits, we protect only those input patterns that significantly impact the values of the tuning knobs, thereby the output of the analog circuit.

4. Furthermore, we judiciously perform all these steps to minimize area, power, and delay overheads.

2.1.8 Contributions

This chapter has the following contributions:

- The first technique that can protect AMS designs against overproduction using digital logic locking techniques, including the attacks demonstrated in [43–45, 51].

- A sensitivity analysis that can maximize the impact of protection, thereby reducing the overhead. The number of tuning knobs is increased using the same analysis for a higher deterministic error experienced by the attacker for an incorrect key.

- The AMS lock technique proposed in this work applies to a wide variety of analog circuits. It is demonstrated on three different circuits: BPF, LNA, and LDO, including experimental results from a BPF chip.

- The effect of aging on the locked AMS circuits is analyzed. The simulation result proves that the locked circuits are reliable even if the transistors, age over time.

- The effect of SFLL-flex lock on the AMS circuits is analyzed. Results prove that we could achieve higher security with lesser area using SFLL-flex compared to SFLL-HD^0 or SFLL-HD^h.

This chapter is organized as follows. In Section 2.2, we explain the background and previous work related to logic locking and process variations impact on AMS circuits. In Section 2.3, we explain the locking strategy with the BPF circuit as a motivating example. Section 2.4 shows the experimental and simulation results of the proposed technique. Section 2.5 concludes this chapter.

2.2 Background

2.2.1 Logic locking

In this work, we lock the digital section of the AMS circuit using stripped-functionality logic locking (SFLL) [9]. The functionality-stripped circuit (FSC) replaces the original circuit-to-be-protected. The FSC is generated by inserting or replacing a few of the logic gates in the original circuit. The FSC's output is corrupted for those input patterns which are protected by the defender. These patterns are called protected input patterns (PIPs). The output is inverted for the PIP corresponding to the correct key. The restore unit then inverts the inverted output only for the correct key, thereby restoring the correct output. For an incorrect key, SFLL produces an inverted output for the PIP. Both the key and the protected input patterns are the designer's secrets.

There are three variants of SFLL, namely, SFLL-HD^0, SFLL-HD^h, and SFLL-flex [9]. Depending on the variant of SFLL, the restore unit implements one of the techniques given below. The corruption injected by the inversion logic is restored, when

1. the Hamming distance (HD) between the external key (k) and the input pattern equals 0 in SFLL-HD^0.

2. the HD between k and the input pattern equals h in SFLL-HD^h.

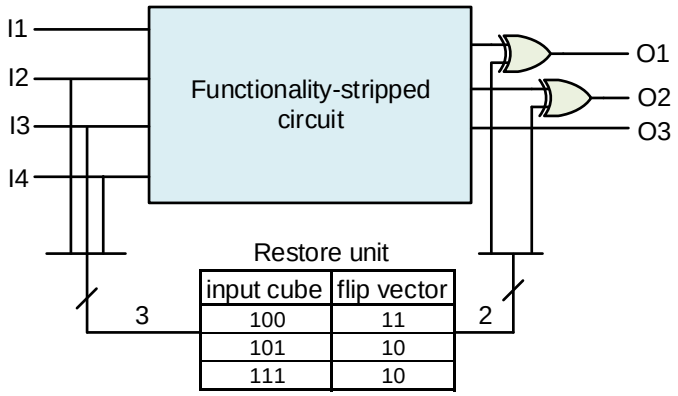

Figure 2.2: Stripped-functionality locking locking (SFLL)-flex [9].

3. the PIPs or "cubes" are stored in a content-addressable memory along with their corresponding flip vectors, as illustrated in Fig. 2.2. The flip vector associated with each protected cube holds information regarding which outputs are to be flipped (restored) for that cube. When the input to the FSC is equal to one of the protected cubes in the content-addressable memory, the corresponding flip-vector is retrieved and XORed with the outputs to restore the original functionality.

The choice of the input patterns to protect is not restricted by the value of HD in SFLL-flex. Hence, the defender can protect any IP-critical input patterns. In the context of this research, one needs to select what input patterns of the optimizer, that compensate the process variations effects. This way, an incorrect key does not eliminate the effect of process variations.

2.2.2 Analog ICs and process variations

The performance of analog circuits is degraded in the presence of process variations. During the design phase, the sizes of transistors and passive components are chosen to meet the required specifications. Being aware of the process variations, the designer performs Monte Carlo and/or corner simulations to tune these sizes to improve the robustness of the design. However, due to the limitations in the fabrication process, it is not possible to fabricate the precise sizes of the transistors and passive components. Adding to this, the intra-die process variations in the manufactured chip have an impact on the circuit's performance. Hence, to nullify the effect of process variations,

researchers have proposed different techniques, such as body voltage tuning and built-in self-test (BIST) optimization techniques. Body-bias voltage tuning is an efficient way to address process variation in terms of power, performance, and area [52].

Likewise, in the BIST technique [50] at start-up, the tuning knobs are in their default settings. The input voltage is applied to the analog circuit, and the corresponding output response is digitized by ADC. This digitized data is sent to the optimizer. The optimizer calculates the deviation in the actual output response from the ideal characteristics. If the magnitude of this difference is high, it indicates that the circuit's response is far from the ideal characteristics and, if low, indicates it is closer to the ideal characteristics. The optimizer chooses a different tuning knob setting such that the cost function calculated is lesser than the cost function value corresponding to the previous settings. This process is iterated until the output response for the chosen tuning knob gives zero cost function value. This tuning process helps in compensating the process variation impact on the fabricated components. The optimizer uses the simulated annealing algorithm to determine the tuning knob settings. The performance of this tuning depends on various factors, such as temperature step size, the maximum number of iterations, and initial temperature.

The body-bias tuning can compensate for process variations only in the transistors but not other passive components, such as resistors and capacitors. Therefore, they cannot be deployed to analog circuits that require tuning of their passive components. Also, as there is no secure analog locking scheme [33], that can lock the supply voltage used by the body-bias tuning technique. Hence, we chose the optimization technique [50] (i) to compensate for process variations in passive components and bias currents and (ii) the digital optimizer can be locked using a provably secure digital locking technique [9]. Our technique is power-, performance-, and area-efficient and can be used in wearable devices and IoT architectures. The wearable IoT ECG sensors [53] consists of AMS circuits showing the possibility of implementing our technique in wearable devices. Also, any IoT SoC has multiple analog modules such as audio units, radio units, sensors, and power management units [54]. Here, a single optimizer can be used to tune all the modules during power-up, justifying the area overhead of the optimization unit. Hence, to show the proof-of-concept

of our technique, we illustrate an optimizer controlling one analog circuit. However, in practical scenarios, a single optimizer can control more number of analog blocks.

2.3 Locking approach for AMS circuit

2.3.1 Choosing analog circuits to demonstrate our locking approach

We have chosen BPF, LNA, and LDO circuits to protect as they are widely used in many domains, such as IoT, communication, and signal processing. The filter circuits, such as BPF, lowpass filter, and highpass filter, are essential in many communication and signal processing systems [55]. The LNA is highly prevalent in RF communication [56, 57], and the LDO voltage regulator is a common entity in the power management unit [58]. Also, the analog circuits considered in this work are used in the individual blocks of the IoT SoC architectures [54]. In this section, we first describe the BPF circuit, which we use as a motivating example to explain our idea. We then describe our locking architecture, and the methodology used in selecting the input patterns to protect. Finally, we explain how the proposed mechanism applies to other analog circuits, such as LNA and LDO.

2.3.2 Motivational example: Bandpass filter

Consider the Tow-Thomas filter illustrated in Fig. 2.3 with a transfer function defined by Equation (2.1).

$$H_{BP}(s) = \frac{s/(R_1C)}{s^2 + s/(R_1C) + 1/(R_2^2C^2)} \tag{2.1}$$

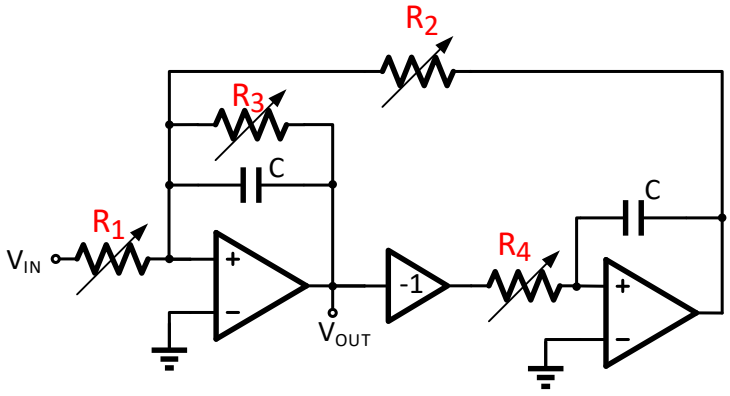

Figure 2.3: Tow-Thomas BPF circuit. The resistors R_1, R_2, R_3, and R_4 are tunable resistors.

Assuming ideal amplifiers, i.e., amplifiers with infinite gain and bandwidth, we set $R_1 = R_3$ and $R_2 = R_4$. The filter characteristics, such as center frequency $f_o = 1/2\pi R_2 C$ and quality factor $Q = R_1/R_2$ are defined by the passive components in the circuit, i.e., resistors and capacitors. Each of the R (C) is replaced by arrays of Rs (Cs) to enable tuning that helps in addressing the impact of process variation. The value of R (C) is tuned to get the optimal circuit performance in the presence of process variation. Such tuning helps to compensate for any changes in resistor and capacitor values due to process variations.

The required tuning resolution controls the size of each passive component in the array. This resolution is defined as the minimum increase or decrease in the value of the passive component between two consecutive tuning knob settings. The resolution of the tuning is determined based on the performance of the optimization technique, the area incurred by this technique, and the power consumption. For example, a very low resolution in tuning (corresponding to bigger component sizes) results in a sub-optimal operating point after optimization. However, a high resolution in tuning (corresponding to smaller component sizes) results in an optimal operating point. Nevertheless, this incurs a large area and higher power consumption. Hence, there is a trade-off between the optimization quality, area, and power consumption in choosing the resolution. In BPF, the tuning knob controls the resistor values, which in turn controls the output response of the BPF. Each resistor in the array has a resistance of 865.38Ω.

The passive components, such as Rs and Cs, are made tunable to compensate for process variations. Hence, the number of these components required is controlled by the maximum variation in the circuit parameters. This variation in the circuit parameters is due to the impact of process variations. In BPF, the variations in the circuit parameters such as center frequency and bandwidth are estimated using Monte Carlo simulations. From these simulations, the defender determines the minimum and maximum value of the filter parameters. Using this range, he/she calculates the minimum and maximum values of the Rs and Cs required. This information gives the range of the value of the passive components that has to be implemented in the array.

2.3.3 Locking architecture

The use of SFLL techniques to lock the AMS circuits does not correspond to a plug-and-play concept. Rather, it involves multiple steps and analyses for successfully locking the digital optimizer, which controls the performance of the analog circuit. The locking architecture consists of the following steps given below:

1. **Choosing the tunable components using sensitivity analysis.** To ensure that the attacker suffers maximum degradation in the performance on applying an incorrect key, we perform a sensitivity analysis to determine those passive components on which the output response of the analog circuit is highly dependent.

2. **Replacing the chosen component with an array of components.** To make the chosen component tunable, replace it with an array of components. The required value of the component is chosen using the tuning knobs.

3. **Determining all possible input patterns to the optimizer.** Simulate the analog circuit-under-protection for each tuning knob setting. The input and output values are determined at the frequency points of interest, depending on the circuit-under-protection.

4. **Determining the cost function corresponding to all the input patterns.** The defender calculates the value of the cost function for each possible input pattern. This value determines if the output response of the circuit-under-protection follows the ideal characteristics of the circuit.

5. **Choosing the input patterns to protect.** Select the minimum cost the attacker should encounter for an incorrect key. Choose all the input patterns that produce the cost equal to or below the selected cost. These patterns are the PIPs.

6. **Locking the optimizer.** Using the PIPs selected in the previous step, lock the optimizer using SFLL-HD^0, SFLL-HD^h, or SFLL-flex.

Hence, the analog circuit-under-protection, along with the locked optimizer, is the functionality-stripped AMS circuit. Only on applying the correct key, the original functionality of the circuit is recovered. The following sections explain the locking architecture in detail.

The AMS design in Fig. 2.1 consists of the BPF circuit-to-be-protected along with the ADC and the logic-locked optimizer. The voltage input and the two tuning knobs from the optimizer are the inputs to the BPF. Each tuning knob setting corresponds to a unique value of the resistor in the BPF circuit, which in turn impacts its frequency response. During the start-up, the tuning knobs are in their default settings. For the given input voltage, the output response of the BPF is digitized by ADC and sent to the logic-locked optimizer. The secret key required for the proper operation of the optimizer is loaded from a tamper-proof memory. The optimizer calculates the cost difference in the measured and the desired output response of the BPF. If the magnitude of this difference is high, it indicates that the BPF response has deviated from the desired response and if low, indicates it is more close to the desired response. The deviation in the output response from the desired response is calculated based on the following equation.

$$\begin{aligned} CF =&(G_{f_2} - (\sqrt{2} \times G_{f_1})) + (G_{f_2} - G_{f_3}) + \\ &(G_{f_3} - (\sqrt{2} \times G_{f_4})) + (G_{f_1} - G_{f_4}) \end{aligned} \tag{2.2}$$

Here, CF is the cost function, f_1 and f_4 are the lower and upper cut-off frequencies, and f_2 and f_3 are the center frequencies ($f_2 = f_3$). G_{f_1}, G_{f_2}, G_{f_3}, and G_{f_4} are gain at f_1, f_2, f_3, and f_4, respectively. The G_{f_2} should be equal to $\sqrt{2} \times G_{f_1}$ and the G_{f_3} should be equal to $\sqrt{2} \times G_{f_4}$ in the ideal characteristics. Likewise, G_{f_2} and G_{f_3} should be equal, and G_{f_1} and G_{f_4} should be equal in an ideal characteristic. The gain is calculated by the ratio of the output voltage to the input voltage. The transient analysis is performed at these four points. There is a total of 18, 10-bit data measured from the input signal and the output response of the BPF. It corresponds to the 180-bit data from the analog circuit. This data is concatenated with the 40-bit control input that configures the optimizer. This concatenated data constitutes the 220-bit data that is fed to the optimizer. As our optimizer is implemented using a combinational logic, we provide all the 18, 10-bit inputs

driven by the ADC along with the 40-bit optimizer configuration input at the same time. These inputs together constitute the 220-bit input to the locked optimizer.

This simple architecture suffers from two challenges:

Issue 1: Not all resistors and capacitors in the BPF are made tunable. Making every component tunable increases the area overhead of the analog circuit. Also, the optimizer now has to tune all the components to address the process variations. Consequently, the area and delay overheads of the optimizer are increased. Thus, one needs to judiciously choose the parameters to tune.

Issue 2: The logic-locking techniques can protect only a limited number of input patterns. In SFLL-fault [59], the input patterns to be protected are chosen based on the VLSI testability metrics such as controllability and observability. This ensures that for an incorrect key, the maximum number of output bits are corrupted. However, in our work, we want to protect a specific set of input patterns, which sets the tuning knobs to optimal values. Hence, we use the SFLL-flex, where the user has the freedom to choose specific input patterns to protect. Also, as increasing the number of patterns protected increases the area overhead, we need to be diligent in choosing the patterns protected.

2.3.4 Sensitivity analysis to solve Issue 1

To ensure that the attacker suffers maximum degradation in the performance on applying an incorrect key, we perform a sensitivity analysis to determine those passive components on which the output response of the analog circuit is highly dependent. The sensitivity is a measure of the variation in a performance metric, such as f_o and Q, due to the change in certain circuit parameters [60]. The normalized sensitivity of the chosen metric p_i with respect to the change in parameter x_j is represented by Equation (2.3). This equation helps in determining the sensitivity of the circuit's response to each of the component considered.

$$S_{x_j}^{p_i} = \frac{x_j}{p_i}\frac{\partial p_i}{\partial x_j} \tag{2.3}$$

There is no difference in selecting the tuning knobs (tunable components) in secured and unsecured implementations. In both these implementations, sensitivity analysis is used to determine the components over which the output response of the circuit-under-protection is highly dependent. This choice ensures that for an incorrect key, the incorrect value of the parameters is chosen. Hence, there is maximum deviation in the output response. The following steps are carried out to perform the sensitivity analysis:

1. The sensitivity of the circuit's response with respect to the circuit parameters is plotted.

2. The component for which the circuit response has the highest sensitivity is chosen. An array of components replaces this component.

3. The optimal value of this component is chosen from the array by the tuning knobs controlled by the optimizer.

For example, in BPF, the center frequency and the quality factor are the circuit metrics on which the sensitivity analysis is performed. The selected tuning knobs are the resistors R_1 and R_2. The optimal value of the component varies from chip to chip due to process variations to achieve the same performance metric. Hence, the defender makes the single component tunable by replacing it with an array of components. Once the chip is manufactured, the component that has the optimal value is chosen, such that the process variation impact is compensated. The defender uses the tunable components only to compensate for the process variation and does not use it to have flexibility in the performance metrics.

The effect of an incorrect key on the locked circuit can be increased by tuning more than two passive components. Let the sensitivity of the metric p_i be calculated based on the chosen passive components. Let j be the total number of components considered for sensitivity analysis. The change in one of the passive components does not affect the other. Hence, the change in p_i, such as transfer function and transconductance, due to change in one of the passive components does not depend upon the change in p_i due to another passive component. Therefore, the total change in p_i due to the changes in more than one passive component is the sum of the partial

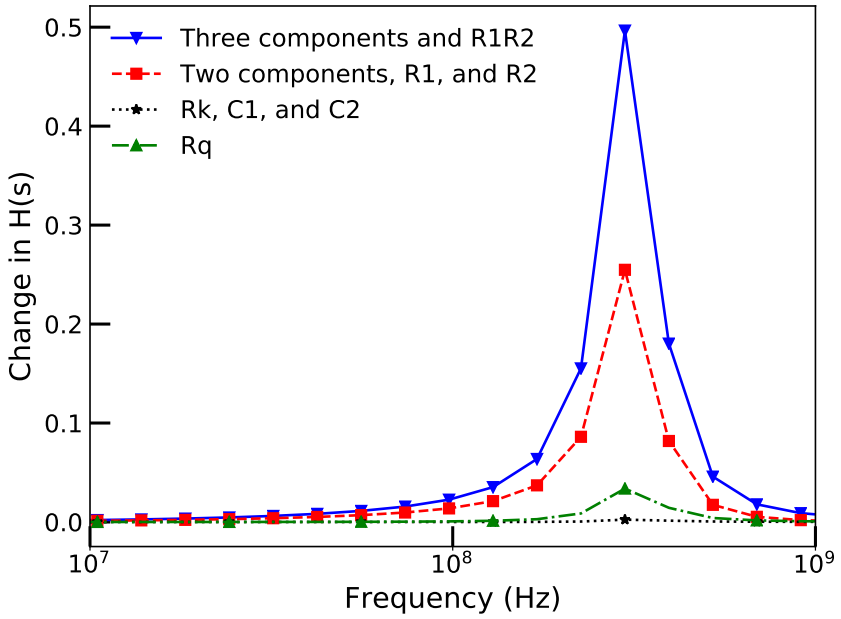

Figure 2.4: Change in $\mathbf{H(s)}$ with respect to the change in component values.

derivative p_i with respect to the passive components considered, i.e, $dp_i = dp_i(x_1, x_2, \cdots, x_j) = \frac{\partial p_i}{\partial x_1}dx_1 + \frac{\partial p_i}{\partial x_2}dx_2 + \cdots + \frac{\partial p_i}{\partial x_j}dx_j$. Thus,

$$\frac{dp_i}{p_i} = S_{x_1}^{p_i}\left(\frac{dx_1}{x_1}\right) + S_{x_2}^{p_i}\left(\frac{dx_2}{x_2}\right) + \cdots + S_{x_j}^{p_i}\left(\frac{dx_j}{x_j}\right) \tag{2.4}$$

The BPF circuit considered in this example has six passive components, as illustrated in Fig. 2.3. The change in the output response of BPF, $H(s)$ with respect to the change in the passive components are measured using Equation 2.4. The change in $H(s)$ is plotted for the following cases: (i) all possible combinations of three passive components are changed, (ii) all possible combinations of two passive components changed, and (iii) one of the passive components is changed. Fig. 2.4 shows the change in $H(s)$ of the BPF due to the change in one or more components. As illustrated in the figure, change in $H(s)$ is highest when all possible combinations of three passive components change and also when one of the combinations of two passive components (R_1 and R_2) change.

2.3.5 Choosing input patterns to solve Issue 2

Based on the minimum deviation in the output response the attacker has to encounter for an incorrect key, the designer protects all those input patterns that correspond to a deviation less than the chosen deviation. This deviation is quantified by the error in the cost function. If the error is close to zero, the output response is close to the ideal characteristics. Otherwise, it deviates from the ideal characteristics. The input patterns to the locked-optimizer are the digitized voltage values of the output response of the analog circuit-under-protection via the ADC. The defender can simulate the analog circuit, for example, the BPF, for each tuning knob settings, to determine the output response for each of these settings. The in-phase and quadrature-phase values of the input and output voltages are measured via transient analysis at the lower cut-off, upper cut-off, and two center frequencies. The voltage values calculated via simulations may differ from the chip results due to the PVT variation effects. These values can differ by a small percentage, thereby changing the LSBs of the measured voltages. Hence, the designer considers the MSBs of the voltage values in the locking, and the LSBs are ignored. Using these values, the designer can calculate the cost for each tuning knob setting. He/She will then choose those input patterns which corresponds to the least cost as protected input patterns.

In the case of SFLL-HD^0, since only one input pattern can be protected, a designer can obviously select the input pattern of the optimizer that results in the minimum cost function. For instance, in the case of BPF, we need to protect the input pattern corresponding to resistor settings $R_1 = 27.68K\Omega$ and $R_2 = 10.38K\Omega$. This setting ensures minimum cost, as illustrated in Fig. 2.5. In other words, this is the input pattern, for which the error between the desired response and the actual response of the BPF is minimum. In the case of SFLL-HD^h, a designer can increase the number of PIPs by increasing the Hamming distance (h). However, this decreases the security level of an n-bit design to $2^{n-k} \times \binom{k}{h}$, where k is the key size. Hence, one can increase the value of h only to an extent. Here, we select those input patterns such that they are at an $HD = h$ away from the one that produces the minimum cost function.

However, as SFLL-flex is devoid of the HD restriction, the defender can choose any input

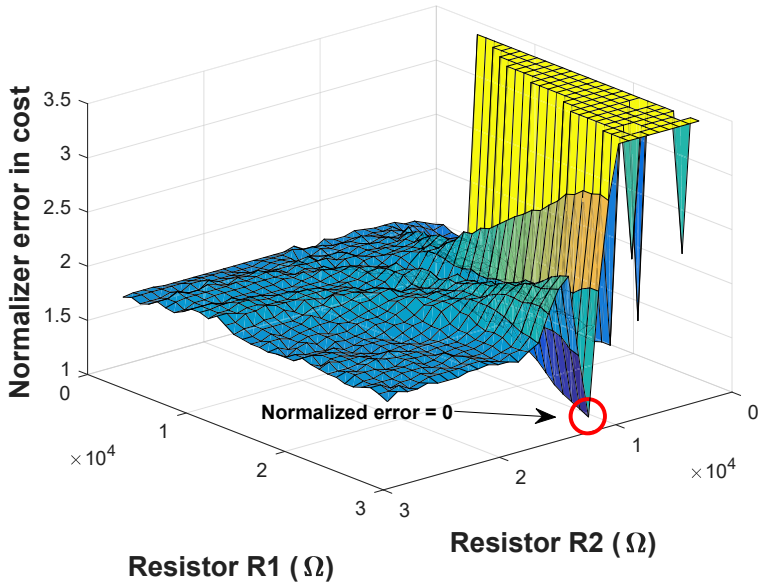

Figure 2.5: Normalized error on the output of BPF for different tuning knobs. The normalized error at $(\mathbf{27.68K\Omega}, \mathbf{10.38K\Omega}) = \mathbf{0}$. The data is collected from a IBM-180nm process BPF chip described in Section 2.4.

patterns he/she want to protect. Also, as the FALL attack [44] and SFLL-hd – Unlocked [43] have broken SFLL-HD^0 and HD^h without the use of an oracle, the optimizer is locked using SFLL-flex as it is resilient against these attacks. The following section explains the locking process using the SFLL-flex technique. The optimizer controls two tuning knobs. Each knob has a 5-bit input. They tune the resistors R_1 and R_2 of the BPF circuit. Though the optimizer's input size is 220 bits, as the total number of tuning knob settings equals 1024, the effective number of input patterns to consider reduces to 1024 from 2^{220}. The optimizer is designed to choose the tuning knob settings, which gives the minimum cost. Therefore, the defender can choose PIPs, whose corresponding costs are low.

2.3.6 Extending to other AMS circuits

This locking technique is illustrated over two other analog circuits–LNA and LDO.

2.3.6.1 LNA

A common-gate topology-based LNA was tested as a study case [57]. The specifications to optimize are the gain (S_{21}) and the input matching (S_{11}) for the given resonance frequency. Based on

sensitivity analysis, the tuning knobs are determined to be the biasing current and the capacitance of the load tank. The minimum bias current is given by $10\mu A$, and the maximum bias current is $55\mu A$. The minimum increment in the bias current is $5\mu A$. The metrics S_{11} and S_{21} are estimated by applying two frequency tones at $f_R \pm \Delta_f$ and connecting the proper matching at the input and output. Then, the signal's amplitude is measured at the input and output of the LNA.

2.3.6.2 LDO voltage regulator

A capless LDO with a PMOS pass transistor and a single-stage error amplifier is tested [61]. The performance metrics to optimize are the power supply rejection (PSR) and the phase margin. The selected tuning knobs are the biasing current of the error amplifier and the compensation capacitor. The minimum bias current is given by $10\mu A$, and the maximum bias current is $60\mu A$. The minimum increment in the bias current is $10\mu A$.

2.3.6.3 Extending to large scale AMS circuits

Though the analog section of the example AMS circuits (BPF, LNA, and LDO) has only a few tens of transistors, the digital section (optimizer) consists of around 50K gates. These AMS circuits are generally a part of bigger analog circuits, such as receivers and phase-locked loops [62, 63]. Hence, for an incorrect key, the degraded performance of the AMS circuit has an impact on the overall performance of the receiver and phase-locked loops.

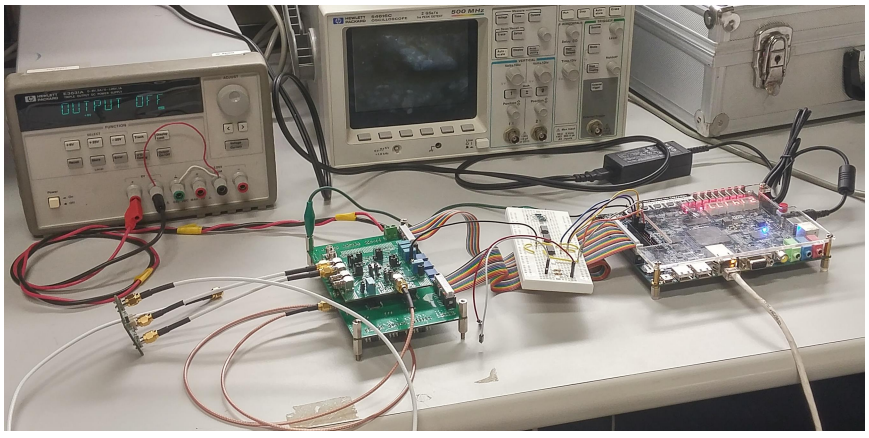

Figure 2.6: Measurement setup of the setting the tuning knobs of the BPF circuit.

2.4 Results

2.4.1 Experimental setup

We demonstrate our analog locking technique on three different AMS circuits: BPF, LNA, and LDO. The specifications for each of these circuits are as follows. The BPF has a center frequency $f_c = 74MHz$ and $BW = 13MHz$. The input size of the optimizer is 220 bits. This input is fed with the digitized output (frequency response) of the BPF. The specifications of the LNA circuit are $S_{21} > 20dB$ and $S_{11} < -20dB$ at a resonance frequency $f_R = 6GHz$, with input size to the optimizer equal to 154 bits. Similarly, the LDO's specifications are $PSR \leq -50dB$ and a phase margin larger than 45° with optimizer input size equal to 234 bits. The experiments are executed on 40, 10-core Intel Xeon processors running at 2.8GHz with 256 GB of RAM. The designs are synthesized using Synopsys Design Compiler tool using Nangate 45nm open cell library [64].

Measurement setup: Fig. 2.6 shows the printed circuit board containing the BPF chip fabricated using the IBM-180nm process. The output response of the BPF for each tuning knob setting is collected from this chip. The optimizer is implemented on an FPGA, and a dual voltage source is used as supply. **Measurement setup for aging analysis.** For analyzing the impact of aging on BPF, V_{th} is increased for the PMOS and NMOS transistors to model the NBTI and HCI effects, respectively, caused due to aging.

The in-phase and the quadrature-phase values of the input and output voltages are measured at the center, lower, and upper cut-off frequencies. The experiment is repeated for all possible tuning knob settings, and the voltage values are logged for each simulation run. The fresh and the aged analog circuits are simulated by applying the correct key to the optimizer. Fig. 2.7 shows the degradation in the center frequency and gain at the center frequency as the circuit ages. The center frequency and the gain of BPF are designed to be $18.84MHz$ and $3.1dB$, respectively. Also, as illustrated in the figure, as the circuit ages, the output response deviates marginally from the original response. The binary equivalent of the voltage values is analyzed to determine the bit positions to lock, as mentioned in Section 2.3.5. As indicated by Fig. 2.7, the output responses

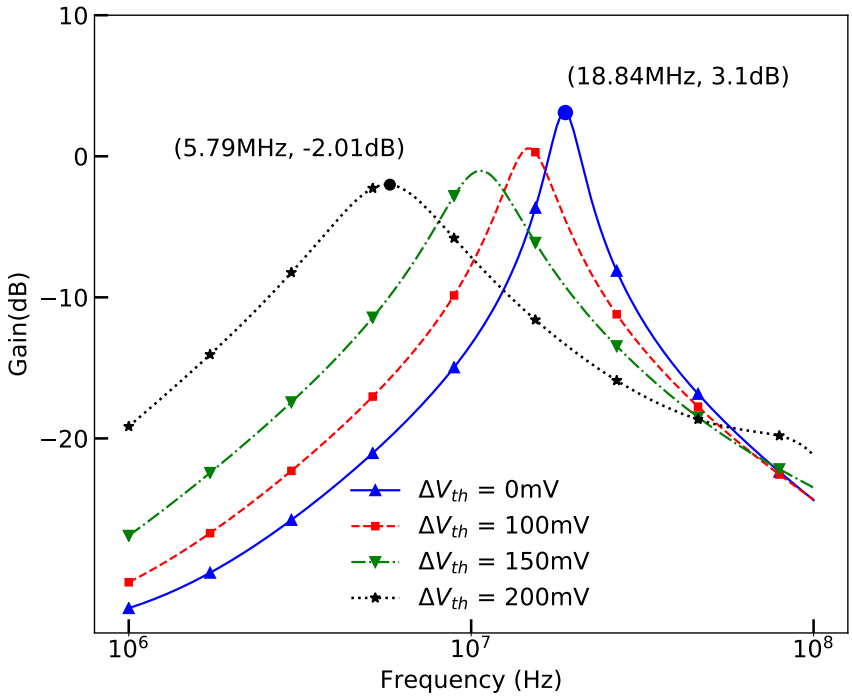

Figure 2.7: The change in output response of BPF due to aging. The BPF is designed for a center frequency of $\mathbf{18.84MHz}$ and gain of $\mathbf{3.1dB}$. As the circuit gets aged, the center frequency and the gain reduce to $\mathbf{5.79MHz}$ and $\mathbf{-2.01dB}$, respectively.

vary marginally, i.e., the MSBs are constant, but the LSBs vary between the measurements before and after the aging of the BPF. Hence, a designer needs to consider the MSB positions for locking and the LSBs as don't cares.

2.4.2 Effect of tuning knobs on BPF's output

For a locking technique on the BPF circuit to be effective, any deviation in the tuning knob values from its ideal set of values should degrade the BPF's response. The normalized error value can quantify this effect. Fig. 2.5 shows the normalized error value for different tuning knob values. As one can see, when the (R1,R2) values are (27.68KΩ, 10.38KΩ), the normalized error value is zero, and for all the other cases, there is a non-zero error. Thus, only on setting the tuning knobs to the correct values, the desired response is obtained. Any deviation from these correct values indicates an incorrect response from the BPF circuit.

Table 2.1: Effect of SFLL-HD^0 and SFLL-HD^h logic-locking techniques on the optimizer circuits of BPF, LNA, and LDO. In the case of BPF, the data is collected from the IBM-180nm process BPF chip described in Section 4. The correct values of the tuning knobs for BPF are $(\mathbf{27.68K\Omega}, \mathbf{10.38K\Omega})$. We used simulation results for LNA and LDO. % error listed is the minimum error on applying any incorrect key.

Analog Circuit	**# of inputs**	**Key size**	**Security level (s)**	**# of patterns protected**	**% error (min.)**	**Key size**	**Hamming distance (h)**	**Security level (s)**	**# of patterns protected**	**% error (min.)**
Bandpass filter	220	220	220	1	8.11	220	1	212	220	8.11
	220	112	112	3.25×10^{32}	44.59	220	20	126	1.19×10^{28}	8.11
	220	87	87	1.09×10^{40}	72.97	220	37	80	1.37×10^{42}	8.11
Low noise amplifier	154	154	154	1	0	154	1	146	154	0
	154	84	84	1.18×10^{21}	3100	154	9	107	1.06×10^{14}	0
	154	81	81	9.44×10^{21}	3100	154	17	80	1.73×10^{22}	0
Low-dropout regulator	234	234	234	1	0.7	234	1	226	234	0.7
	234	135	135	6.34×10^{29}	12.59	234	20	138	4.32×10^{28}	0.7
	234	109	109	4.25×10^{37}	39.58	234	41	81	9.89×10^{45}	0.7

Table 2.2: Impact of aging on the locked AMS circuits using SFLL-flex. The error in the output response for a correct key, for all the cases is 0. The maximum possible error is 255.

Analog Circuit	Input size	Key size	Security level	Area %	% Error before aging	% Error after aging
Bandpass filter	220	80	219	0.14	50	50
	220	220	210	171.30	50	0
	220	220	210	107.53	39.21	0
	220	220	211	70.10	27.45	0
Low noise amplifier	154	84	153	0.13	50	50
	154	154	144	92.46	50	0
	154	154	144	72.01	39.21	0
	154	154	145	46.72	28.23	0
Low-dropout votlage regulator	240	122	240	0.14	50	50
	240	240	230	116.97	50	0
	240	240	231	60.72	39.60	0
	240	240	232	27.03	27.84	0

2.4.3 Effect of logic locking on tuning knob

The following section illustrates the impact of different SFLL locking on the AMS circuits. **SFLL-HD^0 and SFLL-HD^h**. Table 2.1 lists the effect of SFLL-HD^0 and SFLL-HD^h logic locking techniques on the optimizer circuits of BPF, LNA, and LDO. In the case of SFLL-HD^0, when the key size equals the input size of the optimizer circuit, $k = n = 220$, the HD $h = 0$, and only one input pattern can be protected. Based on the normalized error in Fig. 2.5, we choose to protect the pattern that results in the minimum error. Hence, the input pattern resulting in the optimal tuning knob values, i.e., $(27.68K\Omega, 10.38K\Omega)$ is protected; in this case, the normalized error is 0%. For an incorrect key, the optimizer sets the tuning knob that results in a normalized error value of at least 8.11%. Though the security level (s) achieved by this approach is 220, the normalized error value is only 8.11%.

One approach to increase the error value is to protect more number of inputs patterns. This increase in the error can be achieved by reducing the key size. For instance, for SFLL-HD^0 and a key size of $k = 112$, the number of input patterns protected is 3.25×10^{32}, increasing the normalized error to 44.59%. Similarly, by choosing a key size of $k = 87$, a normalized error value is increased to 72.97%. However, one cannot reduce the key size below 80 bits, because this reduces the s

and hence, the search space to less than 2^{80}, making it vulnerable to SAT and brute-force attacks. Another approach to increase the number of protected input patterns and hence, the normalized error value is to use SFLL-HD^h, whose results in Table 2.1.

In case of LNA, SFLL-HD^0 and SFLL-HD^h achieve the normalized error rate of 3100% and 0%, respectively. For LDO, SFLL-HD^0 and SFLL-HD^h obtain the normalized error rate of 39.58% and 0.7%, respectively. As one can see, for the same key size, SFLL-HD^h protects more input patterns compared to SFLL-HD^0. For instance, in case of BPF, for a key size of 220, SFLL-HD^0 protects only one input pattern, whereas SFLL-HD^h, for $h = 37$, protects 1.37×10^{42}. However, the normalized error rate is still the same (i.e., 8.11%) or even lesser (i.e., for LNA it is 0%). This is because SFLL-HD^h requires all the protected input patterns to have the same HD from the key with key size equal to the input size. The probability of all the patterns protected having the same h is very small. This indicates that SFLL-HD^0 results in a higher error than SFLL-HD^h.

SFLL-flex. Table 2.2 lists the effect of SFLL-flex on the AMS circuit. The input to the optimizer is 220 bits. The output of the optimizer is a set of two tuning knobs, where each of them is 5 bits. The input to the optimizer corresponds to the BPF frequency response measurements for the particular tuning knob settings. As the effective size of the tuning knobs are 10 bits, there are 1024 possible settings for the tuning knobs. Hence, there are only 1024 unique BPF responses. Though there is a possibility of 2^{220} input patterns to the optimizer, as the input size is 220 bits, the input depends only on the tuning knob settings. Hence, there are only 1024 unique input patterns to the optimizer. As the SFLL-flex has the flexibility to choose the input patterns to protect, we can protect all the 1024 patterns, as shown in Table 2.2. However, this also incurs a huge area. As it is not necessary to protect the input patterns for which the cost is maximum (BPF's frequency response deviated from the original response), we could ignore these patterns from protecting. Based on the minimum error the attacker has to encounter for an incorrect key, the corresponding input patterns can be protected, thereby reducing the area overhead. Also, the security level (s) achieved for all the cases is more than 80, ensuring resiliency against the SAT attack.

The defender protects all the input patterns that correspond to a deviation less than the chosen

deviation. Hence, when the attacker supplies an incorrect key, he/she encounters a deviation in the output response. This deviation is equal to or more than the deviation chosen by the defender. The error in the cost function quantifies this deviation, which is tabulated in Table 2.2. In a few cases, for the same input size, key size, and s achieved, the area overhead value differs. For example, in BPF, for two setups the input size and key size is equal to 220 and s is equal to 210. However, these setups have different area overheads (107.53% and 171.30%). This is because the overhead depends on the number of protected input patterns (c). Also, s is calculated using the SAT resilience formula for SFLL-flex given by $k - \lceil \log_2 c \rceil$. As s is dependent on $\log_2 c$, its value does not change for a marginal increase in c. Adding to this, the impact of aging on the AMS circuit is studied. The deviation in the output response (error in cost function) when an incorrect key is applied to an aged AMS circuit (analog circuit-under-protection and the logic-locked optimizer) is added to Table 2.2. Fig. 2.8.(a) shows the impact of an incorrect key on the optimizer locked using SFLL-flex when the PIPs whose cost function value is (i) less than 70, (ii) less than 100, and (iii) less than 255. The cost function depicts the deviation of the BPF response from the expected one. Hence, as the cost function increases, the deviation of the BPF output from the expected response also increases, as illustrated in Fig. 2.8. Initially, SFLL-HD^0 and SFLL-HD^h were used to lock the optimizer as it does not require an expensive restore unit, where all the protected input patterns are stored. However, by judiciously choosing the input pattern to protect and the key size, both security and lesser area overhead using SFLL-flex is achieved. Also, as shown in Fig. 2.8, there is no degradation in the security even after the circuit being aged.

2.4.4 Security analysis

The following section shows the resiliency offered by the locked optimizer.

Resiliency against SAT. The resiliency against SAT attack offered by SFLL-HD^0 and SFLL-HD^h are k and $k - log_2\binom{k}{h}$, respectively [9]. From Fig. 2.9(a), we can infer that security level s achieved for the BPF is the maximum when $h = 0$ or $h = 220$ and the minimum when $h = 110$. To ensure that the locked circuit is SAT attack resilient, we need to choose h and k such that the security level is greater than 80. Hence, the allowable h values can be $0 \leq h \leq 37$ or $183 \leq h \leq 220$, and the

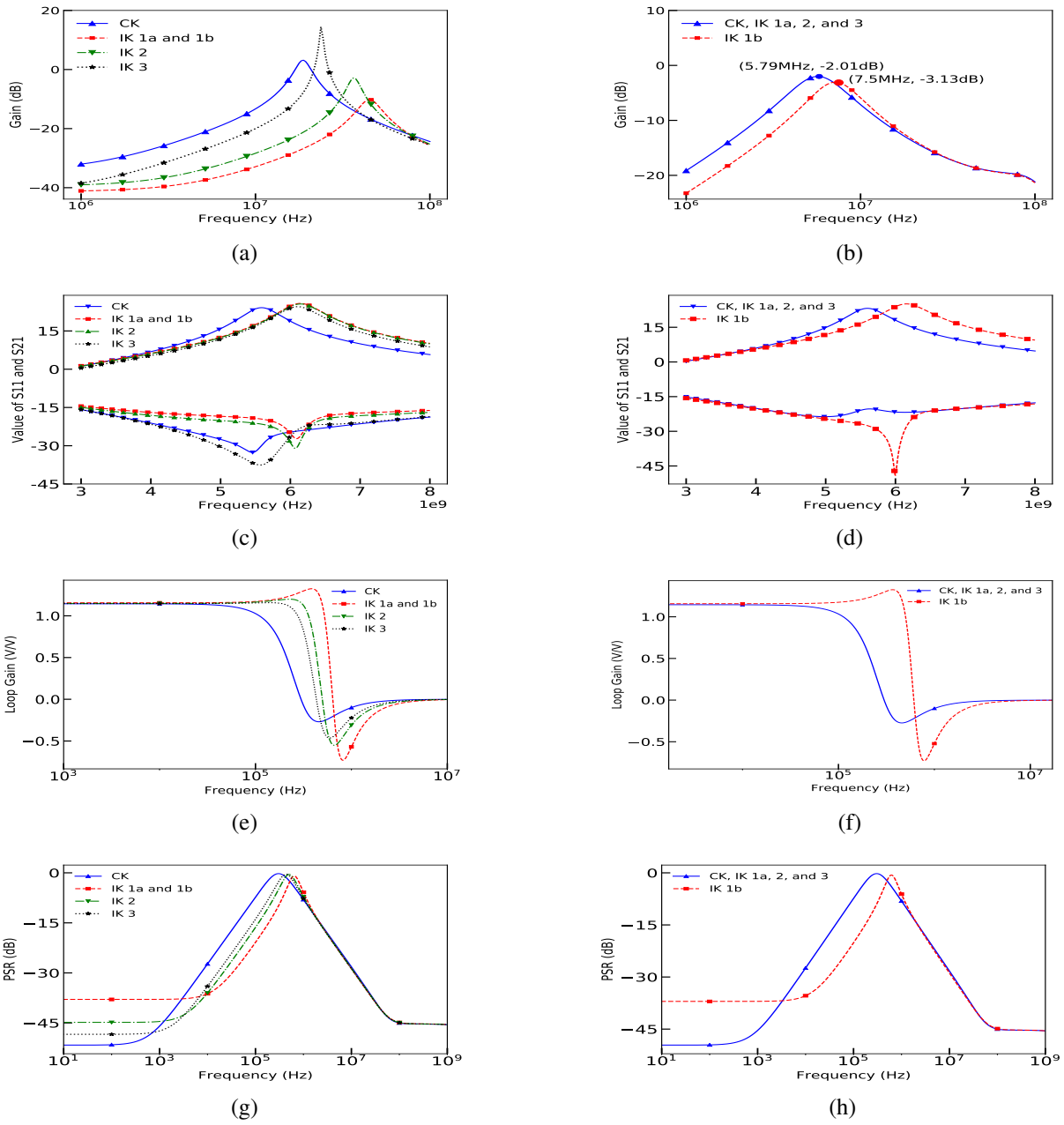

Figure 2.8: The impact of aged analog circuit on SFLL locked-optimizer. (a) Impact on the frequency response of new BPF circuit. (b) Impact on the frequency response of aged BPF circuit. (c) Impact on the S11 and S21 parameters of the new LNA circuit. (d) Impact on the S11 and S21 parameters of the aged LNA circuit. (e) Impact on the loop gain of the new LDO circuit. (f) Impact on the loop gain of the aged LDO circuit. (g) Impact on the PSR of the new LDO circuit. (h) Impact on the PSR of the aged LDO circuit. Correct key (CK), incorrect key (IK), and power supply rejection (PSR).

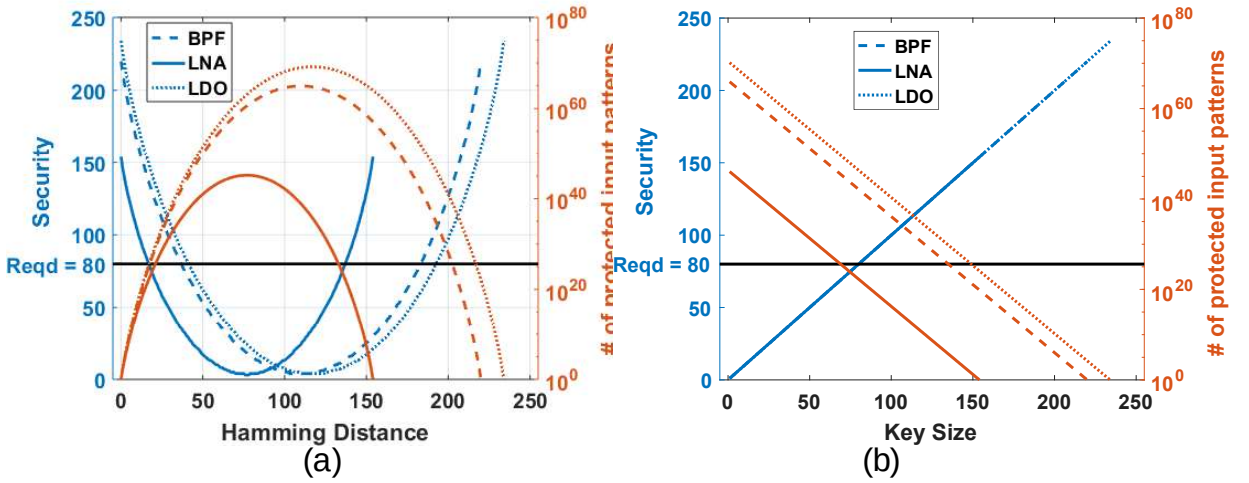

Figure 2.9: (a) The impact of HD on SAT attack resiliency and the number of input patterns protected in SFLL-HD^h. The right-hand side y-axis is in log scale. (b) Key size vs. SAT attack resiliency and the number of input patterns protected for BPF, LNA, and LDO. The right-hand side y-axis is in log scale. The Hamming distance $h = 0$. The security level achieved by BPF, LNA, and LDO are the same and hence, are superimposed.

corresponding number of input patterns which can be protected are $1 <$ # of patterns protected $< 1.37 \times 10^{42}$. Similarly, for LNA, the allowable value of h is $0 \leq h \leq 17$ or $137 \leq h \leq 154$ and the number of input patterns which are protected ranges $(1, 1.73 \times 10^{22})$. For the LDO, $0 \leq h \leq 41$ or $193 \leq h \leq 234$ and the input patterns protected are in the range $(1, 9.89 \times 10^{45})$.

From Fig. 2.9(b), the security increases with the increase in key size, whereas the number of input patterns protected reduces with the increase in key size. A key size, $k \geq 80$ ensures resilience against the SAT attack. Hence, the number of input patterns that can be protected ranges $(1, 1.39 \times 10^{42})$ for BPF. Likewise, the number of input patterns protected for LNA ranges $(1, 1.89 \times 10^{22})$ and that of LDO is $(1, 2.28 \times 10^{46})$. The time required for the attack, as shown in Fig. 2.10, increases exponentially with the input size. For the input size of 14, the attack takes close to 1.5 hours to identify the key. This trend indicates that our technique is secure against the SAT attack. Similarly, it is also secure against AppSAT [65], as we protect only a linear number of input patterns.

Resiliency against removal attack [46]. An attacker cannot remove the locked optimizer circuit and make the analog circuit functional because the tuning knobs are not set to the optimal values

due to process variations, thus preventing removal attacks. If he removes the locked optimizer unit, the circuit parameters will be fixed to the default value. The probability of this value being equal to the desired value to address process variations is negligible. An attacker cannot set the tuning knob value to its optimal value through a focused-ion-beam (FIB) because even identifying the value of one chip cannot be used to set the value for another chip, as these values are different because of process variations. In other words, the amount of compensation varies from one chip to another chip.

Resiliency against bypass attacks [30]. Bypass attack finds the PIPs that give an incorrect output for an incorrect key. The attacker adds a bypass circuitry around the protection block to restore the output for those PIPs. However, the bypass attack cannot compute all the PIPs from the circuit protected using SFLL. This is because, in SFLL, a PIP produces the same incorrect output for most of the incorrect key values. Also, the output corresponding to the PIP may be restored correctly even for an incorrect key. Hence, the bypass attack does not consider the corresponding input pattern as PIP. Therefore, the construction of the bypass circuitry using the incomplete set of PIPs will be erroneous. Thus, it renders the attack unsuccessful.

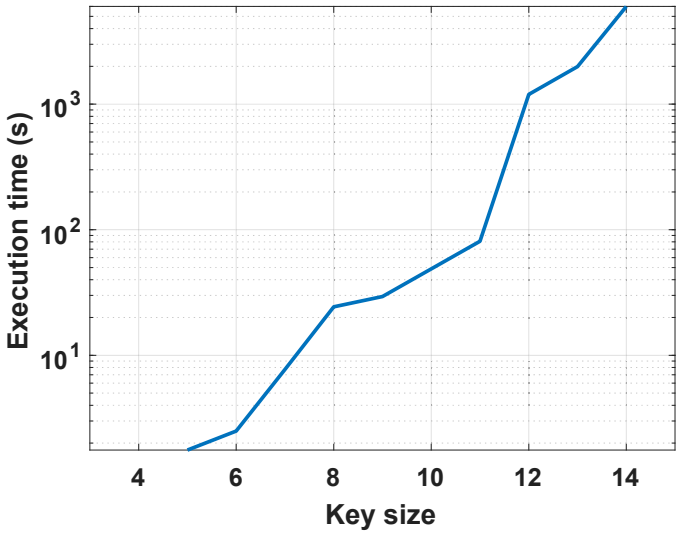

Figure 2.10: Execution time of the SAT attack for BPF. The time required for the attack to find the key increases exponentially with respect to key size. Note that y-axis is in log scale.

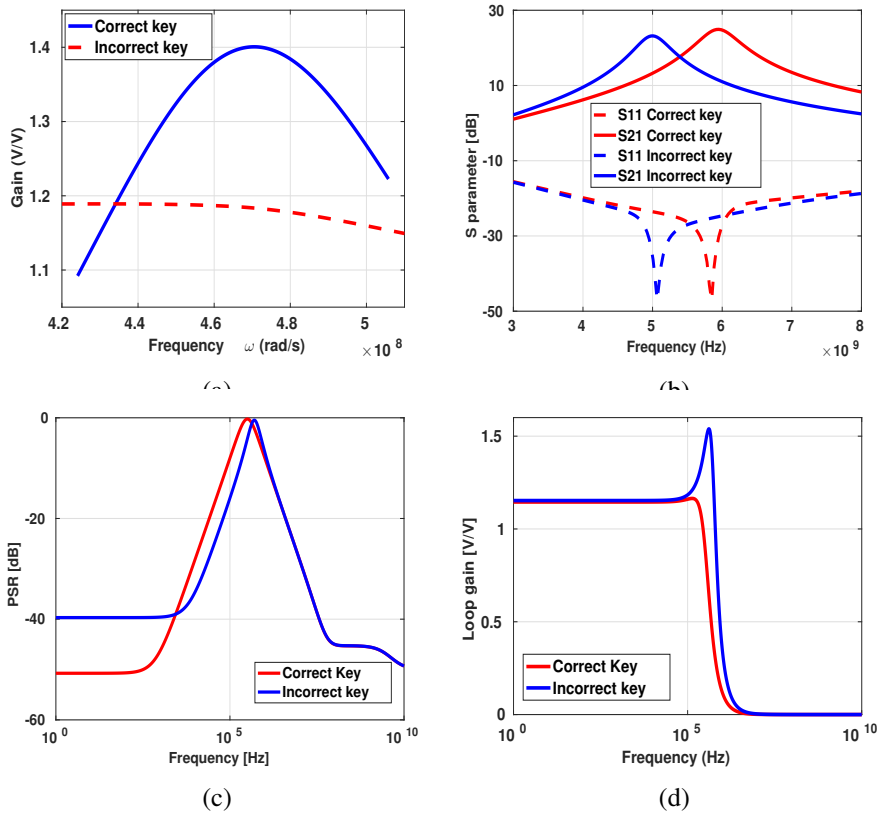

Figure 2.11: Behavior of the analog circuits for correct and incorrect keys on using SFLL-HD^0. The key size used for BPF, LNA, and LDO are 87, 81, and 109, respectively. (a) Frequency response of BPF, (b) S-parameters of LNA, (c) Power supply rejection (PSR) of LDO, and (d) Loop gain of LDO.

2.4.5 Effect of incorrect keys

The response of the circuits for a correct and an incorrect key are compared for the most commonly used metrics for the analog circuits-under-protection [55, 57, 58]. The BIST structure should be enhanced to measure advanced metrics, e.g., LNA noise. This is because the transient analysis and the quadratic sampling are not sufficient to measure this metric. Fig. 2.11(a) shows the difference in the frequency response of the BPF. In this case, the correct key allows the optimizer, tune the circuit to the target ω_o and BW, while an incorrect key forces the optimizer to tune to a lower frequency and also reduces the Q and gain values. Fig. 2.11(b) compares the difference in the S-parameters of the LNA targeting $f_R = 6GHz$. One can observe an error of $1GHz$ in f_R and

an error on S_{11} and S_{21} of at least 15 dB. Finally, the deviation on the LDO performance for the two cases was evaluated. Fig. 2.11(c) shows a degradation close to 10dB in the PSR. Fig. 2.11(d) shows a large peaking on the loop gain for the incorrect key, which indicates a low phase margin and potential instability.

From Fig. 2.11(b) it is evident that the deviation in gain S21 is 10dB. As mentioned in [57], the gain of the LNA must be large enough to minimize the noise contribution, specifically in the downconversion mixers. This 10dB reduction leads to higher input noise corrupting the signal. Also, the LNA is designed such that the gain has its peak value at the frequency band of interest. If the amplifier resonates at another frequency, it does not only mean that the signal of interest is not amplified enough, but also that the receiver is acquiring the signal from a different frequency band. In communications, this causes interference between different channels and the channel of interest.

2.4.6 Analog circuit's performance for a random tuning knob setting

The minimum percentage normalized mean-squared-error (NMSE) due to arbitrary tuning knob settings can be less than the NMSE due to process variation, thereby giving a performance close to the desired one. However, as the attacker does not have the resources to modify the layout, it is not possible to choose an arbitrary tuning knob setting. The following are the reasons why choosing the tuning knob at random does not always work:

1. Please note that the NMSE in the output response when the correct tuning knob is chosen is 0 (as the actual response is equal to the expected response). However, the probability of randomly choosing this optimal tuning knob setting is 1/1024, as there are a total of 1024 tuning knob settings.

2. As this technique is resilient only against the overproduction attack, the attacker does not have the resources for layout-level modifications such as removing the locked optimizer or changing the circuit parameters to control the gain of LNA.

3. Even if the attacker removes the locked optimizer, the optimal tuning knob setting will

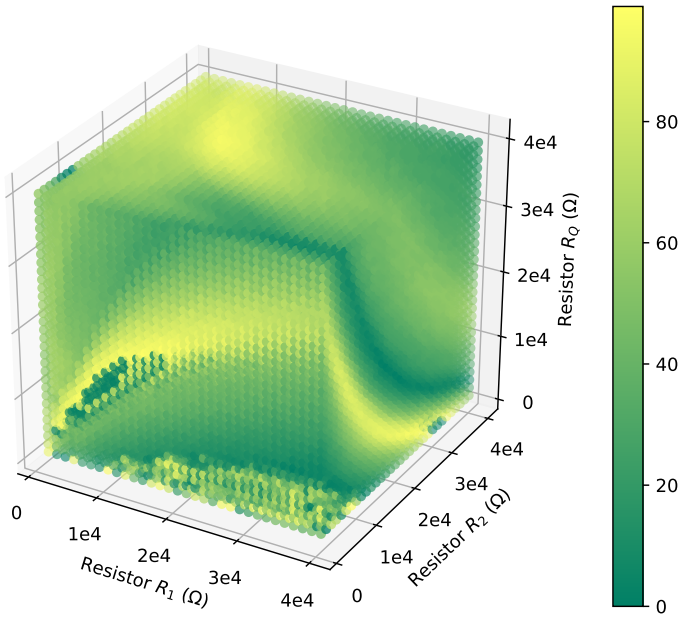

Figure 2.12: Cost function (error) for all possible settings of three tuning knobs ($\mathbf{R_1}$, $\mathbf{R_2}$, and $\mathbf{R_Q}$).

change chip to chip due to process variations. Hence, choosing the same setting on all chips will not compensate for process variation.

A statistical representation of the NMSE for more number of arbitrary tuning knob settings may not help infer the impact of random tuning knob selections due to the above reasons.

2.4.7 Impact of increasing the number of tuning knobs

In this section, we explain the effect of increasing the number of tuning knobs, i.e., the tunable parameters on the security of the locked AMS circuit. Along with the resistors R_1 and R_2, R_Q is also made tunable. Hence, there are three 5-bit tuning knobs. Fig. 2.12 shows the cost function value for all possible tuning knob settings. As illustrated in the figure, very few tuning knob settings give the desired circuit performance, i.e., minimal cost function value. The optimizer controlling the three tuning knobs is locked using SFLL-flex. Here, all possible input patterns (2^{15}) are protected. Fig. 2.13 shows the output response of the BPF circuit for the (in)correct keys supplied to the locked optimizers controlling two and three tuning knobs.

As shown in the figure, the deviation of the output response from the desired response, for an incorrect key, increases as the number of tuning knob increases. As shown in Fig. 2.13, for the

correct key, both the optimizers tune the resistors for providing the required $f_c = 18.84MHz$ and $Gain = 3.1dB$. For an incorrect key, the optimizer controlling two tuning knobs sets the BPF to function with $f_c = 34.47MHz$ and $Gain = 0.362dB$. However, the effect of an incorrect key on the optimizer controlling three tuning knobs has greater impact on the BPF's performance. It sets the tuning knobs such that $f_c = 23.71MHz$ and $Gain = 24.34dB$. Unlike two tuning knobs, where only f_c varies considerably and $Gain$ varies marginally, in three tuning knob settings, both the metrics vary considerably.

2.4.8 Effect of aging on the locked-optimizer

Analog circuits are subjected to aging effects such as negative bias temperature instability (NBTI) [66] and hot carrier injection (HCI) [67]. PMOS transistors are affected by NBTI. An increase in threshold voltage can model this effect in the PMOS transistors. The change in V_{th} due

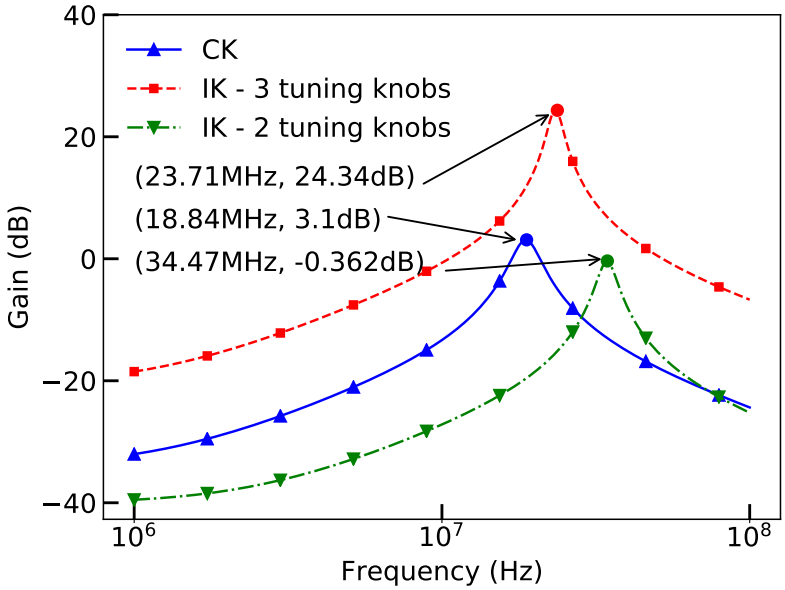

Figure 2.13: The optimizer of the BPFs with two ($\mathbf{R_1}$, $\mathbf{R_2}$) and three tuning knobs ($\mathbf{R_1}$, $\mathbf{R_2}$, and $\mathbf{R_Q}$) are locked using SFLL-flex. The output response for the correct key (CK) is same in both the cases as indicated in the figure. However, the output response for an incorrect key is deviated from the original response much more for the BPF with three tuning knobs rather than two.

to NBTI is modeled by [66] as,

$$\Delta V_{th} \approx exp(\alpha_1 V_{GS}) t^{n_p} + V_{GS}^{\alpha_2}(C_R + n_R log_{10}(t)) \tag{2.5}$$

Here, α_1 and α_2 are voltage scaling factors, which are 0.26 and 2.4, respectively. n_p and n_R are time exponents. They are process-dependent parameters similar to C_R. V_{GS} is the gate to source voltage. The impact of HCI on the NMOS transistors degrades the drain current, which is depicted as an increase in the threshold voltage. The change in V_{th} due to HCI is modeled by [66] as,

$$\Delta V_{th} \approx \frac{1}{\sqrt{L}} exp(\alpha_3 V_{GS}) exp(\alpha_4 V_{DS}) t^{n_{HC}} \tag{2.6}$$

Here, α_3 and α_4 are process-dependent voltage scaling factors, $n_{HC} \approx 0.5$ is a time exponent, L is the transistor length, and V_{DS} is the drain to source voltage.

Consider the locked analog circuit used in applications such as transceivers and communication protocols. As only the analog circuit is powered-on for a longer duration, the impact of aging is comparatively higher on this circuit than the other circuits which are powered-off after the tuning knobs are selected. Hence, the analog circuit is simulated with an increased V_{th} in PMOS and NMOS transistors to model the aging process. This increase in V_{th} causes a marginal difference in the analog circuit response, which in turn varies the input voltage to the locked optimizer marginally. As the increase in V_{th} in PMOS and NMOS increases with the age of the circuit, the output response of the analog circuits changes with age. This trend is illustrated in Fig. 2.7, where the x-axis denotes the frequency of operation, and the y-axis is the frequency response of the second-order BPF. It is evident from the figure that as the analog circuit ages, its output response deviates more from the desired response. A change in the output response results in the change in the input patterns to the locked-optimizer. As the defender has protected only a handful of input patterns, the new input patterns from the aged analog circuit may not be protected. Hence, it is imperative to study the effects of aging on the locked-optimizer.

In the following section, the impact of aging is studied on the logic-locked AMS circuits. The

defender chooses the PIPs that result in a minimum error, i.e., the deviation of the output response from the desired response the attacker has to experience for an incorrect key. The optimizer is locked using SFLL-flex to protect (i) the input patterns corresponding to all possible tuning knob settings which give an error of 50% (ii) the input patterns which give an error of less than 27%, and (iii) the input patterns which give an error of less than 39%. The PIPs corresponds to the analog circuit before aging. Figures 2.8 (a), (c), (e), and (g) show the output responses for the analog circuits when the locked-optimizer is supplied with correct and incorrect keys for the above three cases. As the number of PIPs increases, the error in output response for an incorrect key increases. The impact of aging on the frequency response, S11 and S21 parameters, and PSR and loop gain, are plotted for BPF, LNA, and LDO, respectively. For the analog circuits considered, the output response corresponding to the incorrect keys, IK 1a and IK 1b output responses are for the optimizer where the input patterns corresponding to all the are tuning knobs are protected. For the incorrect key IK 1a, all the bits of the input pattern are considered by the locking mechanism, thereby achieving a key size of 220, 154, and 240 bits for BPF, LNA, and LDO, respectively. Whereas for the response due to incorrect key IK 1b, only the MSBs and the constants bits of the voltage inputs are considered by the locking mechanisms, and the rest of the bit positions in the input patterns are considered as don't cares. This reduces the key size to 80, 84, and 122 for BPF, LNA, and LDO, respectively. Similarly, the output responses corresponding to incorrect keys 2 and 3 belong to the optimizer, which protects the input patterns whose error in output response is less than 39% and less than 27%, respectively. The optimizer considers all the bits similar to the response for incorrect key IK 1a case, thereby achieving a larger key size equaling the input size.

The analog circuit is now replaced with the aged circuit to study the effect of aging on the SFLL locking. Figures 2.8 (b), (d), (f), and (h) shows the output responses for the analog circuits, when the optimizer locked using SFLL-flex, is supplied with correct and incorrect key IK 1b based on the input patterns they are protecting. As few of the bit positions in the input patterns to the optimizer (from the analog circuit) have changed due to aging (especially LSBs), the locked optimizer considering all the bit positions in the input patterns fails to secure the circuit. This trend

is illustrated in the figures 2.8 (b), (d), (f), and (h). However, the optimizer which considers only the MSBs (and/or the bit positions whose values are constant) with lower security level (but more than 80) protects the analog circuits as shown by the output response for incorrect key IK 1b.

2.4.9 Discussion

Why can we not protect all the digital components in the AMS circuit? A simple solution is to protect all the digital components of the AMS circuit. However, this seemingly straightforward approach is not simple and may not meet the desiderata for analog circuits, for reasons below. The desiderata for protecting AMS circuit via logic locking of digital components:

- An attacker should not be able to identify the locked digital part, remove it, and make the resultant analog circuit functional[†].
- Logic locking the entire digital circuit may not necessarily yield incorrect responses from the analog part. Hence, the digital circuit needs to be locked such that the analog component becomes non-functional when an incorrect key is applied.
- State-of-the-art logic locking techniques can protect only a linear number of input patterns in key size [9]. Hence, one needs to select which input patterns to protect, such that incorrect keys will have the highest impact on the functionality of the analog circuit.
- Locking the entire circuit incurs high area, power, and delay overhead. Hence, one has to be judicious in selecting which components to protect.

Power, delay, and area overheads. In this implementation, the power overhead is not a concern since the optimization and security platform is consuming power only at the start time for a short period. Once the optimization finds a solution and sets the tuning knobs, the digital core is turned off. Concerning the delay overhead, there is a delay between the circuit turn-on time and the time at which the regular operation starts. This delay is the time taken by the optimizer to choose the optimal tuning knob settings. Area overhead of SFLL-HD^0 for BPF, LNA, and

[†]Here, we consider an analog design as functional when it produces the expected response.

LDO is 8.79%, 2.61%, and 3.08%, respectively. Similarly, for SFLL-HD^h, the overhead is 8.78%, 5.84%, and 4.91%, respectively, for the h values listed in Table 2.1. In case of SFLL-flex, the overhead is 0.14%, 0.13%, and 0.14%, for BPF, LNA, and LDO, respectively, proving to be the most area-efficient variant of SFLL. **Effect of temperature and environmental noise.** The on-chip optimizer can measure the performance of the AMS circuit with respect to the circuit components along with operating conditions, such as temperature and noise on the power supply. Thus, the optimizer can recalibrate and set the tuning knobs to obtain the desired response—but only when the correct key is in place. Thus, our technique can ensure the effect of locking, even in the presence of temperature variation and environmental noise.

Can we individually attack the analog and digital sections of the AMS circuit? As we are targeting resilience only against overproduction attacks, the threat model assumes that the attacker can only overproduce the chip but cannot physically modify the existing layout. Hence, he/she cannot remove the optimizer circuit and independently target the analog circuit. Even if the attacker simulates the analog circuit for the different tuning knob settings and determines the correct settings, he/she cannot change the tuning knob as they are not controllable by the attacker.

What guarantees that an incorrect key does not produce the correct circuit performances? As the optimizer is locked using SFLL-flex, the guarantees that an incorrect key does not produce the correct circuit response is given by the security metrics of SFLL-flex. This is because the circuit response depends on the unique settings of one of the 1024 tuning knob settings, which in turn is the output of the locked optimizer. As the input patterns which have the highest impact on the correct tuning knob settings are protected, unless the optimizer is provided with the correct key, the analog circuit does not produce the desired output response. Only the correct key selects the optimal tuning knobs, whereas all the incorrect key selects sub-optimal tuning knobs that produce sub-optimal performance. Hence, as given in [9], the output is corrupted for those input patterns which are protected. This is given by the SAT and removal attack resiliencies, which are $k - \lceil \log_2 c \rceil$ and $c \times 2^{n-k}$, respectively. Here, n, k, and c are the input size, key size, and the number of protected input patterns, respectively.

Is it possible to determine the correct key to unlock the optimizer using the quantified Boolean formulation? An attacker can try finding the key using the input and output relationship of the analog circuit. In the oracle, an attacker can control and observe the input and output ports, respectively, of the analog circuit. He/she can send in the desired input and observe the corresponding output as the key input is loaded with the correct key. He/she can formulate the operation of the analog circuit as QBF equations and try to find the optimizer key that satisfies this relationship. However, an attacker cannot find this key to unlock the optimizer because of the nature of digital locking techniques. Furthermore, an attacker does not know the complete input and output relationship of the optimizer to build his/her own optimizer. This is due to the following reasons: (i) the underlying digital locking techniques are secure in revealing the complete functionality of the optimizer and (ii) every locked chip only reveals one or few input-output relationships of the optimizer, and this is not enough to build the complete optimizer. This happens because the effects of process variations are different for different chip and thus applying different inputs to the optimizer.

Due to the large size of the passive components, the number of components in an array is limited. Does this limit the key search space? As this technique is resilient only against overproduction, the key size is not dependent on the number of passive devices in the array. Instead, it is dependent on the size of the input to the locked-optimizer. Depending on the process variation impact, the correct value of the passive device is chosen by the optimizer via the tuning knobs. Hence, it is necessary to unlock the optimizer, which is protected by SFLL. In this locking technique, the key size should be less than or equal to the input size. For example, in BPF, as the input size to the optimizer is 220 bits, the key size can be a maximum of 220 bits. Hence, this key size and hence, the key search space is not limited by the size or number of passive devices in the array.

2.5 Conclusion

In this chapter, we propose the first technique to thwart the overproduction of AMS circuits, by securely locking the digital part, which is controlling the tuning knobs judiciously. Our analysis indicates that by properly selecting two tuning knobs, we can secure several performance metrics

of different analog circuits—a BPF, an LNA, and an LDO. On applying an incorrect key, our approach achieves at least 27.45% error and at most 50% error in the circuit's response when the optimizer is locked using SFLL-flex. Our technique is agnostic to logic locking techniques: we have used SFLL-flex [9], as it can prevent SAT [51], AppSAT [65], removal [46], sensitization [28], and bypass attacks [30]. Our approach is provably-secure, as it leverages the properties of SFLL. More importantly, it is well integrated with the analog component, without sacrificing the security properties of SFLL.

From the simulation results, one can conclude that the SFLL-flex technique secures the analog circuit irrespective of aging. We can also increase the error in the output response of the analog circuit experienced by the attacker for an incorrect key by increasing the number of tunable parameters in the circuit. Our future work entails: (i) Exploring the effect of other logic locking techniques; (ii) Embedding secret keys as part of analog designs, not just digital; and (iii) Exploring techniques to prevent piracy and not just overproduction.

3. BREAKING ANALOG LOCKING TECHNIQUES*

3.1 Introduction

As building an integrated circuit (IC) fabrication unit costs billions of dollars, most companies have gone fabless [68]. Fabrication outsourcing has led to the vulnerability of supply-chain attacks, such as intellectual property (IP) piracy, counterfeiting, overproduction, reverse engineering (RE), and hardware Trojan insertions [68]. The works [27] and [9] propose several design-for-trust (DfTr) techniques, such as IC metering, watermarking, logic locking, split manufacturing, and camouflaging. These techniques help to thwart the supply-chain attacks. Logic locking protects the circuit against an attacker in an untrusted foundry, an end-user, or both. It is the preferred DfTr technique, as it defends against the attackers across the supply-chain. Logic locking modifies the design such that on applying the (in)correct key, the circuit gives the (in)correct output. Existing DfTr techniques mostly target digital ICs. However, analog ICs are more prone to supply-chain attacks than digital ICs as they are easier to reverse engineer [69]. This high vulnerability is due to their low transistor count compared to their digital counterparts. They also have predefined layout patterns e.g., common-centroid, to tolerate process variations [47].

3.1.1 Related works on analog locking

Different DfTr techniques based on logic locking and camouflaging have been developed for analog ICs [1–5]. They are designed to combat one or more of the following supply-chain attacks, namely, IP piracy, overproduction, illegitimate access of the chip, and RE. We shall now discuss the existing analog-only locking schemes. Researchers use a memristor-based voltage divider, which tunes the body-bias voltage required for offset voltage cancellation in sense amplifiers [3]. The configurations of the memristors are given only to the authorized user. Another work proposes an SMT-based combinational lock [1]. A configurable current mirror (CCM), whose output current

is controlled by the key inputs, produces the bias current (I_B). Circuit specifications, such as center frequency (f_c), bandwidth (BW), and oscillation frequency (ω_{osc}), depend on the precise value of this I_B. Similar to [1], the effective width of the transistor depends on the key inputs in parameter-biasing obfuscation [2], which determines the bias.

A trained analog neural network (ANN) provides the necessary bias voltages to the LNA to work as per the circuit specifications [4]. Only a unique set of input voltages can give the desired bias value, as the ANN is preprogrammed with fixed weights. In [5], analog circuits are camouflaged by replacing the nominal threshold voltage (V_{th}) transistors (NVT) with resized high-V_{th} (HVT) and low-V_{th} (LVT) transistors. The NVT transistors control the circuit specifications, such as f_c, BW, and ω_{osc}. The high-resolution pictures of the depackaged chip cannot reveal the V_{th} type of transistors; thus, the original design cannot be recovered.

3.1.2 Related works on analog and mixed-signal (AMS) locking

Techniques for protecting AMS circuits have been proposed in [6, 7, 37, 42]. The work in [6] consists of an analog circuit and a logic-locked optimizer. The key input controls the working of the optimizer. This optimizer sets the correct value of the passive components in the analog circuit, which enables the desired circuit performance. Similarly, in Mixlock [37], unless a correct key is given, the locked digital circuit sets one or more circuit specifications of the analog circuit outside the acceptable range [37]. This technique is demonstrated on a $\Sigma\Delta$ analog to digital converter (ADC) [42]. In shared dependencies, the analog and the digital parts of the AMS circuits are locked using parameter-biasing obfuscation [2] and stripped functionality logic locking (SFLL)-HD^0 [9], respectively [7].

3.1.3 Attacks against digital logic locking

SAT attack is a Boolean satisfiability-based attack on combinational logic-locked circuits [51]. It uses a SAT solver to weed out incorrect keys. SMT attack [70] that is a superset of SAT attack, can handle non-Boolean variables, eg., logic delay. This attack can break the delay logic locking [71]. Removal attack identifies the protection logic and removes it to recover the original

circuit [45]. Bypass attack finds the distinguishing input patterns that give an incorrect output for an incorrect key. The attacker adds a bypass circuitry around the protection block to restore the output for those distinguishing input patterns [30]. The attacks such as SFLL-hd – Unlocked [72] and FALL attack [73] break SFLL-HD^h [9]. These attacks determine the protected input patterns (PIPs) by structurally analyzing the locked netlist. It then extracts the key using the determined PIPs.

3.1.4 Limitations of existing attacks

The attacks on digital locks cannot be applied on analog-only locks because,

1. The output of the analog-only locks is a non-Boolean variable, such as bias current, bias voltage, and transconductance. As the above attacks can handle only Boolean variables, they cannot break analog-only locks.

2. The bias circuit is locked using the techniques proposed in [1–3]. Launching removal attack on them removes the bias, which is required for the circuit to be functional.

3. The bypass attack replaces the locked bias circuit with the precise current/voltage source. This attack is feasible only if the attacker knows the precise value of the bias inputs and can pirate the design. Otherwise, it is infeasible.

4. The SMT formulation in [70] targets the delay logic locking [71] and speeds up the SAT attack. The constraints to model the analog locks are different from delay logic locking. Hence, new SMT formulations are required to break the analog-only locks.

Also, the SFLL-HD^h used in [6, 37] is resilient against all the attacks mentioned above, except [72] and [73]. In [72] and [73], for a given input size, key size, and Hamming distance (HD), all possible PIPs are considered to determine the key. However, as detailed in the attack section, the attacker does not have access to all possible PIPs. Hence, we need updated SAT formulations that can return the correct key. Therefore, there arises a need for developing an evaluation technique for

existing analog-only and AMS locks. In this chapter, we focus on building such evaluation techniques. New SMT (SAT) formulations are developed to break the analog [1–5] (AMS [6,7,37,42]) locks.

3.1.5 Naïve manual attack against analog locking

Attacking a lock of analog IC like [1] means to figure out the correct key or configuration of the locking circuit. Conceivably, there are two naïve approaches: (i) brute-forcing all key combinations; (ii) redesigning the circuit without the lock. The former approach takes exponential time with respect to key size, which is prohibitively expensive. The latter entails high design expertise and design effort. As such, none of them is appealing to attackers pursuing fast and cheap solutions.

3.1.6 Our approach and contributions

In this work, we propose the SMT- and the SAT-based attacks to break the analog locks and digital locks in AMS circuits, respectively. Based on the information collected from various sources tabulated in Table 3.1, an attacker can find the correct key for proper circuit operation using SMT and SAT formulations. We have developed attacks to break (i) analog-only locks [1–5], (ii) digital locks in AMS circuits [6, 7, 42], and (iii) analog locks in AMS circuits [7]. The contributions of this chapter are:

- We propose new SMT formulations to break analog locks [1–5].
- We demonstrate our attack on the combinational locks [1] and the parameter-biasing obfuscation [2]. We demonstrate it on the following analog circuits due to their ubiquitous presence in wireless communication networks: Gm-C BPF, LC oscillator, quadrature oscillator, and class-D amplifier.
- We validate our attack on memristor-based protection [3].
- We propose SAT formulations to break digital locks in AMS circuits and demonstrate this attack on [6].

Table 3.1: Sources of information available to the attacker. Resistor (R), capacitor (C), width (W) and length (L) of the transistor, mobility (μ), oxide capacitance ($\mathbf{C_{ox}}$), oxide thickness ($\mathbf{t_{ox}}$), threshold voltage ($\mathbf{V_{th}}$), bias current ($\mathbf{I_B}$), bias voltage ($\mathbf{V_B}$), input reference current ($\mathbf{I_{ref}}$) and voltage ($\mathbf{V_{ref}}$), transconductance ($\mathbf{g_m}$), bandwidth (BW), and oscillation frequency ($\omega_{\mathbf{osc}}$).

Source	**Information acquired**
Layout file from the foundry or the reverse engineered netlist using the oracle [74]	Sizes of passive components (R, C)
	Key size and transistor count
	W and L of the transistors
	Key connectivity to transistor switches or memristors
Technology library [75] (PDK documentation)	Values of passive components (R, C) and transistor details (μ, C_{ox}, t_{ox}, V_{th})
	Availability of different V_{th} transistors
Circuit specification [76]	Minimum and maximum values of I_B and V_B, which are output of the bias circuit
	Values of I_{ref} and V_{ref}, which are the input to the bias circuit
	Minimum and maximum values of the resistance that can be programmed into the memristors
	Values of circuit parameters (BW, ω_{osc})

- We demonstrate the attack on the defense in [7], which requires both SMT and SAT formulations.
- We extend our attack to evaluate analog camouflaging [5].

3.2 Attack approach

3.2.1 Threat model

As stated in Table 3.1, we consider the following threat model, where both the foundry and the end-user are untrusted entities [6,9,27,37,45,51]. The attacker in the untrusted foundry has access to the layout of the design provided by the designer, the process design kit (PDK) documentation,

and the locking algorithm used as this information is public. He/She can overproduce the chip and sell the excess chips in the black market. Likewise, an end-user as an attacker has access to the RE tools to obtain the netlist of a locked chip [77]. It is relatively easy to reverse engineer analog circuits with several hundreds of transistors compared to SoCs with multi-million transistors. Also, compared to digital circuits, the analog circuits have a bigger transistor size [78] and predefined layout patterns, rendering them easier to reverse engineer. Similar to the attacker in the foundry, the untrusted end-user has access to the locking algorithm used. He/She also purchases a chip that has the correct key loaded. This chip serves as an oracle, where the attacker can observe the output for a given input. The manufacturer provides the specification along with the purchased chip; thus, an untrusted end-user can have access to it.

3.2.2 Attack methodology on analog locks.

The analog locks obfuscate the effective value of the circuit components, such as the width of the transistor, resistance, and capacitance. These components are used in the bias circuit, as the precise bias current (I_B) or voltage (V_B) is required for the proper operation of the analog circuits. These components are called **obfuscated components** as they are made configurable, and their effective value is hidden from the attacker. The key input determines the effective values of these components. Only the correct key can set the effective values of the obfuscated components correctly; this is essential for precise biasing conditions and hence, the proper operation of the analog circuits. Our attack aims to find the key that gives the required bias to make the circuit functional. The attack methodology is described below.

1. **Identifying the obfuscated circuit components and their dependency on the key inputs.** From the locked netlist, the attacker can determine the obfuscated components, such as transistors, resistors (Rs), and capacitors (Cs) [74], by tracing the wire connections from the key input. A component y in the original design is replaced by a set of n obfuscation components $\mathbf{x} = \{x_1, x_2, ..., x_n\}$, which are controlled by an m-bit key vector $\mathbf{q} = (q_1, q_2, ..., q_m)$. We denote the values of y and $x_i, i \in \{1, 2, ..., n\}$, by y_v and $x_{i_v}, i \in \{1, 2, ..., n\}$, respectively. Then,

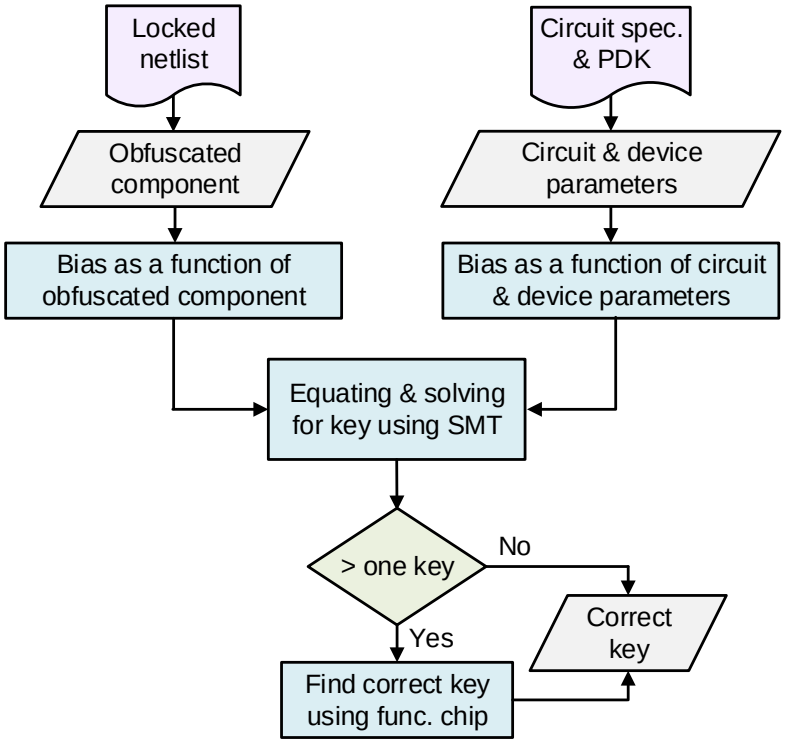

Figure 3.1: Proposed SMT-based attack methodology. Circuit specification (Circuit spec.), process development kit (PDK), and functional chip (func. chip).

the effective value of the obfuscated components $\mathbf{x}$ is,

$$\tilde{y_v} = \phi(\mathbf{x_v}, \mathbf{q}) \tag{3.1}$$

Here, $\mathbf{x_v} = (x_{1_v}, x_{2_v}, ..., x_{n_v})$, and the function ϕ depends on how the obfuscation circuit is constructed. For the correct key $\mathbf{q}^*$, $\tilde{y_v} = y_v$, i.e., the effective value of the obfuscated component is equal to that of the original one.

2. **Finding the equation linking the value of the obfuscated component $\tilde{\mathbf{y_v}}$ to the bias $\mathbf{z_{ob\_comp}}$.** The bias z_{ob_comp} (e.g., I_B or V_B), is a function ψ of the value of the obfuscated component $\tilde{y_v}$.

$$z_{ob\_comp} = \psi(\tilde{y_v}) \tag{3.2}$$

Substituting Equation (3.1) in Equation (3.2) gives the dependency of z_{ob_comp} on the key $\mathbf{q}$, $z_{ob\_comp} = \psi(\phi(\mathbf{x_v}, \mathbf{q}))$.

3. **Derive bias $\mathbf{z_{spec}}$ from the circuit parameter $\mathbf{p}$ of the protected analog IC.** The attacker can obtain the circuit parameters, such as g_m, BW, and ω_{osc}, from the circuit specification. He/she then analyzes the target circuit and extracts its characteristic equations. Solving these equations yields the bias point I_B or V_B,

$$z_{spec} = \theta(p) \tag{3.3}$$

Here, θ is the function to compute the circuit parameter p [79–81]. If the attacker knows the bias point, he/she can redesign the entire analog circuit. However, it is sometimes possible to determine only the bias range and not the precise z_{spec}. This is because there may not be a direct equation linking z_{spec} and p. Instead, equations linking the minimum and maximum values of the bias with different circuit parameters are available in the specification [82]. Here, we calculate a range for bias using the equations $z_{spec_{min}} = \theta_1(p_1)$ and $z_{spec_{max}} = \theta_2(p_2)$.

$$z_{spec_{min}} \leq z_{spec} \leq z_{spec_{max}} \tag{3.4}$$

where θ_1 and θ_2 are the functions to compute p_1 and p_2, respectively. $z_{spec_{min}}$ and $z_{spec_{max}}$ are the minimum and maximum values of the bias, respectively.

4. **Putting it all together.** The I_B or V_B obtained from circuit specification and the I_B or V_B estimated from the obfuscated components should be equal or approximately equal for the analog circuit to be functional. Hence, the bias z_{ob_comp} equals z_{spec}, or alternatively, z_{ob_comp} satisfies the inequality specified by z_{spec}. Solving the equations (3.1), (3.2), and (3.4) using the SMT-solver [83] computes the correct key $\mathbf{q}^*$. If the solver returns more than one key, the attacker compares the output response of an unlocked netlist for each of these keys with the oracle's response. He/She can choose the key that gives the desired or close to the desired

output response. The correct key sets the effective value of the obfuscated component equal to the value of the original component, i.e., $\tilde{y_v} = y_v$.

The overall attack methodology is shown in Fig. 3.1.

3.2.3 Attack methodology on digital logic locking

SFLL [9] has been extensively used in locking the digital section of the AMS circuits in [6, 7, 37, 42]. We shall explain the working followed by the attack methodology on this technique.

There are different variants in SFLL [9], such as SFLL-HD^0, SFLL-HD^h, SFLL-flex, and SFLL-fault. It protects only a certain number of input patterns called the PIPs. The output of the original circuit, O1, is inverted only when the input pattern (IN) is a PIP. The functionality-stripped circuit comprises of the original circuit, the inversion logic, and the logic which checks if the IN is a PIP. Depending on the variant of SFLL, the corruption injected by the inversion logic is restored, when

1. the HD between the external key (k) and the IN equals 0 in SFLL-HD^0.

2. the HD between k and the IN equals h in SFLL-HD^h, as indicated in Fig. 3.2.

3. the input equals one of the PIPs that are stored in a content addressable memory in SFLL-flex.

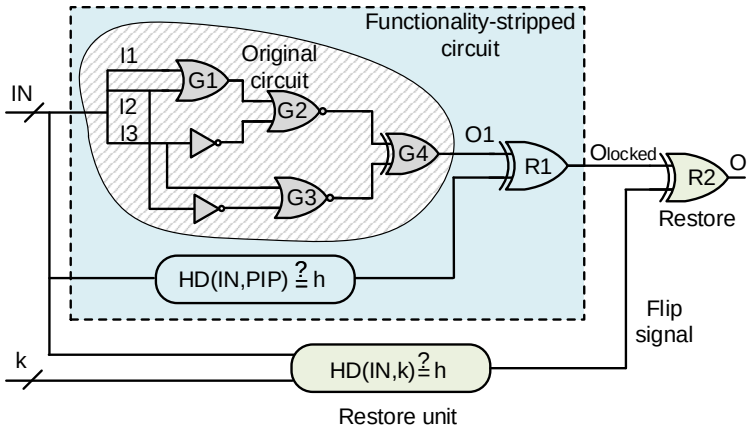

Figure 3.2: Stripped-functionality logic locking [9]. Hamming distance (HD), key (k), input pattern (IN), protected input pattern (PIP).

Attack methodology. As SFLL protects only a handful of input patterns, the probability of finding them is negligible [9]. However, when this technique is used to lock the digital section of the AMS circuits, the PIPs can be found by analyzing the analog-digital interface signals. The key is then determined using SAT formulations, which are discussed next.

1. **Finding the PIPs.** In [6, 7, 37, 42], the analog section of the AMS circuit drives the input to the locked digital section. By simulating the analog section of the netlist, the attacker can determine the INs that drives the digital section. The inputs from the analog circuit are the only INs to the locked digital circuit. Hence, the PIPs should be the subset of these INs.

2. **SAT formulation to determine the correct key.** The attack formulations are unique based on the locking technique used [6,7]. They are explained in the respective attack sections, 3.2.8 and 3.2.9. The correct key is found by solving these formulations.

We first demonstrate our attack on analog locks such as combinational locks [1] and parameter-biasing obfuscation [2]. We then show how this attack methodology breaks other analog IP protection techniques [2–5, 7]. Following this, the attack on digital logic locking techniques protecting the AMS circuits [6, 7, 37, 42] is demonstrated.

3.2.4 Attack on combinational locks [1] and parameter-biasing obfuscation [2]

The defense technique in [1] proposes an SMT-based combinational locking technique. Circuit parameter, such as f_c, BW, and ω_{osc}, depends on the precise value of the bias current (I_B). A CCM generates this I_B. The key input configures the effective width of the mirroring transistor in the CCM. The transistor sizes are modeled using the SMT formulations such that on a correct key, the CCM gives the desired I_B. Otherwise, I_B is outside the range $((1-\Delta)I_B, (1+\theta)I_B)$, where Δ and θ are lower and upper bounds, respectively. The defender sets these bounds. A similar technique [2] chooses the transistor sizes randomly and hence, can have more than one correct key. We consider the operational transconductance amplifier (OTA) shown in Fig. 3.3 (c) as an example to demonstrate our attack. The current mirror supplies the required I_B to the OTA, as illustrated in Fig. 3.3(b). The transistor switch matrix replaces the mirroring transistor y, as

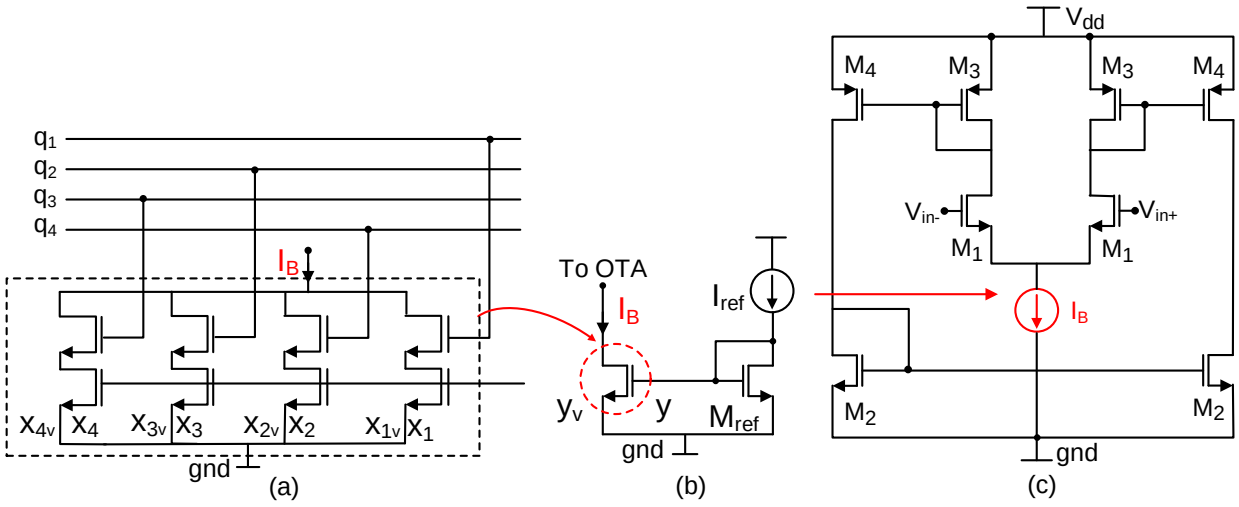

Figure 3.3: Configurable current mirrors (CCM) to thwart IP piracy of analog circuits [1]. (a) Matrix of transistor switches which will replace the transistor $\mathbf{y}$. (b) Original current mirror. (c) Operational transconductance amplifier (OTA) with CCM supplying the bias current.

shown in Fig. 3.3(a) and 3.3(b). The key-bits are connected to the gate terminal of the switches, controlling the magnitude of I_B. The SMT formulations required for this attack are:

1. **Obfuscated component.** y_v is the ratio of $\left(\frac{W}{L}\right)$ of transistor y with respect to $\left(\frac{W}{L}\right)$ of M_{ref} in Fig. 3.3, where $\left(\frac{W}{L}\right)$ is the aspect ratio of the transistor. In the CCM, the key input controls this effective size ratio ($\tilde{y_v}$). y is replaced by n (=4) NMOS transistor switches. The gate terminals of these switches are controlled by the key $\mathbf{q}$. x_{i_v} is the ratio of the aspect ratio of transistor x_i with respect to the aspect ratio of M_{ref}, where $i \in \{1, 2, 3, 4\}$. The attacker can get the values x_{i_v} from the locked netlist, as shown in Table 3.1. He/She can also obtain the details of which key-bit controls which transistor switch from the locked netlist. Hence, the obfuscated size ratio is, $\tilde{y_v} = \sum_{i=1}^{n} x_{i_v} q_k$, where $q_k \in \{0, 1\}$ and $k \in \{1, 2, 3, 4\}$.

2. **Equations linking the bias $\mathbf{z_{ob\_comp}}$ with obfuscated widths $\mathbf{\tilde{y_{1_v}}}$ and $\mathbf{\tilde{y_{2_v}}}$ of the transistors.** In the analog locking techniques [1–3], the bias circuit is designed to be either a current mirror or voltage divider. If the bias circuit is a current mirror as in [1], I_B of CCM is $\psi(\tilde{y_v}) = I_B = \tilde{y_v} \times I_{ref}$, where I_{ref} is the reference current obtained from the circuit specification. For a voltage divider built using resistors y_1 and y_2, the bias voltage is determined by, $\psi(\tilde{y_{1_v}}, \tilde{y_{2_v}}) = V_B = \frac{V_{ref} \times \tilde{y_{2_v}}}{\tilde{y_{1_v}} + \tilde{y_{2_v}}}$. Here, y_1 and y_2 are the original resistors replaced by a set of obfuscated resistors whose

effective values are $\tilde{y_{1_v}}$ and $\tilde{y_{2_v}}$, respectively. [3] realizes y_1 and y_2 as resistors. [2] realizes them as the resistance offered by transistor switches whose resistivity $y_v = \frac{1}{g_m} = \frac{1}{\sqrt{2\mu C_{ox}\left(\frac{W}{L}\right) I_D}}$, where the drain current $I_D = I_B/2$. g_m is the transconductance of the transistor, μ is the mobility of the transistor, and C_{ox} is the oxide capacitance. These values are available in the PDK [75]. V_{ref} is the reference voltage, which is obtained from the circuit specification [82].

3. **Equation linking the bias $\mathbf{z_{spec}}$ with circuit parameter $\mathbf{p}$.** The g_m of the OTA shown in Fig. 3.3(c) is $\frac{g_{m1}g_{m4}}{g_{m3}}$. Here, g_{mi} is the transconductance of transistor Mi. If x_v is the ratio of the aspect ratio of transistor $M4$ to the aspect ratio of $M3$, then $g_m = x_v \times g_{m1}$. The attacker finds I_B using $\theta(g_m) = I_B = \frac{g_m^2}{\mu C_{ox}\frac{W}{L}}$. To calculate this desired I_B, he/she obtains the value of g_m from the specification, device parameters (μ and C_{ox}) from the PDK, and the transistor dimensions (W and L), from the netlist.

4. **Putting it all together.** Solving the equations (A), (B), and (C) in Table 3.2, gives the correct key $\mathbf{q}^*$. This key sets the required g_m in the OTA.

The above attack methodology can break the combinational locking technique [1]. The same methodology can also break the parameter-biasing obfuscation technique [2]. This is because the technique used in parameter-biasing obfuscation [2] is similar to [1]. In [2], the width of the transistor in the bias circuit is obfuscated. This is achieved by replacing the transistor with multiple transistors connected in parallel, as illustrated in Fig. 3.3 (a) and (b). Hence, the attack methodology of [1] can be extended to [2]. The attack results for [1] and [2], are given in Section 3.3.3 and Section 3.3.4, respectively.

Table 3.2: SMT-based attack is effective on the analog locking techniques in [1–3, 5, 7]. The equations shown here are based on the circuit which is protected. The equations may vary based on the circuit topology.

Defense technique	**Bias affected**	**Circuit parameter affected (p)**	**Attack equations**		
			Equation (A): The value of obfuscated component ($\tilde{y_v}$) as a function of key	**Equation (B): Bias input (z_{ob_comp}) as a function of $\tilde{y_v}$**	**Equation (C): Bias input (z_{spec}) as a function of p**
[1,2,7]	I_B, V_B	g_m, BW, f_{osc}, Q, f_c	$\tilde{y_v} = \sum_{i=1}^{n} x_{i_v} q_k$ $\tilde{y_v} = R = \frac{1}{g_m} = \frac{1}{\sqrt{2\mu C_{ox}\left(\frac{W}{L}\right) I_D}}$	$I_B = \tilde{y_v} \times I_{ref}$ $V_B = \frac{\tilde{y_{2_v}} \times V_{ref}}{\tilde{y_{1_v}} + \tilde{y_{2_V}}}$	$I_B = \frac{g_m^2}{\mu C_{ox} \frac{W}{L}}$
[3]	V_{PROG}	V_{BB}	$\tilde{y_{1_v}} = R_{UPPER} = \left(\sum_{i=1}^{U} \frac{q_k}{x_{i_v}} \right)^{-1}$ $\tilde{y_{2_v}} = R_{LOWER} = \left(\sum_{i=1}^{L} \frac{q_k}{x_{i_v}} \right)^{-1}$	$V_{PROG} = \frac{\tilde{y_{2_v}} \times A \times V_{PP}}{\tilde{y_{1_v}} + \tilde{y_{2_v}}}$	$V_{PROG} = \frac{\omega_{PT} \times R_M}{2\pi\rho} \times \left(\sqrt{\frac{\varphi}{\gamma}} \right)$ here, $R_M = \frac{V_{BB} \times R_1}{V_{DD} - V_{BB}}$
[5]	–	g_m, BW, f_{osc}, Q, f_c	$\tilde{y} = \left(\frac{W}{L}\right)_{camouflaged} = \sum_{i=1}^{3} x_i k_i \times \left(\frac{W}{L}\right)_{NVT}$	–	$g_m = \sqrt{2\mu C_{ox} \left(\frac{W}{L}\right)_{camouflaged} I_D}$

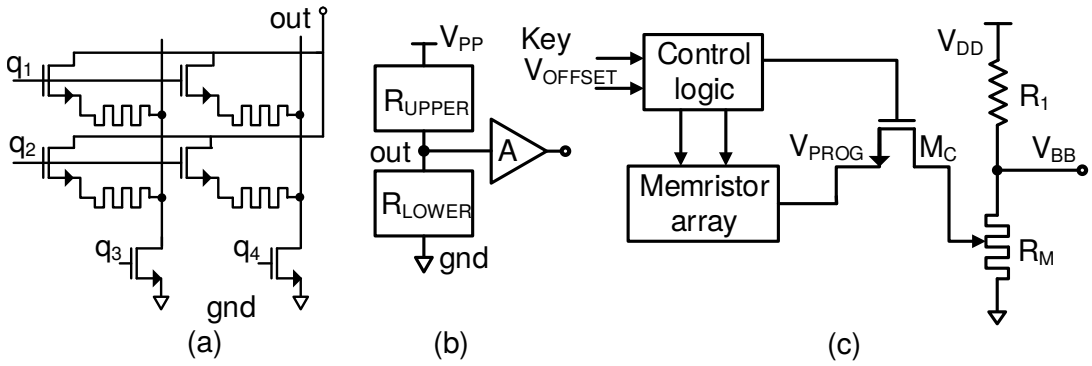

Figure 3.4: Memristor-based obfuscation technique [3]. (a) The memristor crossbar architecture used in the voltage divider circuit. (b) The output of the memristor-based voltage divider is amplified by a factor of $\mathbf{A}$. (c) For a non-zero $\mathbf{V_{OFFSET}}$, the control logic turns on the transistor $\mathbf{M_c}$. This passes the $\mathbf{V_{PROG}}$ generated by the memristor array to program the memristor $\mathbf{R_M}$.

3.2.5 Attack on memristor-based obfuscation [3]

A memristor-based voltage divider in [3] tunes the bulk terminals of the differential pair in the sense amplifier. This tuning is required to cancel the output offset voltage for zero input differential voltage. The voltage divider consists of two memristor crossbars. Each crossbar is constructed using an array of memristors. The key determines the connectivity among these memristors and the effective resistances of the upper (R_{UPPER}) and lower (R_{LOWER}) memristor arrays, illustrated in Fig. 3.4(a) and 3.4(b). Applying the correct key configures the resistivity of the crossbars to provide the required body-bias voltage (V_{BB}). This V_{BB} helps in the offset voltage compensation. An incorrect key provides an undesired V_{BB}, which does not compensate for the offset voltage. This offset voltage affects the sensitivity and reliability of the sense amplifiers.

Attack methodology. The security of this technique lies in the pre-programmed memristor crossbar. It can be compromised if the attacker finds: (1) the value of the pre-programmed resistance in each crossbar, and (2) the connectivity among the memristors, controlled by the key inputs.

1. **Obfuscated component.** The effective resistance of the obfuscated upper and lower memristor crossbars are $\tilde{y_{1_v}}$ and $\tilde{y_{2_v}}$, respectively. Here, $\tilde{y_{1_v}} = R_{UPPER} = \left(\sum_{i=1}^{U} \frac{q_k}{x_{i_v}} \right)^{-1}$, where $k \in$

$\{1, 2, \cdots, m\}, \forall i$ and $\tilde{y_{2_v}} = R_{LOWER} = \left(\sum_{i=1}^{L} \frac{q_k}{x_{i_v}} \right)^{-1}$, where $k \in \{1, 2, \cdots, m\}, \forall i$. Here, $\mathbf{q} = (q_1, q_2, ..., q_m)$ is an m-bit key. This key is shared among both the crossbars. U and L are the number of memristors connected in parallel in the upper and lower crossbars, respectively. x_{i_v} is the pre-programmed memresistance value of memristor i. The attacker can deduce the value of U and L from the layout or reverse-engineered netlist. From the circuit specification, he/she knows the minimum and maximum resistance values of the memristors.

2. **Equations linking the bias $\mathbf{z_{ob\_comp}}$ with obfuscated resistivities $\mathbf{\tilde{y_{1_v}}}$ and $\mathbf{\tilde{y_{2_v}}}$ of the crossbars.** The voltage divider generates the necessary programming voltage, V_{PROG} for the memristor M. The equation linking the bias V_{PROG} with $\tilde{y_{1_v}}$ and $\tilde{y_{2_v}}$ is $z_{ob\_comp} = V_{PROG} = \psi(\tilde{y_{1_v}}, \tilde{y_{2_v}})$. Here, $\psi(\tilde{y_{1_v}}, \tilde{y_{2_v}}) = \frac{\tilde{y_{2_v}}}{\tilde{y_{1_v}} + \tilde{y_{2_v}}} \times AV_{PP} = \frac{\left(\sum_{i=1}^{L} \frac{q_k}{x_{i_v}} \right)^{-1}}{\left(\sum_{i=1}^{U} \frac{q_k}{x_{i_v}} \right)^{-1} + \left(\sum_{i=1}^{L} \frac{q_k}{x_{i_v}} \right)^{-1}} \times AV_{PP}$. Here, A is the amplifier's gain and V_{PP} is the peak-peak voltage. V_{PROG} and V_{PP} can be derived from the circuit specification. The amplifier's gain can be computed from the layout or the reverse-engineered netlist. For example, in a single-stage common gate amplifier, the gain is given as $A = g_m R_D$. Here, g_m is the transconductance of the transistor in the amplifier. This is given by $g_m = \sqrt{2\mu C_{ox} \left(\frac{W}{L} \right) I_D}$, where R_D is the load resistance connected to this transistor. μ and C_{ox} values can be obtained from PDK. The value of I_D is available in the circuit specification. The size of the resistor R_D, W, and L can be extracted from the netlist, and the resistivity R_D is obtained from the PDK.

3. **Equation linking the bias $\mathbf{z_{spec}}$ with circuit parameter $\mathbf{p}$.** The equations connecting V_{PROG} with V_{BB} are $V_{BB} = \frac{R_M}{R_1 + R_M} \times V_{DD}$, where $R_M = \left(\sqrt{\frac{\gamma}{\varphi}} \right) \times \rho \times V_{PROG} \frac{2\pi}{\omega_{PT}}$. Combining the equations gives $z_{spec} = V_{PROG} = \theta(p) = \theta(V_{BB})$. Here, V_{DD} is the supply voltage, R_1 is the fixed resistor in the voltage divider, and R_M is the effective resistance of the memristor M. γ is a constant depending on device parameters such as carrier mobility and device thickness, φ is the flux, ρ is the duty cycle, and ω_{PT} is the frequency of programming pulse. The values of V_B, V_{DD}, φ, γ, ρ, ω_{PT}, and R_1 are available in the circuit specification. The resistivity range with which the memristors can be pre-programmed is (R_{min}, R_{max}), where R_{min} and R_{max} are the

minimum and maximum resistivity of the memristor M. The attacker can obtain the value of R_{min} and R_{max} from the circuit specification of the memristor.

4. **Putting it all together.** Solving these equations gives the correct key and the resistance of each memristor with which it has to be pre-programmed. These equations are consolidated in Table 3.2 as equations (A), (B), and (C). There can be more than one correct key that gives the same V_{BB} due to the memristor array configuration. Hence, the SMT solver is called only once to determine one correct key and one set of memristors' resistance values.

3.2.6 Attack on analog performance locking [4]

In this technique, a trained analog neural network (ANN) provides precise I_B to OTA. This I_B is required for the proper operation of the OTA. The ANN's core shown in Fig. 3.5(a) consists of an $n \times n$ array of synapses (S) and neurons (N). Here, n is the number of rows and columns in the ANN. The first and the last row of synapses are called the input and output layer, respectively. The rows in between them are the hidden layers. Each synapse implements an analog multiplier. Likewise, each of the neurons implements a non-linear activation function, e.g., $tanh$. The network is trained in such a way that for a given set of input voltages, it determines the weights of each synapse to generate the required V_B. Also, it is possible to train the ANN to provide the same V_B for different inputs. Apart from the core, the ANN has:

1. a differential transconductor which converts the differential input voltage to differential currents. These currents are the inputs to the input layer synapses of the ANN.

2. a current-to-voltage converter used for reading the weights of the synapses and the output current from the ANN.

3. a digitally-controlled current source to program the synapses with the weights.

This attack mathematically models the ANN using the SMT formulations. The synapse outputs the product of the inputs along with the weight associated with it. S_{ij} is the synapse output, where i and j are the row and column number of the synapse considered. Sw_{ij} is the weight

associated with the synapse S_{ij}. $IN_{1_{ij}}$ and $IN_{2_{ij}}$ are the two inputs to S_{ij}. Then, S_{ij} is modeled as, $S_{ij} = Sw_{ij} \times IN_{1_{ij}} \times IN_{2_{ij}}, i \neq j$ where, $i, j \in (1, \cdots, n)$. The output of the neurons (N_{ij}) which forms the diagonal elements of the ANN matrix depends on the select signal s_i, given by

$$N_{ij} = \begin{cases} tanh(S_{i(j+1)}) & \text{if } s_i = 1 \wedge i = j = 1 \\ tanh(S_{i(j-1)}) & \text{if } s_i = 1 \wedge i = j = n \\ tanh(S_{i(j-1)} \times S_{i(j+1)}) & \text{if } s_i = 1 \wedge otherwise \\ S_{i(j+1)} & \text{if } s_i = 0 \end{cases}$$

The inputs to the synapses in the input layer ($i = 1$) is given by $IN_{1_{1j}} = IN_{i-1}$ and $IN_{2_{ij}} = S_{1(j+1)}$ and for other layers, the inputs are given by the following equations.

$$IN_{1_{ij}} = \begin{cases} S_{(i-1)j}, \text{if } j \neq i-1 \\ N_{(i-1)j}, \text{if } j = i-1 \end{cases} \quad , IN_{2_{ij}} = \begin{cases} 1, \text{if } j = 1 \vee j = n \\ S_{i(j-1)}, \text{if } j < i \\ S_{i(j+1)}, \text{if } j > i \end{cases}$$

Attack methodology. One method to attack this technique is to remove the ANN and replace it with the bias circuit. However, as stated in [4], this defense technique claims resilience only against illegitimate access to the chips. Hence, the attacker cannot remove the ANN; rather, he/she should program the synapses and neurons with correct weights to produce the desired I_B/V_B. The SMT formulations for the ANN, the differential transconductor, and the current-to-voltage converter are fed to the SMT solver along with the required V_B range. This bias range is essential for the proper operation of the OTA. The solver returns the input voltages to the ANN, weights associated with each synapse, the type of neuron (buffer or $tanh$ activation function), and the value of V_B. The attacker can procure an off-the-shelf digitally-controlled current source to program the new weights into the synapses. The input voltages returned by the attack are fed to the input layer synapses. The ANN thus produces the required I_B or V_B, thereby rendering the attack successful. More than

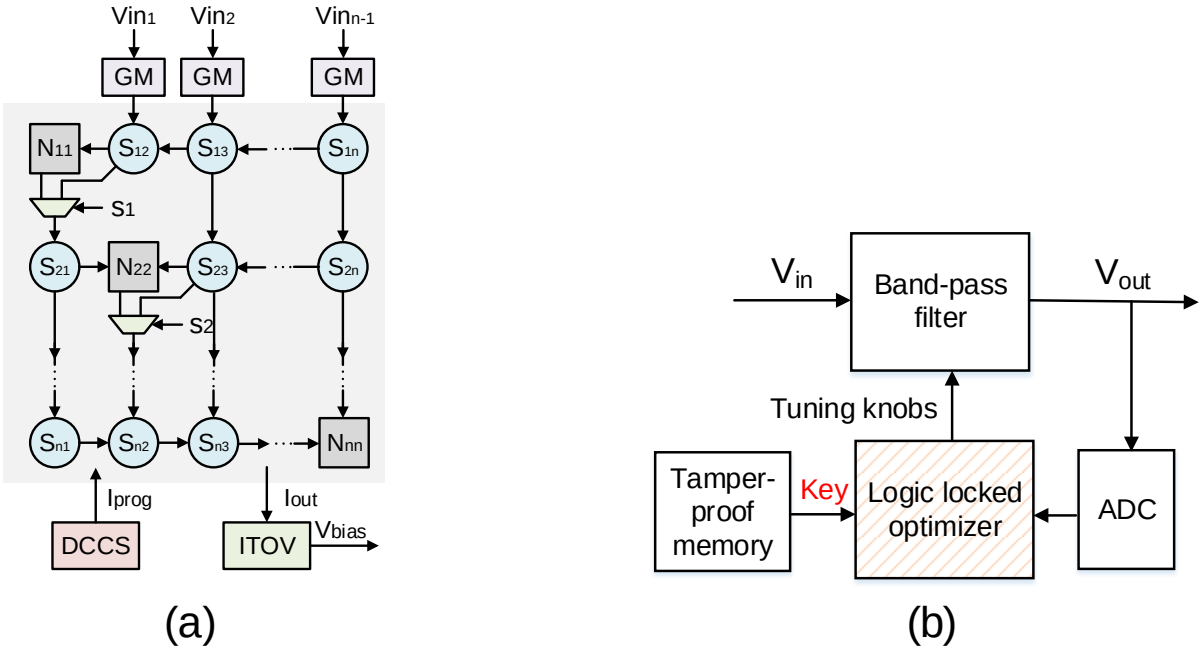

Figure 3.5: (a) The analog neural network along with the input differential transconductance (GM), digitally controlled current source (DCCS), and current to voltage converter (ITOV). The core of the neural network consists of the neurons (N) and synapses (S). (b) Logic locking of the BPF circuit. Only by applying the correct key, the optimizer sets the correct resistor value by considering the effect of process variation [6].

one correct configuration gives the same V_B due to the neural network topology [4]. Hence, the SMT solver is called only once to determine one correct configuration that provides the necessary bias.

3.2.7 Attack on camouflaged analog IPs [5]

The threshold voltage required to switch on a transistor is camouflaged by fabricating the transistor with different dopant concentrations. The threat model considers a trusted foundry and an untrusted end-user. This means the attacker does not have access to the foundry, where he/she can determine the V_{th} type from the layout. TSMC fabricates the transistors in 180nm technology with nominal-V_{th} (NVT), medium-V_{th} (MVT), and native-V_{th} (NaVT). In this technique, few of the NVT transistors are replaced by MVT and NaVT transistors [5]. The functionality is maintained

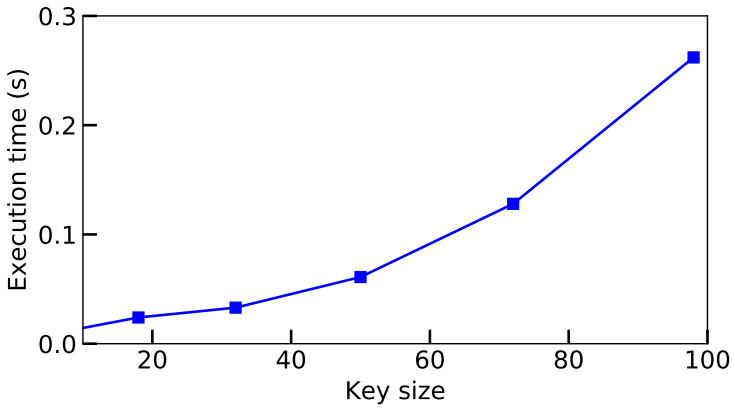

Figure 3.6: The (successful) attack time for increasing key size in memristor-based protection technique [3].

by resizing the replaced transistors. Thus, each camouflaged transistor can be modeled as,

$$\left(\frac{W}{L}\right)_i = x_i \times \left(\frac{W}{L}\right)_{NVT} \tag{3.5}$$

Here, $i \in \{NVT, MVT, NaVT\}$.

Attack methodology. The attacker can transform individual camouflaged transistors to logic locked transistors [84].

$$\left(\frac{W}{L}\right)_{camouflaged} = \begin{cases} \left(\frac{W}{L}\right)_{NVT} & \text{if } q_1 = 1, \\ \left(\frac{W}{L}\right)_{MVT} & \text{if } q_2 = 1, \\ \left(\frac{W}{L}\right)_{NaVT} & \text{if } q_3 = 1. \end{cases} \tag{3.6}$$

Here, q_1, q_2, $q_3 \in \{0, 1\}$ are the key-bits controlling transistors of type NVT, MVT, and NaVT, respectively. Also, $q_1 + q_2 + q_3 = 1$, as each camouflaged transistor can be only one of the three types. The attacker performs the SMT-based attack on the transformed logic-locked circuit as follows:

1. **Obfuscated component.** The transistors in the design are the obfuscated components, as their V_{th} type is unknown to the attacker. From equations (3.5) and (3.6), the aspect ratio of the

camouflaged transistor $\tilde{y} = \phi(x, q)$, is

$$\tilde{y} = \frac{W}{L}_{camouflaged} = \left(\sum_{\forall i} x_i q_i \right) \times \left(\frac{W}{L} \right)_{NVT} \tag{3.7}$$

Here, $i \in \{NVT, MVT, NaVT\}$ and $q_i \in \{0, 1\}$ is the key-bit controlling transistor of type i.

2. **Linking $\tilde{y}$ to the circuit specification of the locked analog IC.** Unlike techniques that obfuscate the bias circuits, analog camouflaging obfuscates the transistors in the design that affect the circuit specification. Considering the fourth-order Gm-C BPF, the transconductance is given by $g_m = \sqrt{2\mu C_{ox} \tilde{y} I_D}$, where $I_D = I_B/2$. Solving equations (A) and (C) for [5] in Table 3.2 gives the required key for the precise operation of the BPF. From this key, the attacker determines the variant of the transistor.

3.2.8 Attack on AMSlock [6]

The AMSlock and Mixlock [37] are the same locking technique, as both use SFLL-HD0/h [9] to lock the digital section of the AMS circuit. The digital optimizer in AMSlock and the digital decimation filter in the Mixlock are locked using SFLL-HD^0 or SFLL-HD^h [9]. These locked circuits receive inputs from the ADC in AMSlock and from the $\Delta\Sigma$ ADC in Mixlock. In [6], the ADC is fed by the analog circuits, such as BPF, LC oscillator, or triangular waveform generator. Whereas, in [37], the $\Delta\Sigma$ ADC is fed with the audio input, which has to be modulated. The defender chooses the PIPs by analyzing the inputs from the ADC or the $\Delta\Sigma$ ADC. Therefore, the attacker determines the PIPs by simulating the analog circuit and ADC in [6] and analyzing the audio signals sent to the $\Delta\Sigma$ ADC in [37]. He/She then uses SAT formulations to determine the correct key. The following section explains our attack on AMSlock [6]. The only difference between these two techniques is the circuit over which the defense is implemented. However, the underlying defense algorithm remains the same. Therefore, the attack which we have shown on AMS lock can break Mixlock too.

The purely digital optimizer controls the value of the passive components, such as R and C, of

the analog circuit-under-protection. This optimizer is locked using the SFLL [9]. For the correct key, it chooses the correct value of these components via the tuning knobs, considering the impact of process variations. However, for an incorrect key, the optimizer does not consider the impact of these variations. Hence, incorrect values are chosen, leading to circuit malfunction.

Attack methodology. This defense thwarts an attacker from overproducing the chip but cannot thwart him/her from modifying the layout of the design and pirate (IP piracy). Hence, the attacker can assume access only to the layout but cannot modify the same. This design has 1024 tuning knob settings corresponding to 1024 unique resistor settings. These tuning knobs are internal to the chip and are not available as top-level ports. Adding to this, the optimal settings of the tuning knobs vary chip to chip due to the process variations. Therefore, the attacker cannot simulate the analog circuit-under-protection for all the tuning knob settings to determine the correct settings as it changes chip to chip. Only the optimizer can control the tuning knob settings. As the attacker cannot modify the tuning knob settings directly, it is necessary to determine the correct key to unlock the locked optimizer.

1. **Obfuscated component.** To reduce the impact of PVT variations and mismatch, passive components, such as R and C, are often implemented as banks of elements to enable calibration. The correct value of the passive components is chosen by the locked optimizer and cannot be computed by analyzing the netlist. Hence, we identify this component as the obfuscated component.

2. **Equation linking the obfuscated component and the key inputs to the optimizer.** In the BPF circuit, the resistors R_1 and R_2 are the obfuscated components. The correct value of these resistors is chosen by two 5-bit tuning knobs, which are controlled by the locked optimizer. Each of the 1024 tuning knob settings corresponds to a unique resistor setting. The output response of the analog circuit can be determined from the transfer function given by $H(s) = \frac{s/(R_1C)}{s^2+s/(R_1C)+1/(R_2^2C)}$. Here, C is the fixed capacitor. The attacker can simulate the output response of the circuit for unique resistor settings, via transistor-level simulations. As there are only 1024 unique tuning knob settings, the analog circuit can have 1024 unique output re-

sponses. The output response is digitized using the analog to digital converter. These digitized output responses are the input patterns (INs) that are fed to the locked optimizer. The optimizer chooses the tuning knobs based on these inputs. These INs are required to determine the SAT formulations for the attack.

3. **Breaking SFLL-HD$^{\mathrm{h}}$ [9].** The SFLL-HD$^{\mathrm{h}}$ technique can have more than one correct key for a PIP when $h > 0$ [9]. If the attacker finds one key that ensures the correct output for all 1024 PIPs, it can be used to unlock the overproduced chip.

   Finding PIPs. The attacker can determine the PIPs in the 1024 INs with the help of oracle. The entire AMS chip loaded with the correct key constitutes the oracle. Only the input and output ports of the analog circuit-under-protection in this chip are available to the attacker. Hence, the attacker has to simulate the analog circuit for different tuning knob settings to determine the input patterns to the locked-optimizer. The signal generator gives the required input to the oracle [85], and he/she can observe the oracle's response on the output port. If the locked optimizer gives an incorrect output for an IN, then it is a PIP. Otherwise, it is not a PIP. Hence, if there are p PIPs out of 1024, then the remaining n patterns $(1024 - p)$ are unprotected IPs.

   SAT formulations. Along with the locked netlist (N_{locked}) and the HD used by the defender, the following are the other constraints added to the SAT formulations.

   (a) The output response (O) of the analog circuit corresponding to each PIP (PIP) is found using the oracle. The corresponding constraint is given by $(PIP_1 \Rightarrow O_1) \wedge (PIP_2 \Rightarrow O_2) \wedge \cdots \wedge (PIP_p \Rightarrow O_p)$.

   (b) HD between each PIP and the key (K) must be equal to h, $\sum_{i=1}^{p} \wedge (HD(PIP_i, K) = h)$.

   (c) HD between other n INs and K should not be equal to h, $\sum_{i=1}^{n} \wedge (HD(IN_i, K) \neq h$.

Combining all the above constraints gives,

$$
\begin{aligned}
& N_{locked} \wedge PIP_p \wedge O_p \wedge (PIP_1 \Rightarrow O_1) \\
& \quad \wedge (PIP_2 \Rightarrow O_2) \cdots \wedge (PIP_p \Rightarrow O_p) \\
& \quad \wedge \sum_{i=1}^{p} \wedge (HD(PIP_i, K) = h) \\
& \quad \wedge \sum_{i=1}^{n} \wedge (HD(IP_i, K) \neq h)
\end{aligned}
\tag{3.8}
$$

Equation (3.8) helps in determining the correct key to unlock the locked optimizer. Section 3.3 presents the results of this attack.

3.2.9 Attack on shared dependencies [7]

This technique improves the resiliency against IP piracy and overproduction by locking the analog and digital parts of an AMS circuit. The AMS circuit-under-protection consists of a sensor, a common-source (CS) amplifier, a 7-bit flash ADC, a peak detection, and a counter circuit. The CS amplifier, the peak detection circuit, and the counter are locked using parameter-biasing obfuscation [2], SFLL-HD^0 [9], and random logic locking (RLL) [27], respectively. A set of N transistors in parallel, $\{x_{11}, x_{12}, \cdots, x_{1N}\}$ replaces the transistor y_1 in Fig. 3.7. Similarly, y_2 is replaced by $\{x_{21}, x_{22}, \cdots, x_{2N}\}$. The single transistor getting replaced with N transistors is similar to the replacement shown in Fig. 3.3(a) and (b) illustrating [1]. This is because combinational lock [1] and parameter-biasing obfuscation [2] are similar and replace a single transistor with multiple transistors. The effective width of y_1 and y_2 is controlled by key ($\mathbf{q}$) of size $2 \times N$. Each of the digital and analog locked sections has a dedicated part of the whole key.

Attack methodology. The attacker has to find the correct key required to unlock the analog and digital parts of the AMS circuit. This technique's threat model assumes an untrusted foundry and an untrusted end-user. Hence, he/she can access the layout, the oracle, and the circuit specification. The attacker targets the analog and digital locks separately.

Breaking the digital lock. The output of the ADC is the input to the peak detection circuit

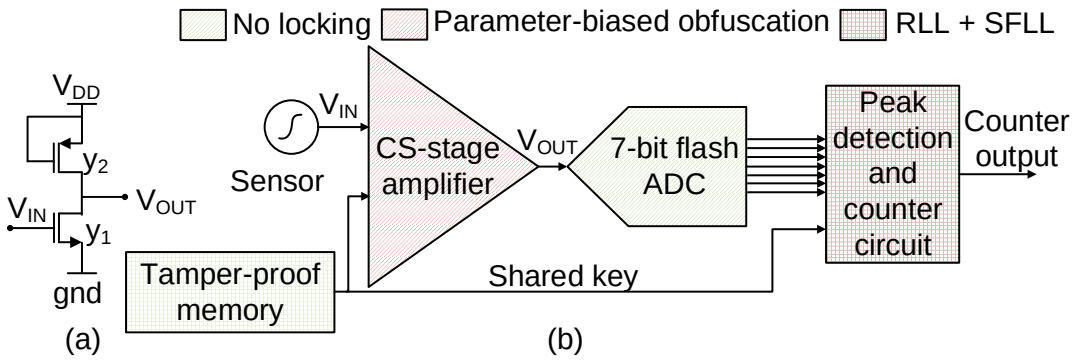

Figure 3.7: (a) Common source (CS) amplifier. (b) Shared dependencies lock [7]. The CS-stage amplifier is locked using parameter-biasing obfuscation technique [2]. The digital circuit is locked using random logic locking (RLL) and stripped functionality logic locking (SFLL) [9].

locked using SFLL-HD^0. This circuit sets the peak detection signal to one when the ADC output is maximum and resets to zero when the ADC output falls below the maximum value. The PIP corresponds to the maximum ADC value. In SFLL-HD^0, as the key is equal to the PIP, the attacker can unlock the locked peak detection circuit. He/She could verify the correctness of the key found, using the SAT formulations in Section 3.2.8. The SAT attack [51] can unlock the counter circuit. Thus, the attacker has determined the key to unlock the digital part of the AMS circuit even without unlocking the locked analog circuit.

Breaking the analog lock. The following SMT formulations help to determine the key to unlock the analog circuit.

1. **Obfuscated component.** The effective aspect ratio of the obfuscated transistors y_1 and y_2 are y_{1_v} and y_{2_v}, respectively. Here, $\tilde{y_{1_v}} = \left(\frac{W}{L}\right)_{y_1} = \sum_{i=1}^{N} q_k \times x_{1i_v}$, where $k \in \{1, 2, \cdots, N\}, \forall i$. $\tilde{y_{2_v}} = \left(\frac{W}{L}\right)_{y_2} = \sum_{i=N+1}^{2\times N} q_k \times x_{2i_v}$, where $k \in \{N+1, N+2, \cdots, 2\times N\}, \forall i$. x_{1i_v} and x_{2i_v} are the aspect ratios of transistors x_{1i} and x_{2i}, respectively. The attacker knows the value of N from the layout.

2. **Equations linking the gain $\mathbf{z_{ob\_comp}}$ with obfuscated aspect ratios $\mathbf{y_{1_v}}$ and $\mathbf{y_{2_v}}$ of the transistors.** The CS amplifier generates the necessary analog input to the ADC. The equation linking the amplifier's gain (G_{CS}) with y_{1_v} and y_{2_v} is $z_{ob\_comp} = G_{CS} = -\frac{1}{1+\eta}\sqrt{\frac{\tilde{y_{1_v}}}{\tilde{y_{2_v}}}}$. Here, η is the

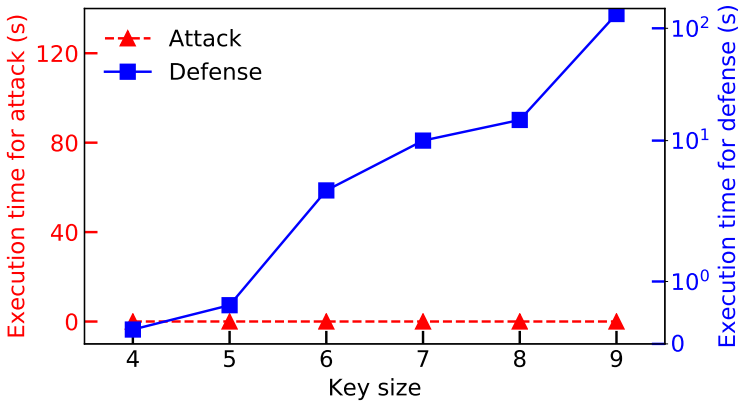

Figure 3.8: For a constant number of transistor switches, the defense time shows an exponential pattern with increasing key size, whereas the attack requires a constant time of $0.01s$.

backgate transconductance available in PDK.

3. **Equation linking the gain $\mathbf{z_{spec}}$ with circuit parameter.** The gain of the CS amplifier is in the specification.

4. **Putting it all together.** Solving the above equations gives the correct key and hence, the effective aspect ratios of the obfuscated transistors.

Using the above formulations, the attacker can find the correct key to unlock the AMS circuit. Section 3.3 gives the attack results on this defense technique.

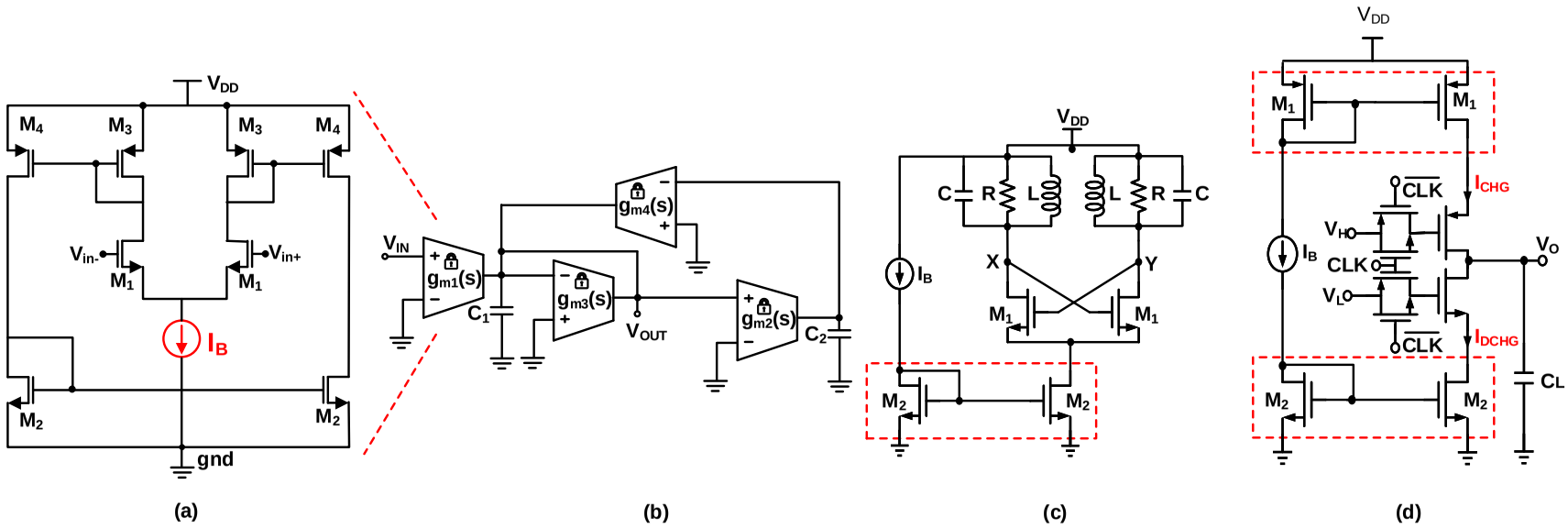

Figure 3.9: (a) Operational transconductance amplifier (OTA) with the locked current bias indicated in red. (b) Second-order Gm-C bandpass filter [1] constructed using four locked OTAs. (c) The current mirror producing the required bias to the LC oscillator is locked using combinational lock [1], as shown by red dashed box. (d) Schematics of Triangular waveform generator (TWG). The two current mirrors generating $\mathbf{I_{CHG}}$ and $\mathbf{I_{DCHG}}$ are locked using combinational lock [1] as indicated in red dashed boxes.

3.3 Attack results

3.3.1 Estimation of unlocked circuit performance

Once the attacker determines the key using the SMT/SAT solvers, this key is applied to the locked circuit. As the attacker has layout or the reverse-engineered netlist as indicated in 3.1, we have considered access to layout to demonstrate our attacks. In a real attack, the output response of the unlocked circuit is measured using an oscilloscope. This response is then compared with the oracle's response to check the correctness of the deduced key. In this work, the simulations are based on transistor-level netlist from the layout available in the foundry. The attacker simulates this netlist, which does not have any deviation from the defender's design. The extracted netlist comprises the analog circuit-under-protection and the locked bias circuitry. He/She also has access to the reverse-engineered netlist, as mentioned in Table 3.1. As RE is an imprecise and expensive process, the extracted netlist may not be accurate with respect to the dimensions of the transistors and passive components. As there can be differences in the output responses of the circuit via simulations of the extracted netlist, the impact of imprecise RE on our attack is included in Section 3.4.

3.3.2 Experimental setup

The transistor-level schematics of the analog and AMS circuits used to evaluate the proposed attack are based on the IBM 180nm technology library using Cadence Virtuoso. Our attack is demonstrated on:

1. the OTA, fourth-order Gm-C BPF, quadrature oscillator, LC oscillator, and triangular waveform generator (TWG) used in class-D amplifiers locked using combinational locks [1] and parameter-biasing obfuscation [2].

2. the BPF, LNA, and LDO locked using AMS lock [6].

3. the OTA locked using analog performance locking [4].

We use iSAT3 solver [83] and pycosat [86] to solve all the SMT-based attack formulations and

SAT-based attack formulations, respectively. These experiments are run on x86 architecture, 64-bit Intel Xeon CPU processor with 16 cores per socket and two threads per core.

3.3.3 Attack results on combinational locks [1]

Attack and defense time. In the combinational locking [1], the equations relating the key inputs, I_{ref}/V_{ref}, I_B/V_B, and the transistor sizes are generated by the SMT solver. The defense time is the time the solver takes to generate these equations. Similarly, the attack time in Table III corresponds to the time the SMT solver takes to solve the attack equations and determine the unique key required to unlock the current mirrors. The time taken to formulate the attack equations is not taken into account, as it is done manually. Fig. 3.8 illustrates the attack and defense times on a combinational locked OTA for increasing key sizes. It is evident from the figure that the attack time is constant compared to the defense time. The attack time does not depend on the key and matrix size. In the combinational lock [1], the SMT constraints for the defense, sizes the transistors such that only one key gives the required I_B. For all other key vectors, it gives an incorrect I_B value. Hence, it requires the enumeration of each key combination in the input constraints to the SMT solver. The defense time also depends on other factors such as the key size, I_{ref}, I_B, Δ, and θ, as explained in [1].

As the number of SMT constraints increases exponentially with respect to the key size, the time taken also increases exponentially, as shown in Fig. 3.8. The results are illustrated on smaller key sizes to show the defense and attack time trends. The maximum key size on which we execute our attack is dependent on the locking technique used. Hence, we could not increase the key size to more than 15 bits for circuits that require only two current mirrors such as LC oscillator, triangular waveform generator, and quadrature oscillator. However, this attack can successfully determine the keys of sizes 80 – 512 bits, as shown in Table 3.3, 3.4, and 3.5. In this attack, the number of calls to the SMT solver is equal to the number of keys that satisfy the attack equations plus one. Each call returns one key, and the last call returns no solution once all the keys are found. The number of circuit simulations is equal to the number of keys that satisfy the attack equations.

Fourth-order Gm-C bandpass filter. It has two second-order BPFs shown in Fig. 3.9(b) in cas-

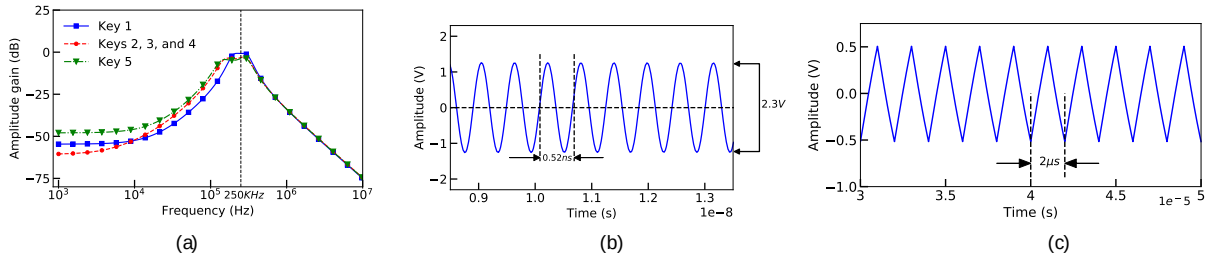

Figure 3.10: (a) BPF simulations for the keys found by proposed attack on an 84-bit key combinational lock [1]. The attack returns five keys, in which key 1 is the correct key. The output response shows a $2dB$ degradation in the gain at the $\omega_{\mathbf{c}}$ for remaining keys. (b) Attack returns exactly one key for LC oscillator locked using combinational lock [1]. (c) Time period of oscillation for the unlocked triangular waveform generator is $\mathbf{2}\mu$s.

cade. It is characterized by $f_c = 250kHz$, $BW = 150kHz$, the transition band $= 200kHz$, and the amplitude gain $= 0dB$. The leading second-order BPF is implemented with the capacitance $C_{11} = C_{12} = 78.95pF$, $f_{c1} = 201.6kHz$, and $BW_1 = 83.6kHz$. The succeeding one has $C_{21} = C_{22} = 51.34pF$, $f_{c2} = 310kHz$, and $BW_2 = 128.5kHz$. Each of the second-order BPF contains four OTAs. The CCM feeds each of the OTAs with the required I_B, as shown in Fig. 3.3. The 12 CCMs in the BPF has a key size of 84 bits. Table 3.3 lists the size of the transistor switch matrix, replacing the mirroring transistor in each current mirror.

Here, C_1 and C_2 are the capacitances. g_{m1}, g_{m2}, g_{m3}, and g_{m4} are the transconductance of OTAs 1, 2, 3, and 4, respectively. From the above equations, we can infer that f_c, G_{BPF}, and Q are dependent on the transconductance g_m. Using the circuit specification and the equations listed above, the attacker can calculate g_{m1} and g_{m3}. Hence, from the equation, $g_m = \sqrt{2\mu C_{ox}\left(\frac{W}{L}\right) I_D}$, where $I_D = I_B/2$, I_{B_1}, and I_{B_3} corresponding to OTAs 1 and 3 can be calculated. Though he/she does not have the information to calculate I_{B_2} and I_{B_4} individually, he/she can calculate the product of these bias currents via the product of transconductances ($g_{m2} \times g_{m4}$). Therefore, I_{B_1}, I_{B_3}, and $I_{B_2} \times I_{B_4}$ are provided to the SMT solver along with equations (A) and (B) listed in Table 3.2 for each of the second-order BPF. The equations corresponding to all the CCMs are solved together.

Though the defense's cumulative time is approximately 22 hours, our attack obtains the key even without knowing the ranges for all the bias currents in $0.829s$, as shown in Table 3.3 for [1].

As f_c and Q depend on $g_{m_2}g_{m_4}$ rather than individual g_m, the attack equations return five possible keys (which include the correct key $\mathbf{q}^*$ from defense). The attacker hence reduces the key search space from 2^{84} to five using this attack. The BPF is simulated for the five different key inputs reported by the attack formulation in Fig. 3.10(a). Key 1 shows the response of the correct key, where $f_c = 250KHz$ with an amplitude gain of $0dB$. As analog circuits are sensitive to biasing conditions, we could see a degradation of $2dB$ near the center frequency for keys 2, 3, 4, and 5. As mentioned in Section 3.2.1, the attacker, as an end-user, has access to the unlocked chip. He/She can compare the output responses of this circuit for the deduced keys with the oracle's response. This comparison helps in finding the correct key. Here, the analog circuits are not simulated for all the key combinations. They are simulated only for the keys reported by the SMT solver after solving the attack equations. Hence, even if the time taken to simulate the analog circuits increases for larger circuits and larger key sizes, its impact on the overall attack time is limited. However, note that this attack can prune a large set of keys into a handful ones, thereby reducing the number of simulations significantly. This process can be automated using the analog simulation tools offered by companies, such as Cadence, Synopsys, and Mentor Graphics.

LC oscillator produces a clock signal with the oscillation frequency $f_{osc} = 1/2\pi\sqrt{LC}$ [80], where L is the value of the inductor and C is the capacitance. An LC oscillator is designed with $f_{osc} = 2GHz$ and the amplitude of the output oscillation $V_o = 2.3V$. The defender protects V_o, which is equal to $4I_B(w_{osc}L)^2/\pi R_s$, by locking the current mirror that generates I_B, as shown in Fig. 3.9(c). Table 3.3 lists the size of the switch matrix for the individual CCMs and the time taken to find the sizes of the switch transistors for [1]. Here, $\omega_{osc} = 2\pi f_{osc}$, and R_s is the series resistance. To obtain the correct key, the attacker gathers the values of L, C, and R_s using the locked netlist and PDK. He/She determines the value of V_o and w_{osc} from the circuit specification.

Due to the SMT constraints in defense explained in Section 3.3.3, the key size cannot be more than 14 bits. Furthermore, the CCMs are in cascade. Hence, the attacker also includes $I_{ref_2} = I_{B_1}$ in the SMT formulations. Solving these equations along with the corresponding equations in Table 3.2, gives the correct key to unlock the current mirrors in $92ms$. The values of the bias

currents from the unlocked CCMs are equal to the original circuit's values, thereby making the chip functional. Fig. 3.10(b) shows the simulation results indicating the frequency of operation and output voltage swing for the identified key. These values match with the circuit specification, thereby demonstrating a successful attack.

Triangular waveform generator (TWG). In class-D amplifiers, the output of the pulse-width modulated signal depends on the frequency of charging (I_{CHG}) and discharging (I_{dCHG}) ramps of the carrier signal generated by the TWG [81] illustrated in Fig. 3.9(d). The carrier signal has the frequency of $f_{TRI} = 500KHz$ with the output capacitor, $C_{TRI} = 100pF$, maximum voltage $V_H = 500mV$, and minimum voltage $V_L = -500mV$. As this carrier signal significantly impacts the total harmonic distortion of the amplifier, the defender locks the current mirrors of the TWG with a 15-bit key. Table 3.3 lists the switch matrix sizes and the upper and lower bounds of I_B for each CCM. The attacker finds the relationship between the key inputs and I_B from the locked netlist using the equations (A) and (B) corresponding to [1, 2] in Table 3.2. From the circuit specification, the attacker can determine T_{TRI}, C_{TRI}, V_H, and V_L. Thus, he/she can formulate the equation to calculate the charging (I_{Chg}) and discharging (I_{DChg}) currents: $T_{TRI} = C_{TRI}(V_H - V_L)\left(\frac{1}{I_{Chg}} - \frac{1}{I_{DChg}}\right)$. As shown in Table 3.3, though the cumulative time for defense is 31 hours, our attack takes only $95ms$. Fig. 3.10(c) shows the time period of the triangular waveform for the key found by the attack. It is equal to $2\mu s$, which is the desired time period. Thus, the key returned by the attack is indeed valid.

Quadrature oscillator. The center frequency of the oscillations is given by $f_c = 2.34MHz$. The oscillator consists of two OTAs. The transconductance (g_m) of the OTAs is set to $1mA/V$. There are two current mirrors locked using [1] with only a 12-bit key, due to the SMT constraints in defense explained Section 3.3.3. Table 3.3 lists the sizes of the switch matrix, time taken to find the sizes of the transistor switches, and the upper and lower bound values of the I_B. The attacker can obtain the value of I_{ref} and f_c from the circuit specification. Using the reverse-engineered netlist, he/she obtains the value of W and L of the transistors in the OTAs. He/She can then calculate the product of bias currents required by the OTAs and provide this constraint to the SMT solver, along

with the equations corresponding to [1, 2] in Table 3.2. As illustrated in Table 3.3, the attacker can find the key in $95ms$, even without knowing the individual I_B values.

3.3.4 Attack results on parameter-biasing obfuscation [2]

The defense and attack time calculations are similar to the calculations explained in Section 3.3.3 for the combinational locks [1]. In [1], the transistors are sized such that only one key gives the required bias, and the bias is out of range for all other keys. However, this technique chooses the transistor sizes randomly and does not have a constraint on the number of keys that give the desired bias. Hence, the defense time depends only on the ratio of I_B to I_{ref} and not on the key size. This dependency enables the creation of a lock with a key size of 512 bits, as shown in Table 3.3. As this technique can give the desired I_B for more than one key, the attack returns more than one correct key. However, the SMT solver is not called iteratively to find all possible keys. The run is exited as soon as one of the valid keys is found. Table 3.3 shows the attack time required to find one of the correct keys.

Table 3.3: Combinational lock [1] and parameter-biasing obfuscation [2] defense and the proposed attack results for Bandpass filter (BPF), LC oscillator (LC osc.), quadrature oscillator (Quad osc.), and triangular waveform generator (TWG). For each incorrect key, the configurable current mirror in the defense setup is configured such that I_B is either $< (1-\Delta)I_B(I_{B_{min}})$ or $> (1+\theta)I_B(I_{B_{max}})$, where $\Delta = 0.8$ is the lower bound and $\theta = 3$ is the upper bound on I_B. instance (inst).

Benchmark	**Circuit details**							**Defense**				**Attack**								
				[1]		**[2]**		**[1]**			**[2]**	**[1]**							**[2]**	
	# CM inst	$\mathbf{I_{ref}}$ (μA)	$\mathbf{I_B}$ (μA)	**Matrix size**	**Key size**	**Matrix size**	**Key size**	$\mathbf{I_{B_{min}}}$ (μA)	$\mathbf{I_{B_{max}}}$ (μA)	**Time reqd. (s)**	**Time reqd. (s)**	**Individual CM I (μA)**					**Time reqd. (s)**	**# of calls**	**Time reqd. (s)**	**# of calls**
												$\mathbf{I_{B1}}$	$\mathbf{I_{B2}}$	$\mathbf{I_{B3}}$	$\mathbf{I_{B4}}$	$\mathbf{I_{B5}}$				
BPF	1	0.1	2	2×6	84	–	512	0.4	8	1351		2	2	2	2	2	0.829	6	0.158	1
	2	2	38	2×6		1×64		7.6	152	1080	3	38	38	38	38	38				
	3	0.1	2	2×6		–		0.4	8	1351	–	101	2	2	2	2				
	4	2	112	2×6		1×64		22.4	448	9893	2	606	10	10	10	112				
	5	1	20	2×5		1×64		4	80	8	9	20	20	20	20	20				
	6	1	38	2×5		1×64		7.6	152	1249	3	7	420	425	425	38				
	7	0.1	4	2×6		–		0.8	16	15	–	4	4	4	4	4				
	8	4	112	2×5		1×64		22.4	448	1260	7	112	112	112	112	112				
	9	1	56	2×6		1×64		11.2	224	27300	4	56	56	56	56	56				
	10	0.5	20	2×6		1×64		4	80	16980	3	20	20	20	20	20				
	11	0.1	2	2×5		–		0.4	8	8	–	2	2	2	2	2				
	12	2	56	2×7		1×64		11.2	224	18060	19	56	56	56	56	56				
LC osc.	1	2	30	2×6	14	1×256	512	6	120	240	1983	20					0.092	2	0.184	1
	2	30	270	2×7		1×256		54	1080	5940	270	270								
TWG	1	2	100	2×6	15	1×256	512	20	400	86400	170	100					0.095	2	0.212	1
	2	2	100	2×7		1×256		20	400	25200	1079	100								
Quad osc.	1	4	280	4×3	12	1×256	512	56	1120	76800	413	280					0.094	2	0.314	1
	2	4	556	4×4		1×256		111.2	2224	50	35	556								

3.3.5 Attack results on memristor-based protection [3]

The resistance value programmed into the memristor depends on the programming pulse voltage V_{PROG}, its duty cycle ρ, its frequency ω_{VT}, and the initial resistance value in the memristor. For a 2×2 crossbar, the designer has programmed the resistivity of all the memristors in the crossbars with $2k\Omega$, the amplifier is designed with the gain A of 5, and V_{PP} is set to $1V$ to produce $V_{PROG} = 2.5V$. The attack returns the correct 8-bit key in $0.012s$. This experiment is repeated for increasing crossbar sizes and hence, increasing key sizes. Fig. 3.6 shows that even for a 7×7 crossbar with a key size of 98 bits, the attack time is as low as 0.3 seconds. The attack returns the required key and the resistance values which have to be programmed into each memristor in the crossbars.

3.3.6 Attack results on analog performance locking [4]

We executed the SMT-based attack on the ANN core of various sizes. The equation linking the I_B or V_B and the circuit specifications of the analog circuit protected such as gain is determined. For example, the equation linking the gain of BPF with the I_B is, $G_{BPF} = \frac{g_{m_1} C_1}{g_{m_3} C_2}$ and $I_B = \frac{g_m^2}{\mu C_{ox} \frac{W}{L}}$. Once the required I_B is calculated, it is given to the SMT solver along with the mathematical model of the ANN shared in Section 3.2.6. The solver returns the input voltages and the weights of each synapse required to produce the necessary I_B or V_B. The defense time is the time required to determine the synapse weights, the neuron types, and the required input voltages for the given neural network size and the required I_B or V_B. Similarly, the attack time corresponds to the time taken to determine the synapse weights, the neuron types, and the required input voltages for the given neural network size and the required I_B or V_B range, as tabulated in Table 3.5. The SMT solver is called once to determine the weights of the synapse, type of the neurons, and the input voltages to the neural network to provide the necessary I_B or V_B.

3.3.7 Attack results on camouflaged analog IPs [5]

We demonstrate this attack on fourth-order Gm-C BPF using TSMC 180nm multi-V_{th} technology. The size ratios of each of the multi-V_{th} transistors with respect to NVT transistors are 1, 0.65,

and 0.39, respectively. If the current mirror transistors are camouflaged, both the reference and the mirroring transistor have the same V_{th} as their gates are shorted. All the 80 transistors in the BPF are camouflaged. Transforming this netlist into a logic-locked netlist has 3^{80} key combinations with a key size of 160 bits. Apart from the 16 transistors which affect the circuit specification, there are 64 transistors in the current mirrors. The V_{th} type of the 16 transistors can be found by solving equations (A) and (C) listed in Table 3.2 for [5]. The V_{th} type of the current mirror transistors is not necessary to determine the V_{th} type of the other transistors, as the attacker can get the size ratio between the reference and mirroring transistors from the reverse-engineered netlist. The reference and mirroring transistors in the current mirrors are considered to be connected to the same key. Hence, there are effectively only 32 transistors. Each of the transistors can be one of the three variants. The total number of possible keys is 3^{32}. However, our attack returns only one correct key. Thus, our attack is also effective against current analog camouflaging [5].

3.3.8 Attack results on AMSlock [6]

The attack is demonstrated on three analog circuits: a BPF, an LNA, and an LDO locked using [6]. The optimizer is locked using SFLL-HD^0 and SFLL-HD^h. The input size of the optimizer is n, the key size is k, and the Hamming distance is h. The effective key size is given by $k - \log_2 \binom{k}{h}$. This should be >80 for SAT attack resilience. As the locking of the digital optimizer is done manually in the RTL level, the defense time is not included in Table 3.4. The attack time required to break each of the setups is given in Table 3.4. It considers the time taken by the SAT solver to determine the correct key. As tabulated in Table 3.4, the SAT solver is called only once for all the circuits to determine the correct key to unlock the digital optimizer. For SFLL-HD^0, the number of PIPs is equal to one, and hence, $PIP = key$. Thus, Equation (3.8) can be reduced to $CNF_{locked} \wedge PIP \wedge O$. The attack time for increasing key size is listed in Table 3.4.

In the case of SFLL-HD^h, the number of PIPs is given by $2^{n-k} \times \binom{k}{h}$. Also, for an n-bit input size, there are 2^n possible INs to the locked optimizer. However, as the analog circuit controls the input to the locked optimizer, there are only 1024 INs. Owing to this constraint, the attacker knows only a subset of the PIPs, but not all. Hence, there can be more than one correct key, which gives

Table 3.4: Attack results on AMS locked circuits [6]. h denotes Hamming distance.

Benchmark	Hamming distance = 0				Hamming distance = h				
	Key size	Effective key size	Time taken (s)	# of calls	h	Key size	Effective key size	Time taken (s)	# of calls
BPF	87	87	1.26	1	7	220	177	90	1
	112	112	1.9	1	15		144	58	1
	220	220	1.5	1	20		126	11507	1
LNA	81	81	1.74	1	4	154	129	35	1
	84	84	1.49	1	11		99	55	1
	154	154	1.78	1	21		68	444	1
LDO	109	109	2.38	1	7	234	191	84	1
	135	135	2.52	1	14		160	103	1
	234	234	2.96	1	28		114	85140	1

the correct output for all the PIPs in the 1024 INs. As the recent attacks on SFLL-HD^h such as SFLL-hd – Unlocked [72] and FALL attack [73] requires all the PIPs in the attack formulation, we cannot reuse these attacks. Therefore, we use the attack formulations given in Equation (3.8) to find the correct key. This correct key ensures the correct output for the 1024 INs. As the Hamming distance between the key and PIP increases from 0 to $\frac{k}{2}$, where k is the key size, the number of PIPs protected by this key also increases (as indicated in Fig. 6 of [6]). This is calculated using the formula, $2^{n-k} \times \binom{k}{h}$ in [9]. Here, n and h are input size and Hamming distance, respectively. This means that for a given m number of PIPs, the number of keys that are at the same HD from these m PIPs will reduce. Hence the time taken to find out these keys from 2^k possible keys increases. To deduce the key using the attack formulations shared in Section 3.2.8, an SMT solver or a SAT solver can be used. However, the increasing trend in the attack time is independent of the type of solver used and is rather dependent on the defense technique (SFLL-HD^h).

3.3.9 Attack results on shared dependencies [7]

As this technique includes locking of the analog and digital sections, the defense time to lock the analog section is given in Table 3.5. However, as explained in Section 3.3.8, the time taken to lock the digital section is not shared as that process is not automated. The attack time is the sum of the times required to unlock the analog and digital sections individually. The attack time

to unlock the analog section is the time taken by the SMT solver to solve the attack equations and determine the key. The attack time to unlock the digital section is the time taken by the SAT solver to determine the correct key. As the analog section is locked using the parameter-biasing obfuscation technique, the SMT solver is called only once to determine one of the correct keys. Likewise, for the digital section locked using SFLL, the SAT solver is called once to determine the correct key, as indicated in Table 3.5. The defense technique illustrated in Fig. 3.7 is implemented for different key sizes given in Table 3.5. If the total key size is equal to 160 bits, 80 bits exclusively control the analog lock and 80 bits exclusively control the digital lock.

The attack formulations to find the key to unlock analog circuit is the same as the formulations given in Section 3.3.4. The time taken to determine the key required to unlock the analog circuit is almost constant, irrespective of the key size. As the digital circuit is locked using RLL and SFLL-HD^0, we assume the entire key to the digital circuit is bifurcated to RLL and SFLL, as shown in the Table 3.5. The circuit locked using RLL can be compromised using the SAT attack. The circuit locked using SFLL can be unlocked using the formulations shared in Section 3.2.8. The peak detection circuit generates a pulse whenever the 7-bit input from the ADC is maximum (7'h7F). This peak detection circuit is locked using SFLL-HD^0. The PIP is the maximum value of the input from the ADC. The time taken to determine the SFLL key is almost constant, as the Hamming distance $h = 0$.

3.4 Discussion

What is the impact of the reduced current range in combinational locks [1] on the attack? The I_B value is outside the range $(I_{B,lo}, I_{B,hi})$ for an incorrect key, where $I_{B,lo}$ is the minimum I_B and $I_{B,hi}$ is the maximum I_B. To ensure the attacker suffers from significant error (e.g., denial of service), the defender chooses the bias range widely. Therefore, he sets $I_{B,lo} = 0.2I_B$ and $I_{B,hi} = 4I_B$. The attacker finds the bias range using the circuit specification. However, this range is a subset of the bias range designed by the defender, e.g., $(0.9I_B, 1.1I_B)$. Hence, as long as this bias relationship is true, our attack can find the correct key.

We attack the locked circuits with the reduced range set by the defender to stress-test our

proposed methodology. In Fig. 3.11, as the bias range reduces, the number of keys found by the attack increases. For the defense setup on BPF, where the defender's bias range is ($0.8I_B$, $1.2I_B$) and the range determined by the attacker is ($0.7I_B$, $1.3I_B$), the attack returns the keys whose corresponding bias values are in the range ($0.7I_B$, $0.8I_B$) and ($1.2I_B$, $1.3I_B$). For a key size of 56 bits, the search space is reduced from 2^{56} to 1190, as illustrated in Fig. 3.11. The attacker can now brute-force the determined keys to find the correct key using the oracle's response. This experiment is repeated for increasing ranges in the I_B determined by the attacker using the circuit specification. As the range increases, more number of keys are reported. Thus, based on the knowledge the attacker has on the analog circuit, he can determine the allowable I_B range and prune the unwanted keys.

What is the impact of approximate models in bias calculations? In this work, the quadratic expression models are used to estimate the I_B. Though these estimations have considerable inaccuracies compared to the charge-based model like an advanced compact model (ACM) and weak/moderate/strong inversion models of the transistors, it does not impact the attack results on the combinational lock [1]. This is because the defender's bias range is much larger (20% to 400% of I_B) compared to the range estimated by the attacker. Also, in parameter-biasing obfuscation [2], memristor-based protection [3], shared dependencies [7], and analog performance locking [4], the

Table 3.5: The attack information is tabulated for shared dependency [7] and analog performance locking [4]. The number of calls to the SMT solver and the SAT solver is equal to one for all the key sizes shown for [7]. The number of calls to the SMT solver for all the analog neural network (ANN) sizes shown is equal to one.

Shared dependency [7]							**ANN [4]**		
Total key size	**Analog**			**Digital**			**ANN size**	**Attack time (mins)**	**Bias voltage**
	Key size	**Defense time (s)**	**Attack time (s)**	**RLL key**	**SFLL key**	**Attack time (s)**			
160	80	0.642	0.101	40	40	0.03	20×20	3	0.705
200	100	0.943	0.113	50	50	0.035	30×30	18	0.425
240	120	1.268	0.122	60	60	0.06	40×40	134	0.872
280	140	1.622	0.127	70	70	0.1	50×50	1243	0.4
320	160	2.183	0.127	80	80	0.1	60×60	3383	0.11

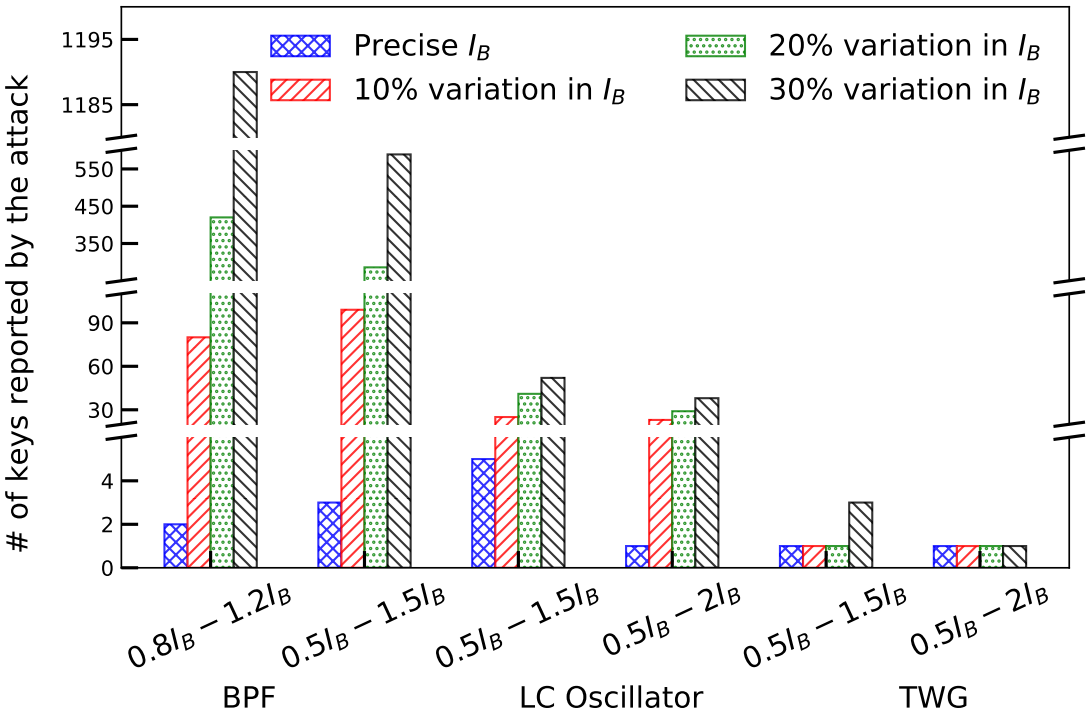

Figure 3.11: Effect of reduced current range in the combinational locked circuits [1].

attack returns the keys which produce any bias value within the bias range calculated by the attacker. As this range is calculated from the circuit specifications, the unlocked circuits' output response adheres to the circuit specification. Hence, our attack is able to determine the correct key even by using the approximate models.

What is the impact of imprecise reverse engineering (RE) on the attack? RE is an expensive and error-prone process. This process is based on etching each layer of the chip and taking high-resolution images of each layer. These images are annotated to get the reverse-engineered netlist. To the best of our knowledge, there are no published works that mention the RE process's inaccuracy. The standard RE process involves the same wet or plasma etching for delayering each metal layer [87] that is used in layout designing. Hence, we consider the minimum resolution in the layout designing as the impreciseness in the RE process. For the 180nm technology node, the minimum resolution is 10nm. Hence, the variation in the transistors' physical dimensions due to errors in RE is considered equivalent to $\pm$ 5.56%. Hence, the impreciseness is modeled by giving

Table 3.6: Number of keys reported based on the % variation in the transistor dimensions.

Circuit	Key size	Imprecise RE	
		2% variation	**5% variation**
BPF	84	18,866	20,008
LC osc.	14	100	637
Quad osc.	12	1	4
TWG	15	1	1

error ranges of 2% and 5% to the transistors' physical dimensions. As the number of transistor switches increases, the number of keys reported by our attack also increases, e.g., BPF, as shown in Table 3.6. Therefore, under the impact of RE, though our attack cannot find the correct key, it can considerably reduce search space. The attacker finds the correct key from the determined keys by brute-forcing, as he/she has access to the oracle.

Table 3.7: The attack success on various analog locks. ○ denotes locked netlist. ● denotes the oracle access. ⊥ denotes that the resource availability aiding overproduction but does not aid piracy. ⌀ denotes reverse-engineered locked netlist. ∗ denotes availability of digitally controlled current source. ✓ denotes that the defense is broken. ≈ denotes the attack reduce key search space.

Type	Defense	Circuit locked	TM (defense)	Claimed resilience	TM (attack)	SMT?	SAT?	Broken?
Analog-only locks	Combinational lock [1]	analog	○ ●	IP piracy	○ ●	✓	×	✓
	Parameter-biasing obfuscation [2]		○ ●	IP piracy	○ ● ⊥	✓	×	✓
	Memristor-based protection [3]		○ ●	IP piracy	○ ● ⊥	✓	×	✓
	Analog camouflaging [5]		⌀ ●	Reverse engineering	⌀ ●	✓	×	≈
	Analog neural network [4]		○ ●	Illegitimate access	○ ● ∗	✓	×	✓
AMS locks	AMS lock [6]	digital	○ ● ⊥	Overproduction	○ ● ⊥	×	✓	✓
	Shared dependencies [7]	analog and digital	○ ●	IP piracy	○ ● ⊥	✓	✓	✓

What is the impact of PVT variations on the attack? The minimum and maximum values of the circuit parameters, such as f_c, Q, and ω_{osc}, available in the circuit specification are considered in our attack algorithm, as given in Equation (3.4). These values are calculated by the designer considering the impact of PVT variation. Hence, the attacker need not worry about the PVT impact in the attack equations. Note that the designer designs the circuit and the key such that the correct key should yield the desired performance, even in the presence of variations. Otherwise, he/she cannot sell on the market. Therefore, we also assume that variations in the output responses of the oracle are not possible beyond the ranges quoted in the specification. Therefore, the key found based on the bias range calculated using these specifications by the attacker should give the desired circuit performance. The correctness of this key is verified by comparing the unlocked circuit's output response with the oracle's response.

Can the attacker design the circuit from scratch instead of attacking? This design process is cumbersome and time-consuming. When the attacker knows only the expected performance and the specifications, it requires a process with many iterations to build a functional chip. Given that the design process heavily depends on expertise, skills, and even luck (the number of iterations required to get the desired performance), it is difficult to estimate the design time. Thus, it is easier for an attacker to "steal" an existing design, which is what many of the existing defense techniques aim to prevent.

Can the attacker use approximate attacks? Assume the attacker has the resources to pirate or overproduce the design. If he obtains $I_{B,lo}$ and $I_{B,hi}$, can he set the required bias as the average of these values without using our attack? In analog circuits, the bias is chosen based on the trade-off between multiple specifications. The required I_B need not be the average of $I_{B,lo}$ and $I_{B,hi}$. For example, if the designer is concerned more about power consumption, he must choose a value close to $I_{B,lo}$. The precise selection of I_B is required as the analog circuits are sensitive to them. For instance, in the fourth-order BPF experiment, one of the keys determined by the attack, satisfies the required value of the product of two bias currents. However, it does not satisfy the individual values of each I_B as per the specification. This causes a dip of $2dB$ in the frequency response.

Thus, the proposed attack is essential in determining the correct key and the corresponding I_B.

Can this attack help an attacker who can pirate or overproduce the chip to find the correct key? In the case of IP piracy, assume the attacker obtains the precise value of the bias using Equation (3.3). The locked bias circuit can be replaced with a circuit generating this required bias input. This replacement makes the analog circuit functional. However, if he obtains only the range of the bias input using Equation (3.4), he cannot replace the locked circuit as he does not know the precise I_B value. Hence, the SMT solver is fed with the equations (3.1), (3.2), and (3.4) to return the correct key and the correct bias value. The attacker can then manufacture the chip replacing the CCM with the current mirror that generates the required I_B, thereby enabling him to pirate the chip. Now, let us consider overproduction. Here, the attacker can only overproduce the chip, but netlist level modifications are not feasible. He either obtains the precise value or the range of the bias. The corresponding equations are solved using SMT solver to find the correct key. Hence, our technique unlocks the chip even if the attacker has little control in the foundry.

What is the implication of this attack on various analog locks? None of the defenses are resilient against IP piracy, as shown in Table 3.7. **In the combinational lock,** each chip has a unique user key. This key is XORed with the input from the chip identification unit, such as physically unclonable functions, to produce the common key. This common key controls the CCM [1]. The SMT formulations in Section 3.2.4 helps in determining the common key using the locked netlist and circuit specification. However, to remove the dependency on the unique key, the attacker has to remove the chip identification unit from the layout. Hence, the attacker should have the necessary resources to modify the layout. Unlike [1], the **parameter-biasing obfuscation [2] and memristor-based protection [3]** do not have two sets of keys. The SMT formulations in Section 3.2.4 and Section 3.2.5 help in determining the key for [2] and [3], respectively. The resources required to overproduce the chip are sufficient to break this technique.

In **analog camouflaging [5]**, the attacker does not have access to the foundry, and hence, our SMT formulations can only reduce the search space. The **analog performance locking [4]** can be broken only if the attacker can have access to a standalone digitally-controlled current source,

which is integrated into the ANN. This helps in programming the precise weights in the synapses. Our SAT formulations in Section 3.2.8 can determine the correct key using the overproduction threat model, thus compromising the **AMSlock** [6]. This defense claims resilience against overproduction. In **shared dependencies [7]**, the SAT formulations are used to determine the correct key to unlock the digital part, and SMT formulations are used to unlock the analog part. Though the attack claims resilience against IP piracy, it could be broken without any manual modification in the layout (mask) of the locked chip.

3.5 Conclusion

Our SMT attack has shown the vulnerabilities in the existing analog-only locking techniques [1–5], thereby enabling the attacker to find the key to pirate, overproduce, or reverse engineer a chip. The attack time does not depend on the key size; we demonstrated our attack on a 512-bit key. Additionally, we extended this attack on camouflaged analog IPs [5], where we could reduce the key search space from 3^{80} to 3^{32}, thereby enabling the brute-force attack to find the correct key. We can also successfully break AMS locking techniques [6, 7, 37] by providing the required SAT formulations to compromise the SFLL-HD^{h} lock used in securing the digital part of the AMS circuits. Analog security being a fast-growing field, has two new defenses, [88] and [89]. The former proposes a technique to secure programmable analog ICs, while the latter prevents unauthorized access to analog ICs using floating gate transistors. We shall evaluate the resilience of these techniques in our future works. Also, while digital locking techniques have now obtained reasonable solutions against oracle-based attacks, we urge the community to undertake a theoretical approach in developing defenses for analog circuits.

4. SECURING CLOUD FPGAS AGAINST POWER SIDE-CHANNEL ATTACKS: A CASE STUDY ON ITERATIVE AES

4.1 Introduction

4.1.1 Security vulnerabilities in cloud FPGAs

Hardware accelerators based on the field programmable gate arrays (FPGAs) support parallel processing and have better performance and power efficiency than CPUs and GPUs, respectively. Owing to these performance benefits, they are best suited for deploying compute-intensive tasks such as big data analytics, real-time video processing, and genomics research [90]. Hence, to meet user demands, cloud servers such as Amazon Web Services (AWS), Microsoft Azure, IBM Cloud, and Texas Advanced Computing Center (TACC) have replaced CPUs with FPGAs in their cloud infrastructures [90–94]. The use of FPGAs in cloud has dramatically improved their computing capabilities [90, 95]. However, an FPGA instance can be underutilized when allocated to a single user, leading to FPGA resource wastage. Therefore, the cloud servers can deploy multi-tenanting to avoid this wastage. Multi-tenanting allocates each FPGA instance to more than one user leading to increased FPGA utilization and profit margin [96]. However, in hindsight, it introduces several security vulnerabilities in cloud—the attackers leverage these vulnerabilities to perform several attacks, namely, power side-channel (PSC) attack [10, 13, 22, 23, 97, 98], fault attack [15, 18, 99–101], and covert channel attack [12, 102–104].

Researchers have proposed several works demonstrating PSC attacks on cloud FPGAs. The works [22] and [23], demonstrate the correlation power analysis (CPA) attack on a 128-bit AES. Likewise, the work [10] demonstrates a simple power analysis (SPA) attack on Rivest, Shamir, Adleman (RSA) for a threat model considering cloud FPGAs. Similarly, there have been works proposing defenses to thwart PSC attacks on AES [8, 22, 24]. However, it is important to note that to perform any valid/accurate security assessment, it is imperative that we consider realistic scenarios of the underlying application and also have access to attack vectors that compromise the

security guarantees of the underlying application. Hence, in the third and final piece of work in this dissertation, we focus on addressing the challenges in the existing attacks and defenses in cloud FPGA domain. The following section discusses these challenges.

4.1.2 Challenges in power side-channel (PSC) attack on cloud FPGAs

Here, we discuss the specific challenges that we encounter for PSC attacks on cloud FPGAs. The two primary challenges in the existing attack techniques are (i) repeatability of the attack, and (ii) the impact of sensor design and placements. These challenges are discussed next in detail.

Repeatability. To demonstrate the efficacy of any defense technique, we need an attack technique that is 100% repeatable, i.e., launching the attack every time should retrieve the entire key. If the attack is not repeatable, then the increase in minimum traces for disclosure (MTD) may be due to improper implementation of the attack technique and not due to the proposed defense technique. The previous works either provide only 42% attack repeatability rate [22] or do not study this aspect [23].

The impact of TDC-based sensor's placement and noise. The sensor's placements (further details regarding TDCs are discussed in Section 4.2) and the impact of noise on the efficacy of the CPA attack must be studied, as these factors lead to an increase in the MTD. The attack feasibility reduces as the MTD increases. The increased MTD mandates the attacker to have access to an increased number of power traces. The prior works either report a very high MTD ($500K$) [22] to retrieve the 128-bit AES key [22] or retrieve only one AES key byte [23]. However, to show a practical attack, it is essential to retrieve the entire key for the cipher under consideration.

4.1.3 Challenges in power side-channel (PSC) defenses on cloud FPGAs

The challenges in the existing defenses are discussed next.

Realistic scenario. Crypto algorithms such as AES and RSA are a part of an SoC or wireless communication network. Nevertheless, there are no previous works that study the impact of additional circuits that reside along with the crypto cores.

Manual placement of FPGA primitives. The design implemented on the FPGA is inferred

using FPGA primitives, such as look-up tables (LUTs), flip-flops, and carry chains. The impact of manually placing these primitives that infer the circuit-under-protection is not studied in the existing works.

Challenges in active fence defense. The existing cloud FPGA defense [8] uses ROs to reduce the SNR of the power traces collected by the TDC-based sensors. However, ROs fasten the FPGA aging by creating hotspots [16,18] and also induce faults, making it vulnerable to fault injection attacks [97]. Additionally, cloud servers such as AWS blocks FPGA bitstream having combinatorial loops. Hence, implementing a defense using RO is not practically possible.

Defense thwarting only RO-based sensors. The current defenses can only detect ROs implemented in the attacker's logic [20]. However, a successful CPA attack on AES is achieved by using TDC-based sensors [8,22,24]. Hence, there is a need to secure the victim's logic from TDC-based sensors. Therefore, this work focuses on addressing the above challenges in the cloud FPGA domain.

4.1.4 Contributions and organization of this chapter

The contributions addressing each of the challenges in the PSC attack and the defenses that thwart it are listed below.

- The sensitivity of the TDC-based sensors is improved by analyzing the impact of dissimilar net delays in the sensor design.
- The impact of junction temperature on the MTD is studied. Maintaining this temperature within a limit aids in achieving a repeatable attack. This repeatability means launching the attack every time retrieves the correct 128-bit AES key.
- The impact of the placement of FPGA primitives inferring the crypto core is analyzed.
- The resilience to the CPA attack is evaluated in a realistic scenario. The defender's logic consists of AES, other crypto cores such as MD5 and SHA256, from the MIT-II common evaluation platform (CEP) [21] SoC platform.

- The realistic defense scenario considered in this work neither affects the reliability of the FPGA nor induces voltage faults.
- The defense scenario considered also thwarts CPA attack irrespective of the type of sensor used by the attacker.

This chapter is organized as follows. In Section 4.2, we explain the background details related to PSC attack, sensors used, and the methodology of the CPA attack. In Section 4.3, we explain the previous work related to the PSC attack and its defenses in the cloud FPGA domain. The experimental setup used in this work is explained in Section 4.4. Section 4.5 discusses our enhanced PSC attack and the corresponding attack results. Section 4.6 shares the different methodologies proposed by this work to secure cloud FPGAs against PSC attacks and their experimental results. Finally, in Section 4.7, the inferences and conclusion from our experiments are explained.

4.2 Background

This section discusses FPGA fundementals, PSC attack, different sensors inferred using FPGA primitives, and the CPA attack methodology.

4.2.1 Understanding FPGAs

As this work uses Xilinx Zynq ZC706 FPGAs, we discuss the FPGA fundamentals using Xilinx terminologies. The FPGA consists of a two-dimensional array of configurable logic blocks. The communication between these logic blocks is via switch matrices, as illustrated in Fig. 4.1. These switch matrices consist of programmable interconnects that route signals. Based on the FPGA device used, each configurable logic block consists of a fixed number of slices. The FPGA used in this work has two slices per logic block. Each slice consists of the FPGA primitives such as LUTs, multiplexers, carry logics (CARRY4), and flip-flops. In this FPGA, each slice is composed of eight LUTs, three multiplexers, one CARRY4, and eight flip-flops. The LUTs, multiplexers, and CARRY4 implement combinational logic, whereas the flip-flops implement sequential logic.

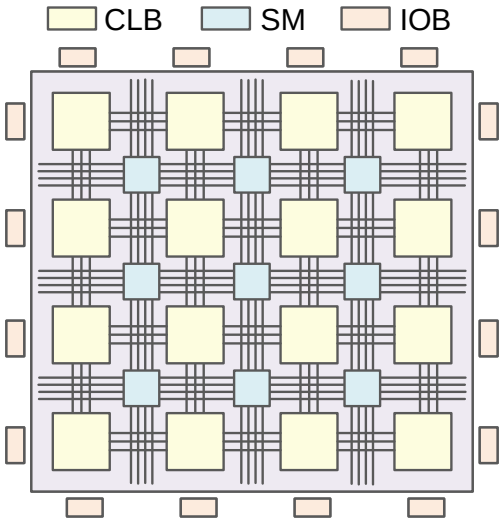

Figure 4.1: Field-programmable gate array (FPGA) consists of configurable logic blocks (CLBs) connected to each other via switch matrix (SM). The FPGA communicates with the outside world using input-output blocks (IOBs).

4.2.2 Power side-channel (PSC) attacks on cloud FPGA

In a conventional PSC attack, the attacker has physical access to the FPGA boards, where cryptographic algorithms, such as AES, are implemented. For a given key and a plaintext, the AES generates a corresponding ciphertext. Using an oscilloscope connected to the power rail of the FPGA, the attacker measures the power consumed (power traces) during the execution of each step in the AES algorithm. First, the power traces are logged for different plaintexts and a fixed key. Following this, he/she runs the PSC attack on the collected power traces to determine the secret key [105]. Unlike the conventional setup, where the attacker has physical access to the FPGA board, he/she does not have this access in the cloud. Thus, in cloud FPGAs, the bitstream loading and debugging are done remotely. However, even without this access, researchers have successfully demonstrated PSC attack on the cloud FPGAs [10, 13, 22, 23, 97]. The following paragraph explains how the PSC attack is feasible even without physical access to the FPGAs on a cloud server.

A successful PSC attack requires the sensing of voltage drop in the power distribution network (PDN) shared between the attacker and the victim logic. The PDN connects the supply voltage

to the FPGA primitives, such as LUTs and flip-flops. Irrespective of the change in load current to these primitives, the supplied voltage must be constant. However, due to the PDN structure's asymmetry, the supply voltage drops in the presence of load current variations [10, 22–24]. This variation in the load current is due to the data-dependent operations executed by the cryptographic algorithms. That is, the voltage drop on the PDN reflects the power consumed by these algorithms. The work [10] utilizes this relationship to successfully perform the simple power analysis attack on the 1024-bit RSA algorithm implemented on a cloud FPGA. Thus, even in the absence of physical access, the attacker uses a ring oscillator (RO) or a time-to-digital converter (TDC) circuits constructed using LUTs and flip-flops to measure the voltage drop rather than using an oscilloscope. The following section details the different sensors used in previous works launching PSC attack on cloud FPGA.

4.2.3 Sensors used in cloud FPGAs

The most commonly used sensors for performing the PSC attack on cloud FPGAs are ROs [10–13, 19] and TDCs [22–24]. These sensors sense the voltage drop during the execution of the cryptographic algorithms. This sensing is feasible, as the voltage supplied to the logic gates has an inverse proportionality impact on the propagation delay of these gates [106]. Therefore, a change in the propagation delay reflects the change in voltage drop. We now explain the different sensors using which the attackers measure the voltage drops in the FPGAs.

4.2.3.1 Ring oscillators (ROs)

The ROs are widely implemented in cloud FPGAs to aid PSC attacks [10–13, 19], thermal covert-channel attacks [16], voltage covert-channel attacks [17], and attacks based on inducing timing constraints faults. The conventional ROs are inferred using the LUTs. Hence, when the power consumption around these LUTs changes, the voltage drop varies. This variation gets reflected as a change in the propagation delay in these LUTs [106]. Therefore, the change in this propagation delay affects the time period and hence, the frequency of oscillations of the ROs. These ROs introduce several vulnerabilities listed below in the cloud FPGAs:

- In [16], ROs are used as heat generators to aid the formation of thermal covert channels. As multiple users share a single FPGA, this work shows that the heat generated by one user can be observed by another using the same FPGA instance at a different time.
- The ROs also generate severe voltage fluctuations [18]. These fluctuations crash the FPGA within a few microseconds. As a result, the system can be used only after power-cycling.
- They support voltage covert channel implementation [17].
- The attacker can induce timing constraint faults in the defender logic using ROs [15]. These faults are critical as they are temporary and hence, impossible to detect.

As ROs introduce various vulnerabilities in cloud FPGAs, the cloud service providers such as AWS blocks FPGA bitstreams containing combinatorial loops from getting deployed [107]. However, researchers designed ROs using latches and flip-flops to escape this filter in the AWS firewall [14], as illustrated in Fig. 4.2(b) and 4.2(c).

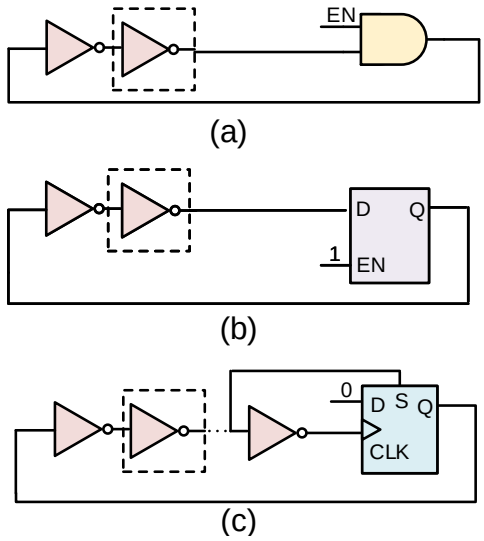

Figure 4.2: Ring oscillator (RO) variants used in the previous works [10–20]. The dotted box corresponds to an even number of inverters. (a) A simple RO based on a combinatorial loop. A single LUT infers each of these inverters in the FPGA. (b) RO based on the latch, which avoids the formation of a combinatorial loop. (c) Another RO variant, based on flipflops that avoids the formation of combinatorial loops. This variant can escape Amazon AWS firewall filters that block FPGA bitstreams with combinatorial loops.

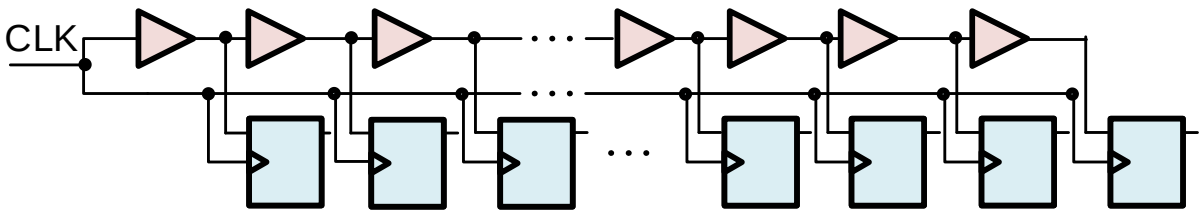

Figure 4.3: Time-to-digital converter.

4.2.3.2 Time-to-digital converters (TDC)

TDCs are also used in PSC attacks [22–24] to sense voltage drop variations. These sensors were originally developed to measure the propagation delay of logic gates and LUTs in ASICs and FPGAs, respectively [108–110]. As the change in propagation delay reflects the change in voltage drop [106], the TDC design is leveraged to sense voltage variations. They can measure the time interval between two signal pulses or the arrival time of a single pulse and then convert it to a digital number. They are used in [14, 19] to measure voltage fluctuation in FPGAs. In [17], the TDC finds its application in the receiver side to sense the voltage change. Also, in [15], it is used as a voltage drop sensor. In this work, TDC-based sensors measure the instantaneous power consumed by the 128-bit AES.

We supply a clock signal to the chain of CARRY4 primitives, as shown in Fig. 4.3. Each CARRY4 primitive consists of four buffers connected in series, and the voltage drop experienced by these buffers influences the propagation delay of these buffers. This delay, in turn, controls the number of buffers the clock signal travels through the chain. Finally, a latch controlled by the same clock signal gets connected to each buffer's output to log the number of buffers the clock has traveled. As TDCs can measure nanosecond delays and has higher accuracy [111], they are used in measuring the power consumed by the AES. Hence, in this work, we deploy TDCs to perform the same functionality. The following section explains the CPA attack methodology.

4.2.4 Correlational power analysis (CPA) attack methodology

A successful CPA attack on a 128-bit AES involves the following steps:

1. Using the TDC-based sensors, the attacker measures the power consumed by the AES for encrypting different plaintexts.

2. The first byte of the plaintext and the key controls the first byte of ciphertext, i.e., the encryption of each byte of the plaintext is independent of the other bytes. Hence, the attacker retrieves one key byte at a time.

3. For the first byte of the ciphertext, the attacker determines the output of the ninth round (first byte of intermediate ciphertext) using one out of 256 possible key guesses. He/She then calculates the Hamming distance between the first bytes of ciphertext and the intermediate ciphertext. Finally, the attacker repeats the Hamming distance calculation for the remaining 255 key guesses.

4. The attacker then correlates the power trace measured in step 1 with the 256 different values of Hamming distances calculated from the previous step.

5. The key guess corresponding to Hamming distance that has maximum correlation with the power trace is the correct key byte.

6. Finally, the attacker repeats steps 3 to 5 for all the remaining 15 key bytes to determine the 128-bit AES key.

4.3 Previous works

This section discusses the challenges in the previous works that proposed the PSC attack on 128-bit AES on a cloud FPGA platform and the defenses to thwart this attack.

4.3.1 Power side-channel (PSC) attack on AES implemented on cloud FPGAs

Several works demonstrated the PSC attack on the AES implemented on the victim's side [22, 23]. In [22], a TDC-based sensor is used to measure the power consumed by the AES for each encryption. This work can retrieve the 128-bit AES key using $500K$ power traces. However, as mentioned in this work, the attack success rate is only 42%, i.e., the attack is not repeatable. Apart

from demonstrating the PSC attack, another similar work [23] shows the impact of the distance between the attacker and victim on the number of traces required to retrieving the AES key. This work requires $1.8K$ traces to retrieve one key byte.

4.3.2 Defenses to thwart power side-channel (PSC) attacks

Active fences [8]. This work proposes to reduce the signal-to-noise ratio (SNR) by increasing the noise in the AES encryptions. This reduction in SNR is achieved by randomly enabling and disabling a group of ring oscillators. With this defense, the MTD to determine one key byte is as high as $300K$ traces. In this technique, the ROs surrounds the crypto algorithms to form a fence. This implementation secures these algorithms against PSA. As explained in the work, the RO is implemented using a single look-up table (LUT) that infers a two-input NAND gate. The output of the NAND gate is fed back to one of the inputs forming the RO. The other input connects to the enable signal, which enables or disables the RO.

CPAmap [24]. This work aims to understand the underlying mechanisms and dependencies of chip-internal side-channel attacks. It investigates the sensitivity of the TDC-based sensors on different locations on the FPGA exhaustively. It is achieved by running the correlational power analysis (CPA) attack on the traces collected by the sensors placed at different locations. The impact of Vivado implementation settings on the bitstream generations is also studied. However, there are few challenges in this work, as mentioned below.

- After executing the PSC attack at multiple locations on the FPGA, the cloud service provider identifies the locations on the FPGA that are not safe for the crypto algorithms. These are the locations where the sensor can determine the key with less MTD. The provider refrains from allocating these locations to any user leading to FPGA resource wastage.

- The results of this work prove that unless an exhaustive experiment is conducted on each FPGA, the cloud service provider cannot determine the sensitive locations on the FPGA. This process is time-consuming, and it might have to be repeated as the chip ages, as aging affects the sensitivity of the FPGAs.

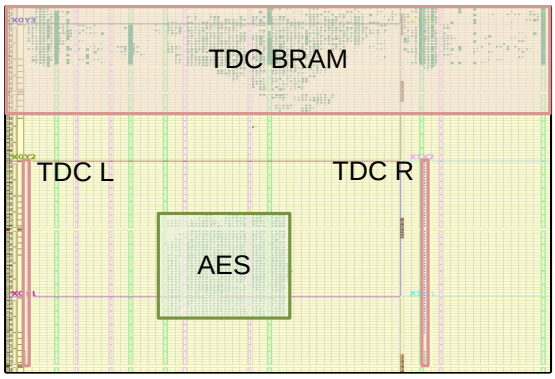

Figure 4.4: FPGA device view of the power side-channel attack (PSA) setup. The victim's logic (green) is the 128-bit iterative AES crypto algorithm. The attacker uses two TDC-based sensors to measure the power consumption and the block RAMs (BRAMs) to store these power traces. The attacker's logic is shown in red.

Mitigating voltage attacks [20]. The work senses or identifies the source of voltage attacks (logics that cause abnormal voltage drops) on the cloud FPGAs. The attacker implements malicious ROs triggers sudden voltage drops causing fault attacks [18]. To defend against this attack, the cloud service provider in [20] deploys sensors at multiple locations on the FPGA. If the voltage drop measured is less than the pre-defined value, the clock to that region is disabled, thereby thwarting the attacks.

4.4 Evaluation plan and setup

This section discusses the evaluation plan of this work. This work focuses on (i) improving the PSC attack and (ii) studying the impacts of primitive-level placement and extra logic placed along with the crypto design. Rather than proposing a new attack or defense technique, this work studies the impact of primitive-level placement in the sensor design (attack) and the crypto design under protection (defense). Also, we evaluate the CPA attack on multiple case studies where other logic designs such as processors and filters are placed along with the AES under protection. Our baseline experimental setup consists of two TDC-based sensors, one on the left and one on the right of the AES, as illustrated in Fig. 4.4.

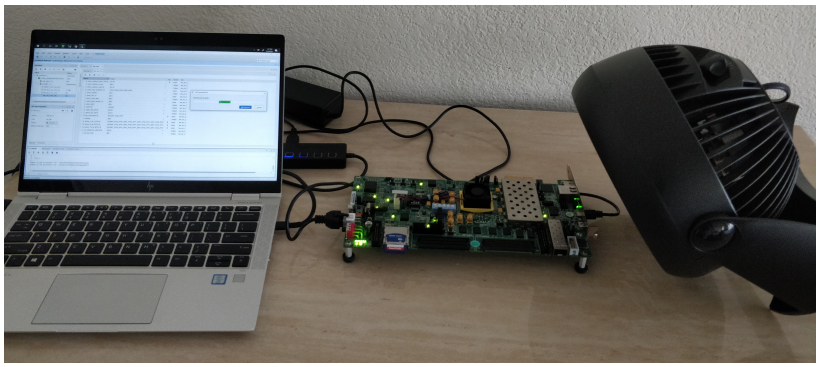

Figure 4.5: All experiments in this work are demonstrated on the Zynq ZC706 FPGA evaluation platform. The board is connected to the host machine, where the power traces are collected to perform correlation power analysis (CPA) attacks. Apart from the fan on the heat sink to reduce the FPGA junction temperature, we also placed a table fan to cool the FPGA board further.

4.4.1 Threat model

This work assumes that the cloud services support multi-tenanting, i.e., multiple users can share one FPGA instance, in which one of the users can be an attacker (a malicious user). As the cloud services correspond to a common resource pool shared by multiple users, there is a high probability that the allocation of the victim's logic and the attacker's (malicious user) logic happens in the same FPGA instance. Additionally, the attacker or defender does not have physical access to the FPGAs. Therefore, he/she cannot probe the power rails to measure the power consumed using oscilloscopes.

This work chooses the 128-bit iterative version of the AES crypto algorithm as the victim's logic, whose source code is available at [112]. It also includes a virtual input output (VIO) debug IP core [113] to transmit and receive data from and to the AES core via the JTAG signals of the FPGA. The attacker's logic uses the TDC-based sensors to measure the power consumed by the AES encryption. Additionally, the attacker's logic includes block RAMs and a VIO debug IP core to store and send the power traces, respectively, to the user. The source code of the CPA attack [114] is tailored to suit our attack setup. This work uses the Zynq ZC706 evaluation platform to execute all our experiments. This experimental setup along with the host machine and the cooling fan is shown in Fig. 4.5. The following section shares our attack contributions and the CPA attack results

on our enhanced attack setup. Similarly, the CPA attack results, power, and area consumption of the different placement-based experiments and experiments with extra logic follow this section.

4.5 Our enhanced attack on 128-bit AES

This section focuses on setting up a repeatable attack to evaluate the resilience offered by our defense securing cloud FPGAs against PSC attack, i.e., the crypto key must be retrieved every time launching the attack on the cloud FPGAs. If the attack is not repeatable, then the increased minimum traces to disclosure (MTD) may be due to improper implementation of the attack itself and not the proposed defense. However, the previous works [8, 24, 115] either do not have a 100% attack success rate or do not evaluate the success rate. Hence, in this work, we propose fine-tuning the sensor design to reduce the MTD required to determine the 128-bit AES key and make the attack repeatable. The following sections discuss (i) the manual placement of FPGA primitives inferring the sensors, (ii) determining the time instant to which the CPA attack has the highest correlation, and (iii) studying the impact of junction temperature.

Sensor placement. In this work, we explore the impact of manually placing each FPGA primitive inferring the TDC-based sensor. As explained in Section 4.2, the TDC consists of a chain of CARRY4 primitives. Each primitive consists of four outputs, where each output gets connected to a latch. These latches are, in turn, connected to the flip-flops. When Vivado places these primitives automatically, the net delay between the latches and their corresponding flip-flops are different for different pairs, as shown in Fig. 4.6 (b). This difference impacts the sensor output. By manually placing the flip-flops (colored in blue) illustrated in Fig. 4.6 (c), the net delay is approximately equal between all the latch-flipflop pairs. This manual placement of these latches and flip-flops is achieved by providing user-defined constraints, such as LOC and BEL, during bitstream generation [116]. With the help of this primitive-level placement, we could retrieve the 128-bit AES key using $10.7K$ and $10K$ traces using the left and right TDC-based sensors, respectively.

Determining the time instant at which the highest correlation occurs. As explained in Section 4.2, in the CPA attack, the key guess corresponding to Hamming distance that has a maximum

Figure 4.6: (a) FPGA device view of the TDC-based sensor. The zoomed version of the section of sensor enclosed in the red box are shown in Fig. 4.6 (b) and (c). (b) The Vivado tool automatically places the FPGA primitives corresponding to the sensor, leading to unequal net delay between the different latch-flipflop pairs. (c) The attacker judiciously places the FPGA primitives such that there is an approximately equal net delay between the different latch–flip-flop pairs.

correlation with the power trace is the correct key byte. Here, the Hamming distance is calculated between the tenth and ninth round ciphertexts. Hence, rather than considering the power consumed during all the ten rounds of AES encryption, we consider only the power consumed by the flip-flops during the tenth round of encryption. As a result, it helps achieve a low MTD compared to the existing works [8, 24, 115]. Additionally, it also reduces the number of samples collected for each encryption.

In this work, by fine-tuning the sensor design, we propose to demonstrate a repeatable attack on the 128-bit AES algorithm implemented on the Zynq ZC706 evaluation board. As shown in Fig. 4.7, the 128-bit key is retrieved using $3.8K$ traces. Furthermore, on all the ten trials, the attack successfully determines the 128-bit key. As explained in Section 4.5, the attack setup consists of

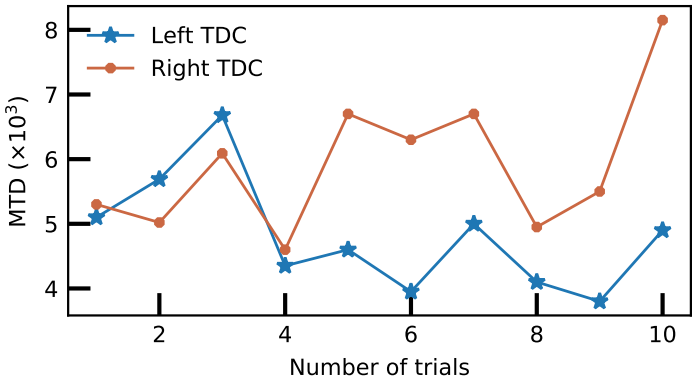

Figure 4.7: Correlation power analysis (CPA) attack results on the enhanced attack setup with two sensors, left and right TDCs. The reliability of the attack is tested by collecting the traces multiple times for the same build. In each trial, all 16 key bytes are retrieved successfully. Minimum number of traces for disclosure (MTD) and time to digital converter (TDC).

the iterative AES algorithm and two TDC-based sensors, one on each side of AES. This setup is illustrated in Fig. 4.4. The MTD reported by the previous works to determine the AES key is either around $1.8K$ traces [8] for one key byte or $500K$ for 16 key bytes. However, as illustrated in Fig. 4.7, the minimum and maximum MTD to determine the 128-bit key are $3.9K$ and $8.1K$ traces, respectively. The manual placement of each primitive in the sensor design aids in reaching this low MTD. This placement ensures approximately equal net delays between each latch–flip-flop pair, thereby reducing the errors in the sensor output, reflecting the power consumed by the defender's logic.

Understanding the impact of junction temperature. Powering on the evaluation board for a long time increases the junction temperature of the FPGA. Additionally, the rate at which the different parts of the FPGA cool depends on these heat sink paths [117]. Hence, the increased junction temperature induces temperature-dependent noise. This noise impacts the CPA attack resulting in increased MTD to determine the 128-bit key. As shown in Table 4.1, with increased junction temperature, the CPA attack requires a higher MTD to retrieve all key bytes. However, maintaining this temperature below 30°C (using a cooling fan), the CPA attack retrieves all key bytes within $6K$ traces.

Table 4.1: Correlation power analysis (CPA) attack results on our enhanced design. The design consists of AES (victim's logic) and the time-to-digital converter (TDC) (attacker's logic). The TDC is used for sensing the voltage variations in the victim's logic. The impact on temperature is studied. Minimum number of traces for disclosure (MTD).

Build number	**No cooling time**		**Half hour cooling time**	
	bytes recovered	**MTD**	**bytes recovered**	**MTD**
1	16	5400	16	3900
2	15	8471	16	4600
3	15	5977	16	5100

Table 4.2 compares the existing works on PSC attack with this work. As listed in this table, our attack has lesser MTD and 100% repeatability compared to the existing works.

4.6 Our defense analysis

4.6.1 The impact of primitive-level placement of AES on correlational power analysis (CPA) attack

As given in Section 4.2.4, for the correct key guess, the correlation between the power trace and the Hamming distance is maximum. This Hamming distance corresponds to the number of bit-flips between the AES's ninth and tenth (ciphertext) round outputs. These bit-flips are the change in flip-flop outputs that are responsible for the power consumption. Unlike the LUTs, the flip-flops' (edge-sensitive) output can change only during the rising edge of the clock. Thus, the flip-flops in the AES predominantly contribute to dynamic power consumption compared to the LUTs. As flip-flops mainly contribute to dynamic power consumption, we study the impact of the placement of these FPGA primitives manually on the CPA attack, rather than allowing the Vivado to perform automated placement of these primitives. Vivado places the FPGA primitives corresponding to the same logic as close as possible to reduce the wirelength delay. However, in this work, to ensure that the sensors cannot sense all the flip-flops of the AES, the defender spreads out the flip-flops across the clock region. The following are the different placement strategies the primitives in the AES algorithm have been placed on the Zynq 7000 series FPGA.

Table 4.2: Comparison of our contributions to PSC attack on 128-bit AES with the existing techniques [22, 23]. Minimum number of traces for disclosure (MTD). Power side-channel (PSC) attack.

Attack property	**PSC attack [22]**	**Inside job [23]**	**This work**
Repeatability	42%	NA	100%
MTD	$500K$	1800	3800
# of key bytes	16	1	16

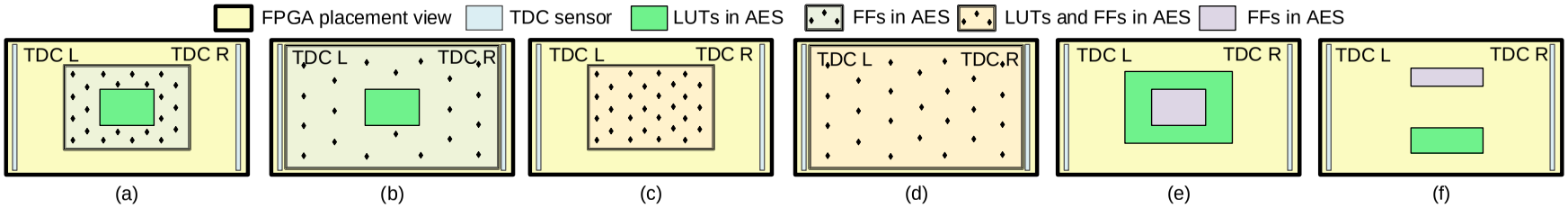

Figure 4.8: Placement strategies. (a) The flip-flops in the AES design have three empty slices between each other along the X and Y axes. (b) The flip-flops in the AES design have six empty slices between each other along the X and Y axes. (c) The flip-flops and LUTs in the AES design have three and two empty slices, respectively between each other along the X and Y axes. (d) The flip-flops and LUTs in the AES design have six and four empty slices, respectively between each other along the X and Y axes. (e) The LUTs surround the flip-flops in the AES design. (f) The flip-flop block (violet) and the LUT block (green) are placed one above the other.

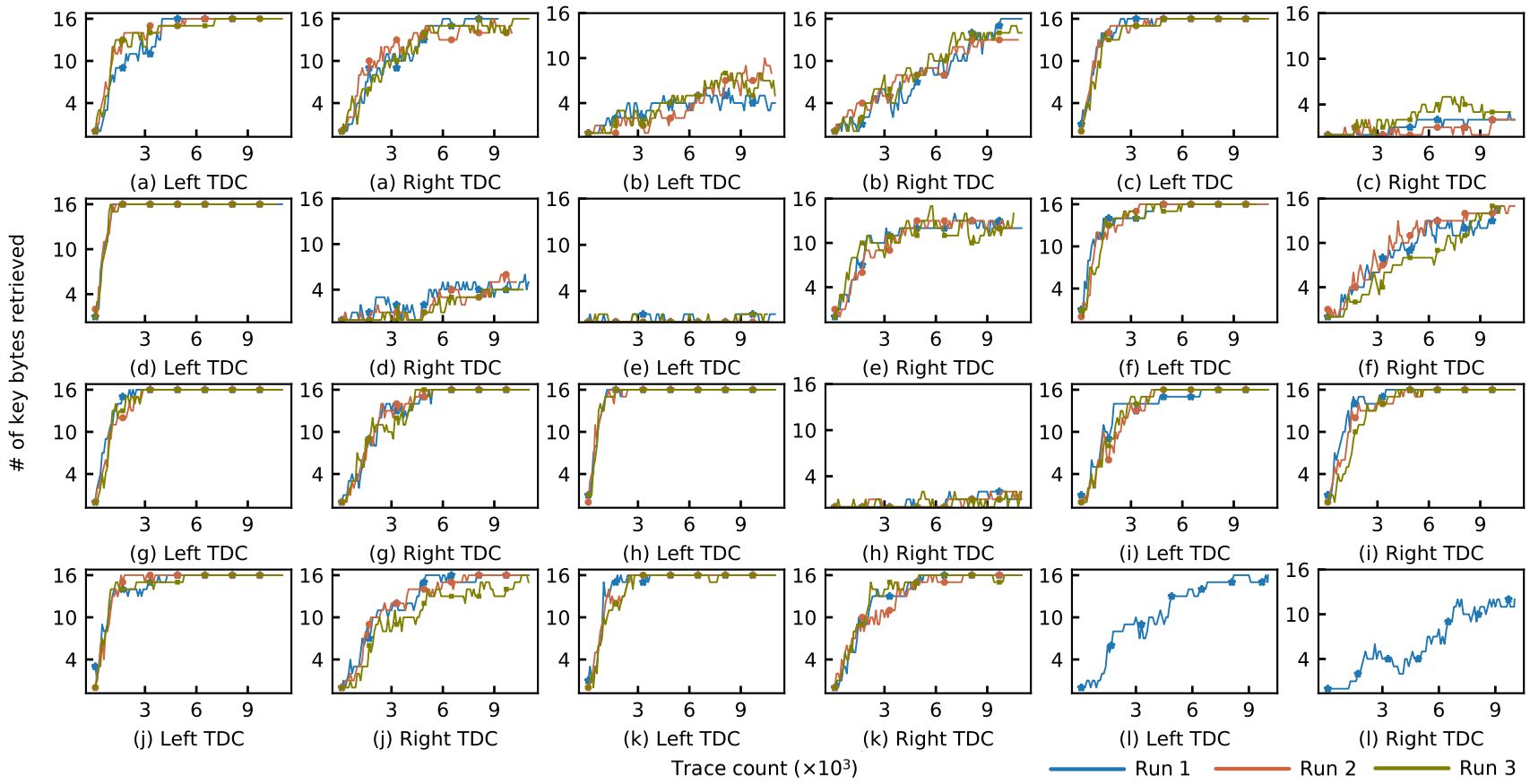

Figure 4.9: Correlation power analysis (CPA) attack on the power traces collected by the left and right TDC-based sensors. The flip-flops in the AES design have (a) two, (b) three, and (c) six empty slices between each other along the X and Y axes. The flip-flops and LUTs in the AES design have (d) three and two empty slices, (e) six and four empty slices, respectively, between each other along the X and Y axes. (f) The LUTs surround the flip-flops in the AES design. The placement of the flipflop block is above the LUT block. There are (g) 22, (h) 75, and (i) 113 empty slices in between the blocks. The width and depth of the flipflop block are 30 and 4, respectively. The width and depth of the LUT block are 30 and 13, respectively. There are (j) 75 and (k) 113 empty slices in between the blocks. The width and depth of the flipflop block are 90 and 2, respectively. The width and depth of the LUT block are 90 and 6, respectively.

1. **Only the flip-flops are spread out.** In this technique, only the flip-flops in the AES design are spread across the clock regions, as illustrated in Fig. 4.8(a) and 4.8(b). The flops correspond to the plaintext and the key registers in the crypto algorithm. Increasing the number of empty slices between each flop along the X- and Y-axis of the FPGA achieves different spread levels.

2. **Both the flip-flops and LUTs are spread out.** In this technique, both the LUTs and flip-flops in the AES design are spread across the clock regions, as illustrated in Fig. 4.8(c) and 4.8(d). Increasing the number of empty slices between each flop and LUT along the X- and Y-axis of the FPGA achieves different spread levels.

3. **Flip-flop block surrounded by the LUT block.** Here, the flip-flops constrained within a block are surrounded by the LUTs, as shown in Fig. 4.8(e).

4. **Flip-flop block and the LUT block are placed one above the other.** Like the previous placement, the flip-flops and the LUTs are constraints to an individual block of regions on the FPGA and placed one above the other. Here, different spread levels are generated with an increasing number of empty slices between the two blocks, as illustrated in Fig. 4.8(f).

We now discuss the CPA attack results for these different placement strategies explained above.

4.6.2 Correlational power analysis (CPA) attack on our defense case studies

4.6.2.1 Impact of primitive-level Placement of AES on the MTD

Spreading the flip-flops and LUTs. The CPA results on the build based on the different primitive placement strategies discussed in Section 4.6.1 are illustrated in Fig. 4.9. When there are two empty slices between any two flip-flops, most of the key bytes are retrieved using less than $5K$ traces, as illustrated in Fig. 4.9(a). Increasing the empty slices to three requires around $10K$ traces to determine 16 and 10 key bytes by the right and left sensors, respectively, as shown in Fig. 4.9(b). However, increasing the number of empty slices between the AES flip-flops has different effects on the left and right sensors.

Fig. 4.9(c) shows that the left sensor determines the 128-bit key using less than $3K$ traces, while the right sensor retrieves less than three key bytes using $11K$ traces. Similarly, as shown in Fig. 4.9(d), when the number of empty slices between the LUTs and flip-flops of the AES is 2 and 3, respectively, the left sensor provides more accurate power traces compared to the right sensor. This difference is because the left sensor requires less than $2K$ traces to retrieve the 128-bit key, whereas the right sensor requires more traces to determine the 128-bit key. When the number of empty slices between the LUTs and flip-flops of the AES is increased to 4 and 6, respectively, the right sensor is more sensitive than the left sensor, as shown in Fig. 4.9(e). The above results show the impact of inhomogeneity in the PDN structure on the sensitivity of the TDC sensors.

Grouping flip-flops and LUTs in separate partitions. As discussed in Section 4.6.1, the flip-flops and the LUTs of the AES design are grouped into separate partitions. In the AES design, 128 flip-flops and around $1.5K$ LUTs corresponding to the ciphertext. The partition block holding the flip-flops has a width of 30 slices and depth of four slices. Likewise, the block holding the LUTs has a width of 30 slices and a depth of 13 slices. There is a total of 75 empty slices between these partition blocks. As illustrated in Fig. 4.9(h), the left sensor's sensitivity is very high, as it can determine the 128-bit key with $1K$ traces. However, the right sensor could not retrieve more than two key bytes. Increasing the block width to 34 slices and reducing the block depth to three slices increases the sensitivity of both the sensors enabling the sensors to determine all the 16 key bytes, as shown in Fig. 4.9(l).

In the following section, we explain the impact of neighboring logic on the CPA attack.

4.6.3 Understanding the impact of neighboring logic on power side-channel (PSC) attack

In this section, we evaluate the resilience against PSA in a practical scenario. Generally, the crypto algorithms such as AES reside along with other designs on the FPGA rather than standalone.

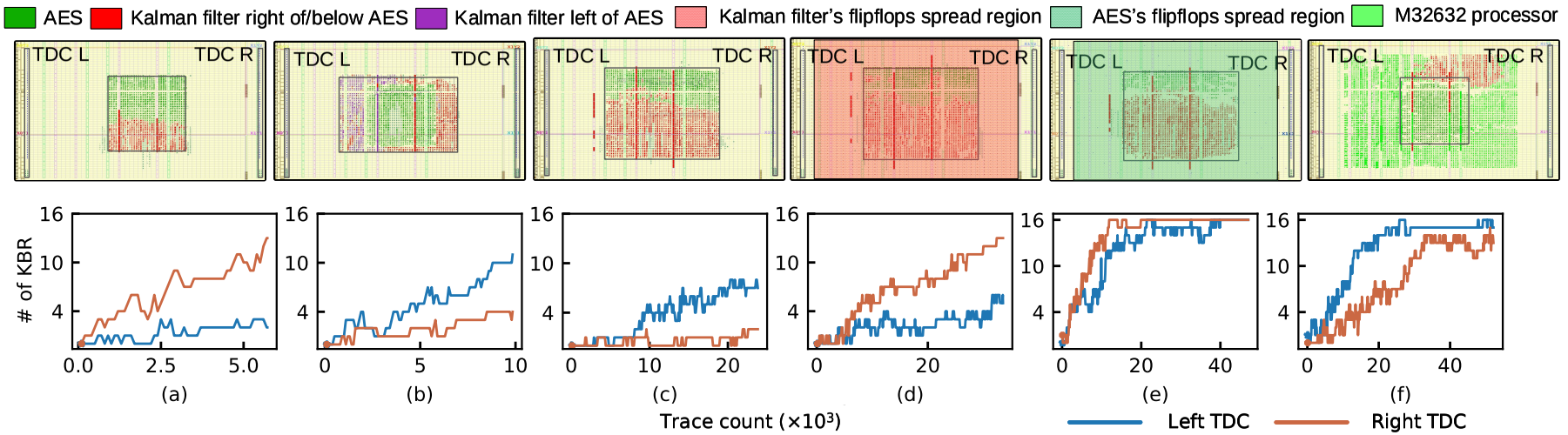

Figure 4.10: The FPGA device view and the correlation power analysis (CPA) attack results on the different experiments with a Kalman filter and AES. (a) One Kalman filter with 16-bit input placed below the AES. (b) One Kalman filter with 16-bit input is placed on each side of AES. (c) One Kalman filter with 48-bit input placed below the AES. (d) This setup is similar to Fig. 4.10(c). Here, the Kalman filter's flipflops are manually spread over multiple clock regions. (e) This setup is similar to Fig. 4.10(c). Here, the AES's flipflops are manually spread over multiple clock regions. (f) This setup has an M32632 processor and Kalman filter along with AES.

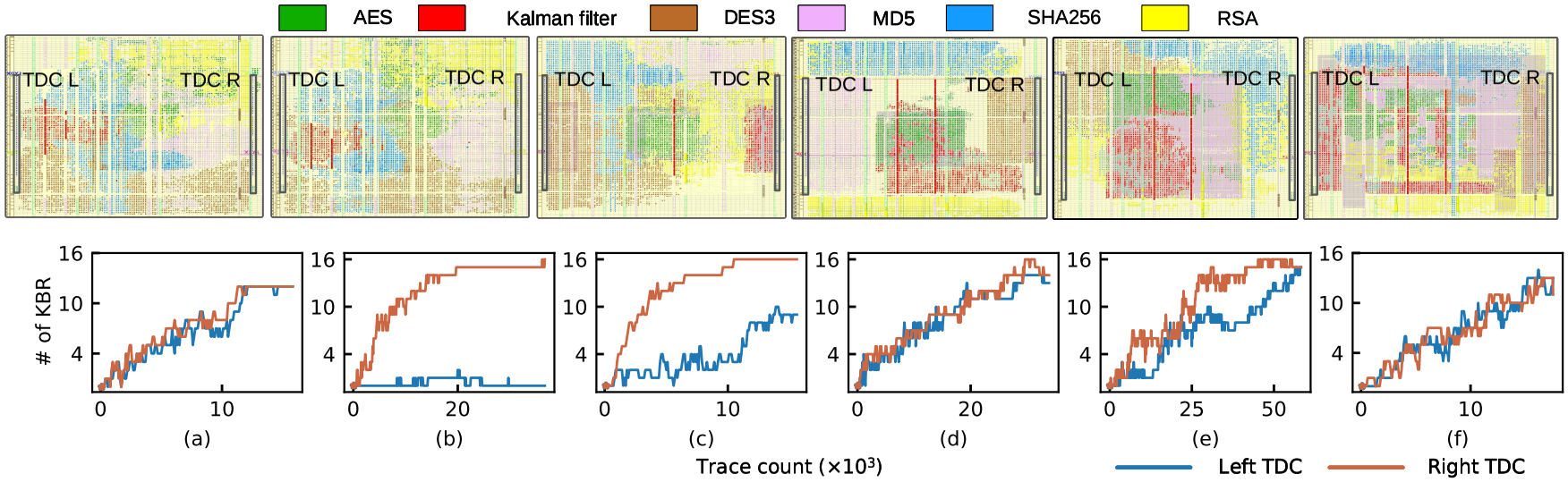

Figure 4.11: The FPGA device view and the correlation power analysis (CPA) results on the different experiments with MIT-II CEP [21] residing along with AES. (a) The Vivado does the automatic placement of the MIT-II CEP cores. (b) The flipflops of the MIT-II CEP cores are placed in the same partition block in which AES resides. (c) The Kalman filter is placed between the right of AES and the right sensor. Similarly, DES3 is placed between the left of AES and the left sensor. (d) The activity factor of all the subblocks inside the MIT-II CEP is aligned with the AES activity factor. (e) As the Kalman filter has high activity factor compared to the other logic design in the MIT-II CEP cores, a 48-bit input Kalman filter is included in the design. (f) The defender manually does the placement, where the AES and MIT-II CEP cores are merged.

Choice of logic residing along with AES. This crypto algorithm is used in different applications such as wireless security, processor security, file encryption, and secure sockets layer/transport layer security [118]. Therefore, in this work, we choose the following designs to reside along with AES: (i) the Kalman filter, (ii) the M32632 processor, and (iii) MIT-II CEP crypto cores. As filters are used predominately in wireless communications, they are chosen as one of the designs to study the resilience against PSA. An open-source processor design, M32632, is another logic chosen for this study. MIT-II CEP [21] is developed to provide an open-source evaluation platform to the users to evaluate their custom tools and techniques. As this platform has a processor integrated with crypto cores, we implement that section of the CEP with the crypto cores that include MD5 (Merkle–Damgård), SHA256 (Secure Hash Algorithm 256), DES3 (Data Encryption Standard), and RSA, along with the AES under protection.

Kalman filters. Our first set of experiments were with Kalman filters. As discussed later in this section, the Kalman filter is either placed below or next to the AES. The FPGA device view of these placements are shown in Fig. 4.10(a), Fig. 4.10(b), and Fig. 4.10(c). The impact of spreading the flip-flops is studied in these experiments as well. In one build, the flip-flops in the Kalman filter are spread. While in the other, the flip-flops in the AES design are spread over more than one clock region.

Evaluation platform. As the crypto algorithms such as AES are used predominantly in processor security, we will evaluate the resilience of the build with an open-source processor M32632 and the MIT-II CEP [21]. In the case of MIT-II CEP, in this work, we implement only a part of this platform that includes all the crypto cores. We analyze the activity factor of each crypto core in the platform. This work assumes the best-case scenario for the attacker — he/she can place the sensors next to the boundary of the defender's logic. Also, the impact of the activity factor over the sensors' output is maximum when they are closest to the sensors [100]. Hence, in this work, the cores having high activity factors are placed close to the boundary of the allocated region, as shown in the device views in Fig. 4.11.

We shall now discuss the impact of placing these extra logic designs along with AES on the

MTD. Fig. 4.10 shows the device view of different experiments with Kalman filter along with the CPA attack results for each of these experiments.

1. Fig. 4.10(a) illustrates the device view with one Kalman filter with an input size of 16 bits. However, the CPA attack can retrieve 13 key bytes in less than $6K$ traces. A single Kalman filter is not sufficient to lower the SNR of the AES encryption, thereby leaving it still vulnerable to PSA.

2. Fig. 4.10(b) shows the device view with one Kalman filter placed between the AES and each of the sensors. Each Kalman filter has a 16-bit input. This setup requires $10K$ power traces to determine ten key bytes, approximately twice the number of traces compared to the previous setup with one Kalman filter.

3. Fig. 4.10(c) shows considerable increase in MTD. This design has only one Kalman filter; however, the input size is 48 bits.

4. Fig. 4.10(d) illustrates the CPA attack results when the M32632 processor and a 16-bit Kalman filter reside along with the AES.

5. From Section 4.6.2.1 it is evident that spreading the flip-flops of the AES reduces the MTD considerably up to $1K$ traces. As the wirelength between the flip-flops and LUTs increases, the power consumed also increases. Thus, in this setup, the flip-flops of the Kalman filter are spread out. This setup increases the power consumed by the filter. Thus, we intend to increase the power consumed by the Kalman filter in the total power consumed by the defender. However, the attack results are similar to those corresponding to the setup that does not have the Kalman filter flip-flops spread out.

6. As the wirelength between the flip-flops and LUTs increases, the power consumed also increases. In some cases, as the AES flip-flops are spread, the sensor cannot sense the power consumption of a few of these flip-flops. The phenomenon will increase the MTD. Thus, the AES flip-flops are spread in the setup that has the Kalman filter with 48-bit input. In this

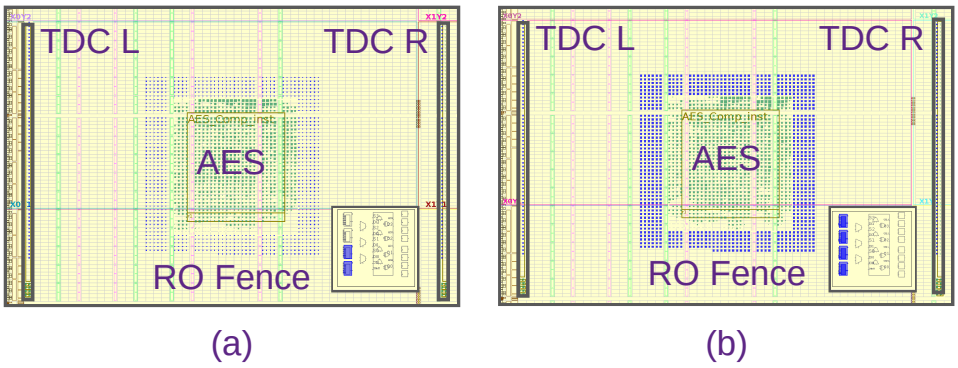

Figure 4.12: FPGA Device view of the active fence implementation [8]. Each slice in the RO fence consists of (a) two and (b) eight ROs. Each device view has an inset figure showing the LUT utilization of the slice.

setup, the SNR of the AES operation increases due to the increased power consumption from the additional wire length. Hence, as illustrated in Fig. 4.10(e), the CPA attack requires less than $11K$ traces to determine all 16 key bytes.

4.6.4 Active fence implementation [8]

To compare our technique with the existing works, in this section, we implement the active fences defense proposed in [8]. We then compare the CPA attack results of our work and this work. The AES implemented on Zynq 706 evaluation board requires 896 slices. Hence, as per the build details in [8], 896 ROs are implemented around the AES. This implementation results in 12.5% LUT utilization per slice, as each slice accommodates eight LUTs. Apart from the design shared in this work, we have tried other implementations of the ROs to check their impact on the CPA results. The other combinatorial RO design tried are as follows:

- As each slice consists of eight LUTs, two single-LUT ROs are implemented per slice, i.e., 1792 ROs implemented in the RO fence surrounding the crypto design.
- All eight LUTs implement the single-LUT RO. This build will degrade the FPGA as the fence has around $7K$ ROs, each oscillating at a frequency of around $1.4GHz$.
- One RO is inferred using eight LUTs. Each of the seven LUTs infers a buffer, and the

remaining one LUT infers an invertor. Finally, the eight LUTs are connected in a daisy chain fashion to implement a RO.

- Similar to the previous design, this design has each of the seven LUTs to infer an invertor, and the remaining one LUT infers a buffer.

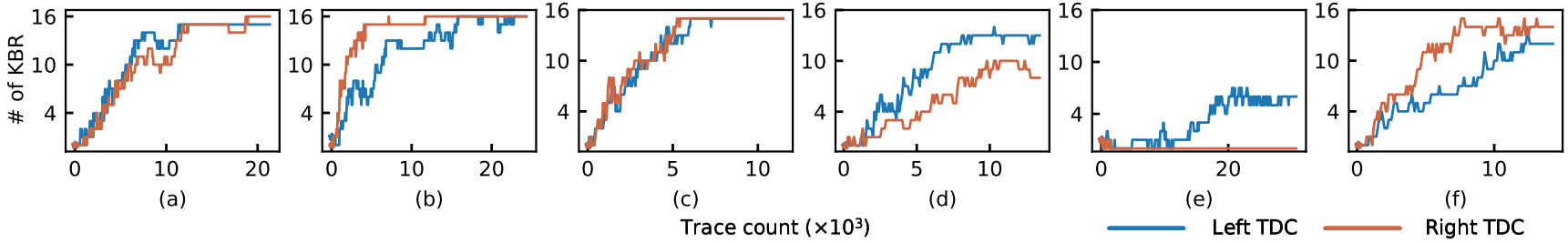

Figure 4.13: Correlation power analysis (CPA) attack results based on the different flavors of active fences work [8]. A total of 896 slices implements the ring oscillators (RO). (a) A fence separates the AES and the TDC sensor made up of 896 ROs. Each slice has one ring oscillator (RO) implemented in a single look-up table (LUT). (b) The RO fence made up of 896 ROs surrounds the AES. The remaining builds have similar placement as in Fig. 4.13. The difference is in the number of ROs in each slice and RO design. (c) The RO is implemented using seven LUTs inferred as invertors, and one LUT inferred as a buffer. (d) Each slice has two ROs, and each of these ROs are implemented using one LUT. (e) Each slice has eight ROs, and each of these ROs are implemented using one LUT. (f) This setup has the RO design shared in Fig. 4.2(c).

4.6.4.1 Impact of active fence on the MTD

The CPA attack results on the different implementations of the active fence [8] shared in Section 4.6.4 are explored in this section. Fig. 4.13(a) plots the CPA results for the setup with RO fence only on the right of AES. Fig. 4.12(a) illustrates the device view of this setup. The attack on the power traces from both the sensors share similar results though the RO fence is only to the right of the AES. Placing the same number of ROs around the AES as in Fig.4.12(b) reduces the MTD (requires less than $12K$ traces) to determine the 128-bit key. This low MTD is due to the reduced noise on each side of the AES induced by the RO. There is only one RO inferred in each slice in the builds so far. Increasing the number of ROs to two and eight per slice increase the MTD required to determine the 128-bit key, as illustrated in Fig.4.13(d) and Fig.4.13(e). This increase in MTD is due to the increased activity factor as the number of ROs per slice increases.

4.6.4.2 Need for ring oscillator (RO) design using flip-flops

Cloud servers such as AWS block this design from getting deployed on the F1 instance, as this technique implements combinatorial loops [107]. As ROs are used in PSC attacks [10], timing fault attacks [15], and also degrades the life of the FPGA [18], these servers thwart the loading of bitstream if they detect ROs in these bitstreams. Hence, we have also implemented the flip-flop-based ROs in the fences. The oscillating frequency of this type of RO is $284.01MHz$, which is less than the oscillating frequency of ROs inferred using a single LUT, equal to $1.4GHz$. Thus, the activity factor of the flipflop-based ROs is less compared to the single LUT ROs. This reduction, in turn, means less dynamic power and hence, less noise due to the ROs. Additionally, the ROs with very high oscillating frequency can have a destructive impact on the FPGA device. This impact is because the maximum clock frequency supported by the FPGA device is 800 to 900 MHz. Hence, the routing paths in the CLBs connecting the data ports of LUTs and flip-flops can support a frequency of up to only 400 to 450MHz. Therefore, a build is generated with ROs based on flip-flops, as shown in Fig. 4.2(c). The CPA results on the power traces collected from this build retrieve 15 out of 16 bytes using $10K$ traces, as illustrated in Fig.4.13(f).

Table 4.3 compares the defenses to thwart PSC attack on AES on cloud FPGAs with the defense proposed in this work. As shown in this table, compared to the existing defense works, this case study based work has the following advantages: (i) study the impact of the practical environment on the CPA results, (ii) study the impact of FPGA primitive-level placements on the CPA results, (iii) does not include ROs in their defense, (iv) this defense secures against TDC-based sensors, and (v) has a higher MTD and retrieves less key bytes compared to the existing defenses. However, unlike [24], we do not test the effectiveness of the defense at multiple locations. The onboard experimental results show that the CPA results are susceptible to the PDN design and routing apart from the placements. Hence, as part of future work, we intend to understand the PDN design in the FPGA and study the impact of routing on the CPA attack. Following these experiments, we will test the defense at multiple locations on the FPGA.

4.7 Inferences and conclusion

Impact of asymmetry in PDN structure. Fig. 4.9 and Fig. 4.11 shows the change in MTD with change in the placement of the FPGA primitives corresponding to the 128-bit AES design. Though the functionality of the implemented design is same, even a small change in the placement

Table 4.3: Comparison of our contributions with the existing defenses: (i) active fences [8], (ii) CPAmap [24] and (iii) votlage attack mitigation [14]. CPAmap [24] requires as high as 10M traces to determine one key byte. However, this is using the active fences [8] that implements combinatorial loop-based ROs. As these ROs cannot be implemented on cloud servers, we have mentioned "not applicable" (NA) for minimum traces for disclosure (MTD) for the CPAmap [24].

Defense property	**[8]**	**[24]**	**[14]**	**Our work**
Practical environment	✗	✗	✗	✓
FPGA primitive-level placement of AES	✗	✗	✗	✓
Absence of ROs	✗	✗	✓	✓
Protection against TDC-based sensors	✓	✓	✗	✓
Tested at multiple locations	✗	✓	✗	✗
MTD	$10K$	NA	NA	$24K$
# of key bytes retrieved	14	NA	NA	8

locations causes a change in MTD.

- **Spreading the flip-flops.** As shown in Fig. 4.9(a), 4.9(b), and 4.9(c) increasing the number of empty slices between the flip-flops increases the difficulty of retrieving the AES key by the right sensor. However, the left sensor can retrive the key using 6K traces when the number of empty slices are two (minimum spread) and six (maximum spread).

- **Spreading the flip-flops and LUTs.** Increasing the number of empty slices between the flip-flops and LUTs decreases and increases the difficulty of retrieving the AES key by the right and left sensors, respectively, as illustrated in Fig. 4.9(d) and 4.9(e).

- **Spreading the flip-flops and LUT blocks.** The Fig. 4.9(h) corresponds to the CPA attack results for the build that has the partition block holding the flip-flops has a width of 30 slices and a depth of four slices. Likewise, the block holding the LUTs has a width of 30 slices and a depth of 13 slices. There is a total of 75 empty slices between these partition blocks. Here, we could not retrieve more than two key bytes from the traces collected from the right sensor. However, increasing the width of the flip-flop and LUT block by four slices in the build makes design vulnerable CPA attack, as shown in Fig. 4.9(l).

- **Placing the flip-flops of the surrounding logic with AES.** The Fig. 4.11(b) corresponds to the CPA attack results for the build that has the flip-flops of MIT-II CEP crypto cores reside along with the FPGA. Here, the traces collected from the left sensor does not retreive even one key byte. However, using the traces collected by the right sensor we can retrieve all 16 key bytes using 40K traces.

Different clock regions. Fig. 4.4 shows that the left TDC and the AES share the same clock regions, X0Y0 and X0Y1; the right TDC is located in clock region X1Y0. Ideally, the left TDC must be more sensitive than the right one, determining the keys with lesser MTD. However, as shown in Fig. 4.11(b) and Fig. 4.11(c) the right TDC is more sensitive compared to the left one. Therefore, we cannot rely on the clock regions to understand the TDC sensitivities.

In this work, we address the different challenges in the PSC attack and its defenses. Firstly, we enhanced the attack by manually placing the FPGA primitives corresponding to the TDC-based sensor design. This manual placement helped achieve an MTD as low as 3.8K traces to retrieve the 128-bit AES key. The impact of the junction temperature on the MTD was studied, which helped set up a repeatable attack. As a result, our attack can determine the 128-bit key every time it is launched on the FPGA. We then studied the impact of spreading the FPGA primitives corresponding to the AES on the CPA attack results. Finally, we also evaluated builds with additional logic residing along with AES; compared their CPA results with builds supported with active fences. Our experimental results show that the AES with additional logic having sufficient activity factor can provide the same or increased MTD compared to the build with AES surrounded by the active fences. Additionally, our defense keeps the reliability of the FPGA intact as it does not include high-speed combinatorial loops.

5. SUMMARY AND CONCLUSION

In this dissertation, we address three challenges in the hardware security domain. In the first work, we proposed a defense technique to secure AMS circuits against overproduction. We secured these circuits by locking the digital section of AMS circuits using SFLL digital locking technique. This digital section controls the performance of the AMS circuit under protection. For an incorrect key, our approach achieves a minimum error of 27.45% and a maximum error of 50% in the circuit's response. In our second work, we show the vulnerabilities in the existing analog-only and AMS locking techniques. We successfully broke the existing defense techniques and showed that our attack time is independent of the key size. With advancements in digital locking techniques, we urge the analog community to a theoretical approach in developing defenses for analog circuits. The final work in this dissertation is a case study-based work. Here, we analyzed the impact of placements on both the PSC attack and the defense that thwarts this attack. The attack requires as low as 3.8K traces to retrieve the 128-bit AES key and is successful every time the attack is launched. The security provided by the defense shown in this work is equal to or better than the existing defenses.

Limitations and future works. The AMSlock [6, 32] is broken by the SMT- and SAT-based attack proposed in this dissertation. Hence, there is a need for stronger defense techniques that are resilient against these attacks. Unlike AMSlock that can secure only AMS circuits, defenses that secure analog-only design must be proposed. Additionally, as analog circuits are more sensitive to process variations than their digital counterparts, apart from simulation results, sharing results from taped-out chips is imperative. The SMT- and SAT-based attack [33, 34] can break the analog-only and AMS locks only when they have access to the functional specification of the design under protection. However, the availability of these specifications is not always feasible. Hence, attack techniques that do not require functional specifications of the protected design must be developed. Regarding the third piece of this dissertation work, the CPA attack results depend on the PDN design. Hence, the next step would be to reverse engineer the PDN design to understand better on

how the PDN influences the CPA attack results. Additionally, future work must involve the study of the impact of routing on the PSC attacks.

REFERENCES

[1] J. Wang, C. Shi, A. Sanabria-Borbon, E. Sanchez-Sinencio, and J. Hu, "Thwarting Analog IC Piracy via Combinational Locking," *IEEE International Test Conference*, pp. 1–10, 2017.

[2] V. V. Rao and I. Savidis, "Parameter Biasing Obfuscation for Analog IP Protection," *IEEE Latin American Test Symposium*, pp. 1–6, 2017.

[3] D. H. K. Hoe, J. Rajendran, and R. Karri, "Towards Secure Analog Designs: A Secure Sense Amplifier Using Memristors," *IEEE Computer Society Annual Symposium on VLSI*, pp. 516–521, 2014.

[4] G. Volanis, Y. Lu, S. G. R. Nimmalapudi, A. Antonopoulos, A. Marshall, and Y. Makris, "Analog Performance Locking through Neural Network-Based Biasing," *VLSI Test Symposium*, pp. 1–6, 2019.

[5] A. Ash-Saki and S. Ghosh, "How Multi-Threshold Designs Can Protect Analog IPs," *IEEE International Conference on Computer Design*, pp. 464–471, 2018.

[6] N. G. Jayasankaran, A. S. Borbon, E. Sanchez-Sinencio, J. Hu, and J. Rajendran, "Towards Provably-secure Analog and Mixed-signal Locking Against Overproduction," *IEEE/ACM International Conference on Computer-Aided Design*, pp. 7:1–7:8, 2018.

[7] K. Juretus, V. Venugopal Rao, and I. Savidis, "Securing Analog Mixed-Signal Integrated Circuits Through Shared Dependencies," *ACM Great Lakes Symposium on VLSI*, pp. 483–488, 2019.

[8] J. Krautter, D. R. Gnad, F. Schellenberg, A. Moradi, and M. B. Tahoori, "Active Fences against Voltage-based Side Channels in Multi-Tenant FPGAs," *IEEE/ACM International Conference on Computer-Aided Design*, pp. 1–8, 2019.

[9] M. Yasin, A. Sengupta, M. T. Nabeel, M. Ashraf, J. J. Rajendran, and O. Sinanoglu, "Provably-Secure Logic Locking: From Theory To Practice," *ACM SIGSAC Conference*

on Computer & Communications Security, pp. 1601–1618, 2017.

[10] M. Zhao and G. E. Suh, "FPGA-Based Remote Power Side-Channel Attacks," *IEEE Symposium on Security and Privacy*, pp. 229–244, 2018.

[11] I. Giechaskiel, K. B. Rasmussen, and J. Szefer, "Measuring Long Wire Leakage with Ring Oscillators in Cloud FPGAs," *IEEE International Conference on Field Programmable Logic and Applications*, pp. 45–50, 2019.

[12] I. Giechaskiel, K. Eguro, and K. B. Rasmussen, "Leakier Wires: Exploiting FPGA Long Wires for Covert- and Side-Channel Attacks," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 12, no. 3, 2019.

[13] C. Ramesh, S. B. Patil, S. N. Dhanuskodi, G. Provelengios, S. Pillement, D. Holcomb, and R. Tessier, "FPGA Side Channel Attacks without Physical Access," *Annual International Symposium on Field-Programmable Custom Computing Machines*, pp. 45–52, 2018.

[14] J. Krautter, D. R. E. Gnad, and M. B. Tahoori, "Mitigating Electrical-Level Attacks towards Secure Multi-Tenant FPGAs in the Cloud," *ACM Transactions on Reconfigurable Technology and System*, vol. 12, no. 3, 2019.

[15] D. Mahmoud and M. Stojilović, "Timing Violation Induced Faults in Multi-Tenant FPGAs," *IEEE/ACM Design, Automation & Test in Europe*, pp. 1745–1750, 2019.

[16] S. Tian and J. Szefer, "Temporal Thermal Covert Channels in Cloud FPGAs," *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, p. 298–303, 2019.

[17] D. R. E. Gnad, C. D. K. Nguyen, S. H. Gillani, and M. B. Tahoori, "Voltage-based Covert Channels in Multi-Tenant FPGAs." Cryptology ePrint Archive, Report 2019/1394, 2019. `https://eprint.iacr.org/2019/1394`.

[18] D. R. E. Gnad, F. Oboril, and M. B. Tahoori, "Voltage Drop-based Fault Attacks on FPGAs Using Valid Bitstreams," *IEEE International Conference on Field Programmable Logic and Applications*, pp. 1–7, 2017.

[19] D. R. E. Gnad, S. Rapp, J. Krautter, and M. B. Tahoori, "Checking for Electrical Level Security Threats in Bitstreams for Multi-tenant FPGAs," *IEEE International Conference on Field-Programmable Technology*, pp. 286–289, 2018.

[20] G. Provelengios, D. Holcomb, and R. Tessier, "Mitigating Voltage Attacks in Multi-Tenant FPGAs," *ACM Transactions on Reconfigurable Technology and Systems*, 2021.

[21] Massachusetts Institute of Technology, "Common Evaluation Platform." `https://github.com/mit-ll/CEP`, 2021. Last accessed on 09/05/2021.

[22] O. Glamočanin, L. Coulon, F. Regazzoni, and M. Stojilović, "Are Cloud FPGAs Really Vulnerable to Power Analysis Attacks?," *Design, Automation & Test in Europe*, pp. 1007–1010, 2020.

[23] F. Schellenberg, D. R. Gnad, A. Moradi, and M. B. Tahoori, "An Inside Job: Remote Power Analysis Attacks on FPGAs," *Design, Automation & Test in Europe*, 2021.

[24] J. Krautter, D. R. E. Gnad, and M. B. Tahoori, "CPAmap: On the Complexity of Secure FPGA Virtualization, Multi-Tenancy, and Physical Design," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2020, no. 3, pp. 121–146, 2020.

[25] M. Rostami, F. Koushanfar, and R. Karri, "A Primer on Hardware Security: Models, Methods, and Metrics," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1283–1295, 2014.

[26] T. S. Perry, "Why Hardware Engineers Have to Think Like Cybercriminals, and Why Engineers Are Easy to Fool," 2017. Last accessed on 01/13/2020.

[27] J. A. Roy, F. Koushanfar, and I. L. Markov, "Ending Piracy of Integrated Circuits," *IEEE Computer*, vol. 43, no. 10, pp. 30–38, 2010.

[28] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security Analysis of Logic Obfuscation," *IEEE/ACM Design Automation Conference*, pp. 83–89, 2012.

[29] Y. Xie and A. Srivastava, "Mitigating SAT Attack on Logic Locking," *Cryptographic Hardware and Embedded Systems*, pp. 127–146, 2016.

[30] X. Xu, B. Shakya, M. M. Tehranipoor, and D. Forte, "Novel Bypass Attack and BDD-based Tradeoff Analysis Against All Known Logic Locking Attacks," *Cryptographic Hardware and Embedded Systems*, pp. 189–210, 2017.

[31] B. Yang, K. Wu, and R. Karri, "Secure Scan: A Design-For-Test Architecture for Crypto Chips," *IEEE/ACM Design Automation Conference*, pp. 135–140, 2005.

[32] N. G. Jayasankaran, A. S. Borbon, E. Sanchez-Sinencio, J. Hu, and J. Rajendran, "Towards Provably-Secure Analog and Mixed-Signal Locking Against Overproduction," *IEEE Transactions on Emerging Topics in Computing*, pp. 7:1–7:8, 2020.

[33] N. G. Jayasankaran, A. S. Borbon, A. Abuellil, E. Sanchez-Sinencio, J. Hu, and J. Rajendran, "Breaking Analog Locking Techniques via Satisfiability Modulo Theories," *IEEE International Test Conference*, 2019.

[34] N. G. Jayasankaran, A. Sanabria-Borbón, A. Abuellil, E. Sánchez-Sinencio, J. Hu, and J. Rajendran, "Breaking analog locking techniques," *IEEE Transactions on Very Large Scale Integration Systems*, pp. 1–14, 2020.

[35] P. Tuyls, G. Schrijen, B. Škorić, J. van Geloven, N. Verhaegh, and R. Wolters, "Read-Proof Hardware from Protective Coatings," *Cryptographic Hardware and Embedded Systems*, pp. 369–383, 2006.

[36] M. Integrated, "DeepCover Security Manager for Low-Voltage Operation with 1KB Secure Memory and Programmable Tamper Hierarchy." `https://www.maximintegrated.com/en/products/embedded-security/security-managers/DS3660.html`, 2010. Last accessed on 01/13/2020.

[37] J. Leonhard, M. Yasin, S. Turk, M. T. Nabeel, M.-M. Louërat, R. Chotin-Avot, H. Aboushad, O. Sinanoglu, and H.-G. Stratigopoulos, "MixLock: Securing Mixed-Signal Circuits via Logic Locking," *IEEE/ACM Design Automation and Test in Europe*, 2019.

[38] A. Baumgarten, A. Tyagi, and J. Zambreno, "Preventing IC Piracy Using Reconfigurable Logic Barriers," *IEEE Design & Test of Computers*, vol. 27, no. 1, pp. 66–75, 2010.

[39] R. A. Rutenbar, "Design Automation for Analog: The Next Generation of Tool Challenges," *IEEE/ACM International Conference on Computer Aided Design*, pp. 458–460, 2006.

[40] "Top 5 Most Counterfeited Parts Represent a $169 Billion Potential Challenge for Global Semiconductor Market." `https://technology.ihs.com/405654/top-`, 2012. Last accessed on 01/12/2020.

[41] Y. Bi, J. Yuan, and Y. Jin, "Beyond the Interconnections: Split Manufacturing in RF Designs," *MDPI Electronics*, vol. 4, no. 3, pp. 541–564, 2015.

[42] J. Leonhard, M.-M. Louërat, H. Aboushady, O. Sinanoglu, and H.-G. Stratigopoulos, "Mixed-Signal Hardware Security Using MixLock: Demonstration in an Audio Application," *IEEE International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design*, 2019.

[43] F. Yang, M. Tang, and O. Sinanoglu, "Stripped Functionality Logic Locking With Hamming Distance-Based Restore Unit (SFLL-hd) -– Unlocked," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 10, 2019.

[44] D. Sirone and P. Subramanyan, "Functional Analysis Attacks on Logic Locking," *IEEE/ACM Design Automation and Test in Europe*, pp. 936–939, 2019.

[45] M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran, "Removal Attacks on Logic Locking and Camouflaging Techniques," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2017.

[46] M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran, "Security Analysis of Anti-SAT," *IEEE Asia and South Pacific Design Automation Conference*, pp. 342–347, 2017.

[47] A. Hastings, *The Art of Analog Layout*. Pearson, 2005.

[48] C. Toumazou, G. Moschytz, and B. Gilbert, *Trade-offs in Analog Circuit Design*. Kluwer Academic Publishers, 2002.

[49] T. McConaghy, K. Breen, J. Dyck, and A. Gupta, *Variation-Aware Design of Custom Integrated Circuits: A Hands-on Field Guide*. Springer, 2013.

[50] J. Wang, C. Shi, E. Sanchez-Sinencio, and J. Hu, "Built-In Self Optimization for Variation Resilience of Analog Filters," *IEEE Computer Society Annual Symposium on VLSI*, pp. 656–661, 2015.

[51] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the Security of Logic Encryption Algorithms," *IEEE International Symposium on Hardware Oriented Security and Trust*, pp. 137–143, 2015.

[52] P. Mroszczyk, J. Goodacre, and V. F. Pavlidis, "Energy Efficient Flash ADC With PVT Variability Compensation Through Advanced Body Biasing," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 11, pp. 1775–1779, 2019.

[53] Y. Zhao, Z. Shang, and Y. Lian, "A 13.34W Event-driven Patient-specific ANN Cardiac Arrhythmia Classifier for Wearable ECG Sensors," *IEEE Transactions on Biomedical Circuits and Systems*, pp. 1–1, 2019.

[54] R. Lowman, "A Holistic Approach to IoT Chip Design." `https://bit.ly/36q37Yi`, 2015. Last accessed on 01/24/2020.

[55] M. V. V. Rolf Schaumann, Haiqiao Xiao, *Design of Analog Filters*. Oxford University Press, 2009.

[56] G. Volanis, D. Maliuk, Y. Lu, K. S. Subramani, A. Antonopoulos, and Y. Makris, "On-Die Learning-based Self-Calibration of Analog/RF ICs," *IEEE VLSI Test Symposium*, pp. 1–6, 2016.

[57] B. Razavi, *RF Microelectronics*. Pearson Education, 2011.

[58] T. Instruments, "Understanding Low Drop Out (LDO) Regulators." `https://bit.ly/37bvvyE`, 2006. Last accessed on 01/24/2020.

[59] A. Sengupta, M. Nabeel, M. Yasin, and O. Sinanoglu, "ATPG-based Cost-Effective, Secure Logic Locking," *VLSI Test Symposium*, pp. 1–6, 2018.

[60] I. Guerra-Gómez, E. Tlelo-Cuautle, and L. G. De La Fraga, "Richardson Extrapolation-based Sensitivity Analysis in the Multi-objective Optimization of Analog Circuits," *Applied Mathematics and Computation*, vol. 222, pp. 167–176, 2013.

[61] J. Torres, M. El-Nozahi, A. Amer, S. Gopalraju, R. Abdullah, K. Entesari, and E. Sanchez-Sinencio, "Low Drop-Out Voltage Regulators: Capacitor-less Architecture Comparison," *IEEE Circuits and Systems Magazine*, vol. 14, no. 2, pp. 6–26, 2014.

[62] Y. Feng, G. Takemura, S. Kawaguchi, N. Itoh, and P. R. Kinget, "Digitally Assisted IIP2 Calibration for CMOS Direct-Conversion Receivers," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 10, pp. 2253–2267, 2011.

[63] B. Xiang, Y. Fan, J. Ayers, J. Shen, and D. Zhang, "A 0.5V-to-0.9V 0.2GHz-to-5GHz Ultra-Low-Power Digitally-Assisted Analog Ring PLL with Less Than 200ns Lock Time in 22nm FinFET CMOS Technology," *IEEE Custom Integrated Circuits Conference*, pp. 1–4, 2020.

[64] N. Inc, "NanGate FreePDK45 Open Cell Library." `http://www.nangate.com/?page_id=2325`, 2011. Last accessed on 01/13/2020.

[65] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "AppSAT: Approximately Deobfuscating Integrated Circuits," *IEEE International Symposium on Hardware Oriented Security and Trust*, pp. 95–100, 2017.

[66] E. Maricau and G. Gielen, "Transistor Aging-induced Degradation of Analog Circuits: Impact Analysis and Design Guidelines," *IEEE Proceedings of ESSCIRC*, pp. 243–246, 2011.

[67] D. Sengupta and S. S. Sapatnekar, "Estimating Circuit Aging Due to BTI and HCI Using Ring-Oscillator-Based Sensors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 10, pp. 1688–1701, 2017.

[68] D. S. Board, "Defense Science Board (DSB) study on High Performance Microchip Supply," 2005. [March 16, 2015].

[69] IHS Technology Press Release, "Top 5 Most Counterfeited Parts Represent a $169 Billion Potential Challenge for Global Semiconductor Market," 2012. Last accessed on 01/12/2020.

[70] K. Azar, H. Kamali, H. Homayoun, and A. Sasan, "SMT Attack: Next Generation Attack on Obfuscated Circuits with Capabilities and Performance Beyond the SAT Attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2019, no. 1, pp. 97–122, 2018.

[71] Y. Xie and A. Srivastava, "Delay Locking: Security Enhancement of Logic Locking Against IC Counterfeiting and Overproduction," *IEEE/ACM Design Automation Conference*, pp. 1–6, 2017.

[72] F. Yang, M. Tang, and O. Sinanoglu, "Stripped Functionality Logic Locking With Hamming Distance-Based Restore Unit (SFLL-hd) – Unlocked," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 10, pp. 2778–2786, 2019.

[73] D. Sirone and P. Subramanyan, "Functional Analysis Attacks on Logic Locking," *IEEE/ACM Design Automation and Test in Europe*, pp. 936–939, 2019.

[74] R. Torrance and D. James, "The State-of-the-Art in Semiconductor Reverse Engineering," in *IEEE/ACM Design Automation Conference*, pp. 333–338, 2011.

[75] C. S. Chang, C. P. Chao, J. G. J. Chern, and J. Y. C. Sun, "Advanced CMOS Technology Portfolio for RF IC Applications," *IEEE Transactions on Electron Devices*, vol. 52, no. 7, pp. 1324–1334, 2005.

[76] Texas Instruments, "Universal Active Filter." `https://www.ti.com/lit/ds/symlink/uaf42.pdf`, 2010. Last accessed on 04/25/2020.

[77] Chipworks, "Reverse Engineering Software." `http://www.chipworks.com/en/technical-competitive-analysis/resources/reerse-engineering-software`, 2016.

[78] T. Iizuka, *CMOS Technology Scaling and Its Implications*. 2015.

[79] R. Sotner, J. Jerabek, N. Herencsar, K. Vrba, and T. Dostal, "Features of Multi-Loop Structures with OTAs and Adjustable Current Amplifier for Second-Order Multi-phase/Quadrature Oscillators," *International Journal of Electronics and Communications*, vol. 69, no. 5, pp. 814 – 822, 2015.

[80] Z. Zahir and G. Banerjee, "A Multi-tap Inductor Based 2.0-4.1 GHz Wideband LC-Oscillator," *IEEE Asia Pacific Conference on Circuits and Systems*, pp. 330–333, 2016.

[81] R. Senani, D. R. Bhaskar, V. K. Singh, and R. K. Sharma, *Sinusoidal Oscillators and Waveform Generators using Modern Electronic Circuit Building Blocks*. Springer International Publishing, 2015.

[82] Texas Instruments, "AWR1243 Single-Chip 77-GHz and 79-GHz FMCW Transceiver." `http://www.ti.com/lit/ds/symlink/awr1243.pdf`, 2012. Last accessed on 01/06/2020.

[83] J. P. Lang, "iSAT3." `https://projects.informatik.uni-freiburg.de/projects/isat3/wiki/ISAT3_004`, 2017. Last accessed on 01/06/2020.

[84] M. Yasin and O. Sinanoglu, "Transforming Between Logic Locking and IC Camouflaging," *IEEE International Design Test Symposium*, pp. 1–4, 2015.

[85] S. Lee, C. Shi, J. Wang, A. Sanabria, H. Osman, J. Hu, and E. Sánchez-Sinencio, "A Built-In Self-Test and In Situ Analog Circuit Optimization Platform," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. PP, pp. 1–14, March 2018.

[86] Ilan Schnell, "pycosat 0.6.3." `https://pypi.org/project/pycosat/`, 2013. Last accessed on 01/06/2020.

[87] R. Torrance and D. James, *The State-of-the-Art in IC Reverse Engineering*, pp. 363–381. Berlin, Heidelberg: Springer, 2009.

[88] M. Elshamy, A. Sayed, M.-M. Louërat, A. Rhouni, H. Aboushady, and H.-G. Stratigopoulos, "Securing Programmable Analog ICs Against Piracy," *Design, Automation and Test in Europe Conference*, 2020.

[89] G. Volanis, A. Antonopoulos, S. G. R. Nimmalapudi, A. Marshall, Y. Makris, and Y. Lu, "Range Controlled Floating-Gate Transistors: A Unified Solution for Unlocking and Calibrating Analog ICs," *Design, Automation and Test in Europe Conference*, 2020.

[90] Amazon, "Amazon EC2 F1 Instance." `https://aws.amazon.com/ec2/instance-types/f1/`, 2016. Last accessed on 09/26/2020.

[91] Microsoft, "Microsoft Azure." `https://azure.microsoft.com/en-gb/`, 2021. Last accessed on 09/05/2021.

[92] IBM, "CloudFPGA." `https://www.zurich.ibm.com/cci/cloudFPGA/`, 2020. Last accessed on 09/26/2020.

[93] F. Chen, Y. Shan, Y. Zhang, Y. Wang, H. Franke, X. Chang, and K. Wang, "Enabling FPGAs in the Cloud," *ACM Conference on Computing Frontiers*, 2014.

[94] U. of Texas at Austin, "Texas Advanced Computing Center." `https://www.tacc.utexas.edu/`, 2020. Last accessed on 09/26/2020.

[95] Avnet, "Where is FPGA in Cloud Computing Today?." `https://www.avnet.com/wps/portal/apac/resources/article/where-is-fpga-in-cloud-computing-today/`, 2019. Last accessed on 09/05/2021.

[96] Z. István, G. Alonso, and A. Singla, "Providing Multi-tenant Services with FPGAs: Case Study on a Key-Value Store," *IEEE International Conference on Field Programmable Logic and Applications*, pp. 119–1195, 2018.

[97] G. Provelengios, D. Holcomb, and R. Tessier, "Power Distribution Attacks in Multitenant FPGAs," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 28, no. 12, pp. 2685–2698, 2020.

[98] F. Schellenberg, D. R. Gnad, A. Moradi, and M. B. Tahoori, "Remote Inter-Chip Power Analysis Side-Channel Attacks at Board-Level," *IEEE/ACM International Conference on Computer-Aided Design*, pp. 1–7, 2018.

[99] J. Krautter, D. R. E. Gnad, and M. B. Tahoori, "FPGAhammer: Remote Voltage Fault Attacks on Shared FPGAs, suitable for DFA on AES," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2018, no. 3, pp. 44–68, 2018.

[100] G. Provelengios, D. Holcomb, and R. Tessier, "Characterizing Power Distribution Attacks in Multi-User FPGA Environments," *IEEE International Conference on Field Programmable Logic and Applications*, pp. 194–201, 2019.

[101] T. Sugawara, K. Sakiyama, S. Nashimoto, D. Suzuki, , and T. Nagatsuka, "Oscillator without a Combinatorial Loop and Its Threat to FPGA in Data Centre," *Electronics Letter*, vol. 55, no. 11, pp. 640–642, 2020.

[102] I. Giechaskiel, K. Rasmussen, and J. Szefer, "Reading Between the Dies: Cross-SLR Covert Channels on Multi-Tenant Cloud FPGAs," *IEEE International Conference on Computer Design*, pp. 1–10, 2019.

[103] I. Giechaskiel, K. Rasmussen, and J. Szefer, "CAPSULe: Cross-FPGA Covert-Channel Attacks through Power Supply Unit Leakage," *IEEE Symposium on Security and Privacy*, pp. 909–922, 2020.

[104] I. Giechaskiel, K. Rasmussen, and K. Eguro, "Leaky Wires: Information Leakage and Covert Communication Between FPGA Long Wires," *ACM Asia Conference on Computer and Communications Security*, pp. 18–27, 2018.

[105] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," *Advances in Cryptology*, pp. 388–397, 1999.

[106] S. Pant, "Design and Analysis of Power Distribution Networks in VLSI Circuits," *Ph.D. dissertation, The University of Michigan*, 2008.

[107] Amazon, "Combinatorial Loops in F1 FPGA." `https://forums.aws.amazon.com/thread.jspa?threadID=264137`, 2017. Last accessed on 09/27/2020.

[108] K. Cui and X. Li, "A High-Linearity Vernier Time-to-Digital Converter on FPGAs with Improved Resolution Using Bidirectional-Operating Vernier Delay Lines," *IEEE Transactions on Instrumentation and Measurement*, pp. 1–1, 2019.

[109] H. Xia, G. Cao, and N. Dong, "A 6.6 ps RMS Resolution Time-to-Digital Converter Using Interleaved Sampling Method in a 29 nm FPGA," *Review of Scientific Instruments*, vol. 90, no. 4, p. 044706, 2019.

[110] J. Wang, S. Liu, L. Zhao, X. Hu, and Q. An, "The 10-ps Multitime Measurements Averaging TDC Implemented in an FPGA," *IEEE Transactions on Nuclear Science*, vol. 58, no. 4, pp. 2011–2018, 2011.

[111] S. Moini, X. Li, P. Stanwicks, G. Provelengios, W. Burleson, R. Tessier, and D. Holcomb, "Understanding and Comparing the Capabilities of On-Chip Voltage Sensors against Remote Power Attacks on FPGAs," *IEEE International Midwest Symposium on Circuits and Systems*, pp. 941–944, 2020.

[112] Tohoku University, "Cryptographic Hardware Prject." `http://www.aoki.ecei.tohoku.ac.jp/crypto/web/cores.html`, 2007. Last accessed on 09/05/2021.

[113] Xilinx, "Virtual Input/Output v3.0." `https://www.xilinx.com/support/documentation/ip_documentation/vio/v3_0/pg159-vio.pdf`, 2018. Last accessed on 09/05/2021.

[114] NUEESS Lab Northeastern University, "Side Channel Analysis Library." `https://bit.ly/2Y6k0ZD`, 2013. Last accessed on 09/05/2021.

[115] O. Glamočanin, L. Coulon, F. Regazzoni, and M. Stojilović, "Are Cloud FPGAs Really Vulnerable to Power Analysis Attacks?," *IEEE/ACM Design, Automation & Test in Europe*, pp. 1007–1010, 2020.

[116] Xilinx, "Vivado Design Suite Properties Reference Guide." `https://bit.ly/2XeABu1`, 2020. Last accessed on 09/09/2021.

[117] Xilinx, "Power Methodology Guide." https://www.xilinx.com/support/documentation/sw_manuals/xilinx14_7/ug786_PowerMethodology.pdf, 2013. Last accessed on 09/05/2021.

[118] R. Thomas, "Advanced Encryption Standard (AES): What It Is and How It Works." https://bit.ly/3zOeQzd, 2020. Last accessed on 09/05/2021.