TAL TECH

DOCTORAL THESIS

Advanced Hardware Protection Mechanisms: A Study on Logic Locking and Circuit Obfuscation Techniques

Antonio Felipe Costa de Almeida

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY
TALLINN 2025

TALLINN UNIVERSITY OF TECHNOLOGY
DOCTORAL THESIS
42/2025

Advanced Hardware Protection Mechanisms: A Study on Logic Locking and Circuit Obfuscation Techniques

ANTONIO FELIPE COSTA DE ALMEIDA

TAL TECH PRESS

TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Department of Computer Systems

The dissertation was accepted for the defence of the Doctor of Philosophy in Information and Communication Technology degree on 5th June 2025

Supervisor: Professor Dr. Samuel Pagliarini,
Department of Computer Systems, Centre for Hardware Security,
Tallinn University of Technology,
Tallinn, Estonia

Co-supervisor: Dr. Levent Aksoy,
Department of Computer Systems, Centre for Hardware Security,
Tallinn University of Technology,
Tallinn, Estonia

Opponents: Professor Dr. Shahin Tajik,
Department of Electrical and Computer Engineering,
Worcester Polytechnic Institute,
Worcester, United States

Professor Dr. Domenic Forte,
Department of Electrical and Computer Engineering,
University of Florida,
Gainesville, United States

Defence of the thesis: 16th June 2025, Tallinn

Declaration:
Hereby, I declare that this doctoral thesis, my original investigation, and achievement, submitted for the doctoral degree at Tallinn University of Technology, has not been submitted for any academic degree elsewhere.

Antonio Felipe Costa de Almeida _____
signature

European Union
European Regional
Development Fund

Investing
in your future

ISSN 2585–6901 (PDF)
ISBN 978-9916-80-323-3 (PDF)
DOI https://doi.org/10.23658/taltech.42/2025

Costa de Almeida, A. F. (2025). *Advanced Hardware Protection Mechanisms: A Study on Logic Locking and Circuit Obfuscation Techniques* [TalTech Press]. https://doi.org/10.23658/taltech.42/2025

TALLINNA TEHNIKAÜLIKOOL
DOKTORITÖÖ
42/2025

Täiustatud riistvara kaitsemehhanismid: uuring loogikalukustamise ja hägustamise tehnikate kohta

ANTONIO FELIPE COSTA DE ALMEIDA

TAL TECH
KIRJASTUS

Contents

List of Publications

The present PhD thesis is based on the following publications.

[I] L. Aksoy, Q. -L. Nguyen, F. Almeida, J. Raik, M. -L. Flottes, S. Dupuis, and S. Pagliarini, "Hybrid Protection of Digital FIR Filters," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 31, no. 6, pp. 812-825, 2023

[II] F. Almeida, L. Aksoy, Q. -L. Nguyen, S. Dupuis, M. -L. Flottes and S. Pagliarini, "Resynthesis-based Attacks Against Logic Locking," in 24th International Symposium on Quality Electronic Design (ISQED), pp. 1-8, 2023

[III] F. Almeida, L. Aksoy, and S. Pagliarini, "RESAA: A Removal and Structural Analysis Attack Against Compound Logic Locking," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 33, no. 3, pp. 1–13, 2025

Other related publications

The author has contributed to other publications during his studies at Tallinn University of Technology.

[IV] L. Aksoy, Q. -L. Nguyen, F. Almeida, J. Raik, M. -L. Flottes, S. Dupuis, and S. Pagliarini, "High-level Intellectual Property Obfuscation via Decoy Constants," in IEEE 27th International Symposium on On-Line Testing and Robust System Design (IOLTS), pp. 1-7, 2021

[V] M. Imran, F. Almeida, A. Basso, S. S. Roy, and S. Pagliarini, "High-speed SABER key encapsulation mechanism in 65nm CMOS.," in Journal of Cryptographic Engineering, pp. 461–471, 2023

[VI] F. Almeida, M. Imran, J. Raik and S. Pagliarini, "Ransomware Attack as Hardware Trojan: A Feasibility and Demonstration Study," in IEEE Access, vol. 10, pp. 44827-44839, 2022

[VII] F. Almeida, L. Aksoy, J. Raik and S. Pagliarini, "Side-Channel Attacks on Triple Modular Redundancy Schemes," in IEEE 30th Asian Test Symposium (ATS), pp. 79-84, 2021

[VIII] M. Imran, F. Almeida, J. Raik, A. Basso, and S. Pagliarini, "Design Space Exploration of SABER in 65nm ASIC," in Proceedings of the 5th Workshop on Attacks and Solutions in Hardware Security (ASHES), pp. 85–90, 2021

Contribution of the Author to the Publications

The author's contributions to the core publications included in this thesis are outlined below:

- **Publication *I*:** As the third author, I contributed specifically to the evaluation of compound logic locking (CLL) schemes. My contributions focused on designing and implementing circuits employing CLL, generating synthesis metrics such as area, delay, and power consumption, and assessing their security properties. Additionally, I participated in attack assessments to measure the effectiveness of the protection mechanisms applied.
- **Publication *II*:** As the first author, I led the research on a resynthesis-based attack against logic locking (LL), managing all stages of the study. This work introduced a methodology for manipulating locked netlists using commercial EDA tools to expose vulnerabilities. I was responsible for designing and implementing the attack framework, conducting experiments, and analyzing results to assess the effectiveness of the proposed approach. Furthermore, I prepared the manuscript, including the presentation of both methodology and results.
- **Publication *III*:** As the first author, I developed the RESAA framework for analyzing and attacking CLL schemes. My work involved designing and implementing a system capable of classifying locked netlists, identifying critical structures, and applying targeted attacks to uncover secret keys. I refined the attack strategies within RESAA, improving their efficiency and increasing their success rate against various protection schemes. Experimental validation demonstrated the effectiveness of RESAA in analyzing LL techniques. I also prepared the manuscript, presenting these findings in a structured and detailed manner.

In addition to these core publications, I have contributed to other research papers published during my PhD studies at Tallinn University of Technology. These works, listed under **Other Related Publications**, cover additional aspects of hardware security, cryptographic implementations, and hardware attack strategies.

Abbreviations

AGR	AppSAT Guided Removal
AI	Artificial Intelligence
AppSAT	Approximate SAT
ATPG	Automatic Test Pattern Generation
BLE	Bilateral Logic Encryption
CAC	Corrupt and Correct
CASLock	Corruptibility and Security Locking
CAVM	Constant Array Vector Multiplication
CG	Critical Gate
CLL	Compound Logic Locking
CNF	Conjunctive Normal Form
CRK	Constant Replacement with Key
DFLT	Double-Flip Logic Locking Technique
DIP	Distinguishing Input Pattern
DSP	Digital Signal Processing
DTL	Diversified Tree Logic
EDA	Electronic Design Automation
Fa-SAT	Fault-Aided SAT
FIR	Finite Impulse Response
FLL	Fault-Based Logic Locking
FPGA	Field-Programmable Gate Array
FSM	Finite State Machine
GNN	Graph Neural Network
HD	Hamming Distance
HLS	High-Level Synthesis
IC	Integrated Circuit
IoT	Internet of Things
IP	Intellectual Property
KA	Knowledgeable Adversary
LL	Logic Locking
LOOPLock	Logic Optimization-Based Cyclic Logic Locking
LUT	Look-Up Table
MCM	Multiple Constant Multiplication
ML	Machine Learning
MUX	Multiplexer
OA	Oblivious Adversary
OG	Oracle-Guided
OL	Oracle-Less
OoT	Out of Time
PO	Primary Output
PSLL	Provably Secure Logic Locking
PUF	Physically Unclonable Function
QATT	Query Attack

QBF	Quantified Boolean Formula
RLL	Random Logic Locking
RTL	Register Transfer Level
SA	Specific Adversary
SAT	Satisfiability
SCOPE	Synthesis-Based Constant Propagation Attack for Security Evaluation
SFLL	Stripped Functionality Logic Locking
SFLT	Single-Flip Logic Locking Technique
SKG	SAT-Resistant Key Gate
SLL	Strong Logic Locking
SoC	System-on-Chip
TGA	Topology-Guided Attack
TMCM	Time-Multiplexed Constant Multiplication
TTLock	Tenacious and Traceless Logic Locking

List of Figures

List of Tables

1 Introduction

The increasing integration of hardware systems across various industries, including automotive, defense, telecommunication, and beyond, has raised concerns about the security of hardware components. As hardware becomes more complex and interconnected, it faces more significant risks to the security of integrated circuits (ICs), mainly when intellectual property (IP) cores and chips are outsourced for manufacturing in the globalized supply chain [1]. The main concerns include reverse engineering, IC counterfeiting, overproduction, IP piracy, and the insertion of hardware Trojans [2].

Reverse engineering allows adversaries to deconstruct and analyze proprietary designs, uncovering their structure, functionality, and underlying technologies. This process exposes sensitive IP to unauthorized access, industrial espionage, and potential exploitation and facilitates the creation of counterfeit or pirated versions of the original designs, causing financial losses to the original creators [3]. Counterfeiting involves the unauthorized reproduction of proprietary designs, replicating original designs without permission. These imitations often compromise quality and reliability, creating significant security risks and introducing potential vulnerabilities [4]. Overproduction occurs when manufacturers produce quantities exceeding authorized limits, often without the knowledge or consent of the intellectual property owner [5]. IP piracy involves the illegal use of designs to produce unauthorized ICs [2], and hardware Trojans introduce malicious logic that can compromise both functionality and reliability [6].

Various countermeasure techniques have been developed, each offering different levels of protection and trade-offs regarding area, power, and delay overheads. These techniques include split manufacturing, hardware metering, watermarking, and hardware obfuscation, which encompasses logic locking (LL).

In split manufacturing, the metal layers of the IC are divided and fabricated at different foundries to mitigate security risks [7, 8]. Hardware metering involves real-time monitoring of resource usage within the IC to prevent piracy by tracking and regulating the allocation of hardware resources, ensuring secure and efficient utilization [9, 10]. Watermarking allows for the detection of IP theft and misuse by embedding signatures into the design without changing functionality [11, 12].

Hardware obfuscation plays an important role in preventing unauthorized access by modifying circuit architecture, making it significantly more difficult for adversaries to decipher or reverse engineer its functionality. This method effectively hides the correct operation of the circuit, safeguarding it from malicious adversaries [13]. LL is a specific kind of obfuscation technique that uses key-driven gates to ensure that the circuit operates correctly only when the appropriate key is provided [14–19].

Over the years, several LL techniques have been proposed, ranging from simple XOR/XNOR-based designs to more sophisticated approaches incorporating multiple locking strategies for enhanced security. While these methods can successfully obscure circuit functionality against older attacks [20], they remain vulnerable to more advanced threats [21]. As a result, continuous research into more robust and efficient LL techniques is essential for ensuring long-term security. However, LL poses challenges, particularly in balancing security with overhead in hardware complexity in terms of area, delay, and power dissipation. High resource usage can be especially problematic for designs with

low-power requirements, such as Internet of Things (IoT) devices [21]. Maintaining this balance is essential for achieving security and efficiency in resource-limited environments.

One of the most well-known attack methods targeting LL techniques is the Satisfiability (SAT)-based attack, which systematically removes incorrect keys by finding distinguishing input patterns (DIPs) [20]. Introduced in 2015, this attack efficiently reduces the key search space, compromising even advanced LL schemes characterized by a large number of key bits and increased hardware complexity designed to improve resiliency against adversarial attacks [22, 23]. In response, designers have developed SAT-resilient strategies to counter SAT-based attacks, which significantly increase the computational difficulty for such adversaries [16, 24]. Additionally, efforts have been made to address other emerging threats, such as structural analysis and removal attacks, which exploit different vulnerabilities in locking mechanisms [25, 26]. However, as adversaries continue to refine and combine these techniques, securing hardware remains a dynamic and evolving challenge.

Effectively addressing SAT-based attacks requires a clear understanding of the limitations of current defenses. While SAT-resilient techniques, such as Anti-SAT [27] and SARLock [16], have been proven to be effective against SAT-based attacks, they suffer from other challenges when integrated into complex designs due to their hardware complexity overhead and limited output corruption [28]. A combination of LL techniques, adding an extra layer of protection, has been explored. Methods like compound logic locking (CLL) have been developed to combine high output corruption with SAT-resilient mechanisms, increasing the difficulty of key recovery for attackers [29, 30]. However, these enhanced security measures can significantly impact the design's complexity, increasing area, power, and delay, which may pose additional challenges [31].

Attacks on LL are generally divided into oracle-guided (OG) and oracle-less (OL) approaches. In addition to the locked netlist, OG attacks leverage a functional IC as an oracle to compare inputs and outputs, systematically deducing the secret key [20, 32]. SAT-based attacks are examples of this kind of attack and are effective in this context. In contrast, OL attacks assume that the attacker has access only to the locked netlist and no functional IC, making the key extraction process more challenging but still feasible through methods such as *resynthesis-based attacks*, which leverage electronic design automation (EDA) tools to resynthesize the design based on various key guesses, aiming to converge toward the correct solution [33] or to generate functionally equivalent versions of the locked netlist and analyze them for vulnerabilities [34].

As OG and OL attacks become more sophisticated, defenses must evolve to account for both the structural weaknesses in designs and the tools attackers may use. This ongoing cat-and-mouse game between attackers and designers drives the continued advancement of security measures.

1.1 Contribution of this Thesis

This thesis is a compilation of three published papers, as shown in Fig. 1. Each paper addresses specific research questions and challenges related to LL and its vulnerabilities, contributing to a deeper understanding of attack strategies and countermeasures.

The TVLSI 2023 paper investigates the following research question: Can an attacker

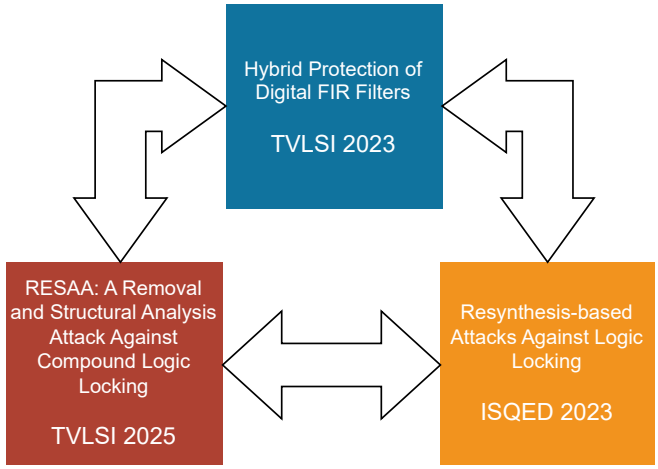


Figure 1: Three foundational works of this thesis.

extract the secret key from an obfuscated finite impulse response (FIR) filter despite existing obfuscation techniques? The paper introduces a query attack that strategically selects input queries to deduce key bits while bypassing current defenses. The results demonstrate that traditional obfuscation schemes fail against this attack, revealing critical security gaps. Furthermore, the paper proposes a hybrid defense strategy that combines hardware obfuscation with LL, enhancing security without significantly impacting implementation constraints [35].

The ISQED 2023 paper questions a common assumption in LL research by asking: How do the synthesis EDA tools impact the security of LL circuits? This study introduces the resynthesis-based attack, showing that transformations introduced during synthesis can weaken LL protections. By generating multiple structurally different but functionally equivalent versions of a locked circuit using EDA tools, attackers can significantly amplify the effectiveness of existing attacks to recover the secret key. The results demonstrate that even advanced LL techniques remain vulnerable, emphasizing the necessity of resilient LL approaches across different synthesis parameters [34].

The TVLSI 2025 paper addresses the research question: Does CLL improve security against attacks or introduce new vulnerabilities? The paper presents RESAA, a framework designed to systematically analyze CLL designs, identify weak points, and execute targeted attacks under both OL and OG models. By partitioning the CLL circuit, RESAA successfully exploits inherent weaknesses in multi-layer LL techniques, improving attack success rates. Experimental results reveal that RESAA can break a wide range of CLL variants, demonstrating fundamental limitations in CLL security and highlighting the need for stronger protection mechanisms [30].

Through these contributions, this thesis advances the field of hardware security by systematically analyzing and exposing weaknesses in both traditional and hybrid LL techniques. The proposed attack methodologies and defensive strategies provide valuable information on the evolving landscape of hardware security, helping shape the development of more resilient countermeasures against LL attacks.

1.2 Outline of this Thesis

The remainder of this thesis is organized as follows:

- **Section 2: Background** – This section provides an overview of hardware security, focusing on essential concepts, such as the IC supply chain and its associated challenges, hardware obfuscation, LL, and various attack models. It establishes a foundation by explaining the core mechanisms and challenges in developing secure ICs.
- **Section 3: Discussions** – This section explores the interrelation between the three papers, covering defensive and offensive hardware security strategies. The first paper introduces a **query attack**, a novel technique capable of breaking obfuscated FIR filters by identifying their hidden coefficients. To counter this, it also presents a **hybrid defense approach** that combines hardware obfuscation with LL, leveraging decoy obfuscation and point functions to enhance security while maintaining competitive hardware complexity. The second paper proposes a **resynthesis-based attack**, which systematically manipulates locked netlists using commercial EDA tools to expose their vulnerabilities. By generating multiple functionally equivalent but structurally different versions of a circuit, this approach reveals weaknesses that remain undetected by traditional attacks, significantly increasing the number of deciphered key bits. The third paper introduces **RESAA**, a framework designed to analyze and attack CLL circuits. RESAA classifies locked netlists, identifies critical gates (CG), and applies structural analysis to expose secret keys. Together, these works contribute to a deeper understanding of both attack methodologies and resilient defense strategies in LL.
- **Section 4: Conclusion and Future Work** – This section summarizes the key contributions from each study and discusses their broader impact on hardware security. It also outlines potential directions for future research, such as enhancing protection techniques, exploring more advanced attack models, and incorporating artificial intelligence (AI)-driven tools to strengthen IC security further.

2 Background

2.1 IC Supply Chain and Security Challenges

Figure 2 presents a generic design flow for ICs composed of many stages. The process begins with **specification and behavioral design**, where the IC's functionality and performance requirements are defined. This step is managed by the design house, a trusted entity that establishes the intended behavior, focusing only on meeting the specifications and design goals. Although this phase involves minimal participation from external parties, there are still potential threats. For example, inside threats or mismanagement of sensitive design data could lead to unintentional leaks or targeted theft. Additionally, adversaries may analyze early design knowledge to identify potential weaknesses in the later stages of the design process [36].

Strict access control policies and secure data management protocols should be implemented to mitigate these risks, especially as cloud-based IC design platforms become more widely adopted [37]. The use of watermarking can also help identify unauthorized use of design data [38]. Regular audits and training for design teams can further minimize the likelihood of insider threats.

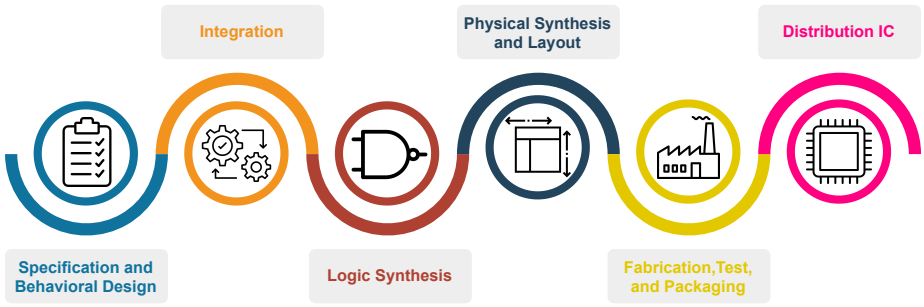


Figure 2: Generic IC design flow.

In the **integration** phase, often done by a contracted team, various IP blocks are assembled into a unified system-on-chip (SoC). Integration poses heightened risks because untrusted entities may gain visibility over the entire chip, enabling them to alter or manipulate the top-level design. An attacker involved in integration could insert hidden vulnerabilities or introduce malicious components. In addition, untrusted IP providers or external design teams could embed subtle backdoors to facilitate reverse engineering or compromise the system's integrity [39].

Verification procedures, such as IP-level integrity checks and top-level validation against tampering, are crucial to mitigate these threats during the integration [40]. Utilizing hardware obfuscation techniques such as LL ensures that sensitive components remain protected even if exposed during integration [41, 42]. Furthermore, employing trusted design environments and ensuring that all IPs are vetted and certified reduces the likelihood of adversarial modifications [43]. Trusted engineers at the design house must closely monitor the integration process to ensure that no unauthorized modifications occur.

In the **logic synthesis** phase, the high-level design is mapped into a gate-level

netlist representing the circuit regarding logic gates and connections. Also, it is the responsibility of the design house, which may involve using integrated third-party IP cores. Unverified third-party IPs integrated during this phase could pose significant risks, including unauthorized data leakage and compromised functionality.

Rigorous verification processes, including checks for malicious alterations and validation of the netlist against design specifications, help ensure the integrity of the output. Hardware security techniques, such as the IEEE P1735 standard for netlist encryption [44], or logic obfuscation, can also deter adversarial exploitation [45]. Collaboration with certified IP providers and maintaining control over the entire synthesis process further mitigates risks and upholds the design's security.

The netlist is converted into a physical representation during **physical synthesis and layout**, from floorplan to place and route information within the IC layout. This phase often involves external parties that may not be fully trusted, especially when outsourcing the physical implementation to third-party companies. Here, untrusted layout engineers or third-party contractors gain access to a more detailed view of the design. An adversary with access to this phase could attempt to reverse-engineer parts of the layout or insert malicious modifications, mainly if they are familiar with layout tools and techniques [46].

In **fabrication, test, and packaging**, the fabrication involves creating the physical IC from its design and translating the layout into silicon. Testing ensures that the fabricated chip functions as intended, identifying defects and failures, and in the packaging step, the chip is enclosed in a protective casing to interface with external systems. These steps, often outsourced to external foundries and facilities, introduce risks as untrusted entities gain access to critical design information. At this stage, adversaries can leverage complete visibility of the chip layout to attempt reverse engineering, necessitating countermeasures such as obfuscation utilized in previous steps to prevent unauthorized duplication or tampering [47].

Finally, the chip reaches the market in **distribution IC**, gets integrated into end-user products, or is deployed for specific applications. Even at this stage, security threats persist, including counterfeiting and tampering during distribution. Adversaries may attempt to duplicate the product, compromise its functionality, or extract proprietary information through reverse engineering [48].

Manufacturers can mitigate these risks by employing anti-counterfeiting measures such as unique identifiers, secure boot mechanisms, and hardware-based authentication protocols [49–51]. Supply chain monitoring and traceability systems ensure that only authorized and secure ICs reach the end-user. In summary, each phase of the IC design flow presents distinct security challenges, requiring tailored countermeasures to enhance the integrity and resilience of the final product.

Figure 3 shows the most common integration of LL techniques to mitigate security threats. LL is introduced by embedding security features directly into the IC's design as part of the **Logic Locking** process. Applying LL early in the flow safeguards critical design components from tampering or reverse engineering attempts, ensuring that the IC remains non-functional until the correct activation key is provided. This approach effectively protects the design's intellectual property and functionality.

Upon completion of manufacturing, the **key activation** phase unlocks the IC's

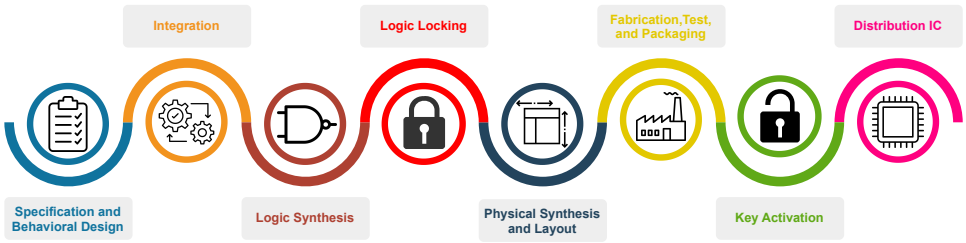


Figure 3: Locking locking in the IC design flow.

functionality. This step is typically managed by the same design house that initiated the design flow and involves securely provisioning the activation (secret) key to the locked design. After fabrication, the locked circuit undergoes the key activation phase, where the secret key is securely stored in a tamper-proof memory within the chip. Ensuring that the key remains protected is critical because if it is compromised, untrusted entities could deactivate or duplicate the IC, rendering the LL measures ineffective.

Once key activation is complete, the IC enters the market as a fully **functional IC**. However, it still faces threats such as counterfeiting, cloning, and reverse engineering, particularly from adversaries with physical access to the chip. These adversaries may exploit invasive or semi-invasive methods to extract sensitive design information. LL mitigates these risks by ensuring that any unauthorized replication or tampering fails without the secret key, preserving the IC's integrity and protecting its intellectual property.

Trusted teams oversee the design, synthesis, and secure key activation processes. At the same time, untrusted actors—including external IP vendors, third-party tool providers, offshore foundries, and unauthorized market participants—pose substantial risks throughout the supply chain. LL is a fundamental defense mechanism, extending protection from the initial design stages to the IC's market deployment. The diversity of adversaries, from insiders to external attackers, underscores the importance of embedding strong security measures at every stage. The following subsection thoroughly discusses hardware obfuscation techniques, emphasizing LL as a key strategy for enhancing IC security.

2.2 Hardware Obfuscation and Logic Locking

2.2.1 Hardware Obfuscation

Hardware obfuscation is a security technique developed to protect the internal design of an IC by making it challenging to analyze, reverse-engineer, and manipulate. By modifying the design to obscure its correct functionality from unauthorized users, obfuscation aims to make it computationally infeasible for adversaries to extract meaningful information, thereby securing IP and preventing malicious alterations [14, 52, 53].

Obfuscation methods can be applied at various stages of the design flow:

- **High-level Obfuscation**: At the high-level design stage, obfuscation targets critical components such as proprietary algorithms, data paths, or constants. This is achieved by introducing decoy elements or misleading logic to obscure sensitive

functions, making it difficult for adversaries to understand the system's intent. Using high-level synthesis (HLS) tools to embed obfuscation directly into the design ensures that sensitive portions' intent remains protected throughout the flow [54–56].

- **Behavioral-Level Obfuscation**: At the register-transfer level (RTL), obfuscation involves altering the control and data flow of the design [57]. Techniques include obscuring finite state machines (FSMs) by modifying state transitions, introducing additional states, and leveraging reconfigurable key-based FSMs [32]. This makes it harder to reverse-engineer the control logic, which explores the interplay between security and functionality in behavioral-level obfuscation [58].

- **Gate-Level Obfuscation**: After logic synthesis, gate-level obfuscation is applied to modify the logical structure of the design. This includes inserting additional gates, modifying or removing existing gates, or embedding techniques such as LL. These modifications obscure the circuit's functionality, requiring a secret key for correct operation, protecting the IP from reverse engineering [15, 16, 28, 59].

- **Layout-Level Obfuscation**: At the physical design stage, layout-level obfuscation or camouflaging is used to protect the physical representation of the IC. This technique involves designing the layout so that different logic functions appear identical, making it challenging for attackers to identify the actual functionality of each component. For example, using standard-cell libraries with indistinguishable layouts or introducing decoy components can significantly enhance protection against physical reverse engineering [25, 60].

By embedding obfuscation techniques throughout the design flow—from high-level to layout-level—hardware designers can ensure robust protection against various adversaries, securing IP and mitigating threats effectively. The following subsection covers the details of LL and its application and efficacy.

2.2.2 LL: A Form of Hardware Obfuscation

LL is a specialized form of hardware obfuscation that works by inserting key-driven gates into the design. These gates ensure that the circuit behaves correctly only when the correct key is applied. Without the secret key, the circuit produces incorrect outputs or remains non-functional, protecting the IP from reverse engineering and unauthorized use [61].

The evolution of LL has seen several variations aimed at improving both security and efficiency. After the introduction of random logic locking (RLL) using XOR/XNOR gates [14], subsequent research expanded to explore different types of key gates, such as AND/OR gates, multiplexors, and look-up tables, while considering the hardware complexity of these gates introduced into the locked circuit [17].

Despite these advances, the original defenses were eventually compromised by the development of the SAT-based attack [20]. Provably secure logic locking (PSLL) was introduced as a paradigm offering formal security guarantees against known attack methods, incorporating point functions that limit the number of incorrect keys DIPs

can eliminate. Recall that a DIP is an input vector that produces different outputs on the locked netlist with two different keys, allowing an attacker to refine the correct key iteratively. By restricting the effectiveness of DIPs, PSLL forces attackers to explore an exponential number of them, making key recovery infeasible [27, 62–65].

In this context, traditional LL techniques can be categorized into two main groups: single-flip locking technique (SFLT) and double-flip locking technique (DFLT). Figure 4(a) presents an SFLT, which relies on a single critical point in the circuit that corrupts an output under a specific input pattern. Although SFLTs demonstrate resilience against SAT-based attacks, they are vulnerable to removal attacks, where an attacker can identify and eliminate the critical point, separating the design into the original netlist and locking unit [25, 66, 67]. On the other hand, in Figure 4(c), DFLTs improve security by introducing two critical points: one in the perturb unit, which initially corrupts an output, and another in the restore unit, which corrects the output when the correct key is applied. While this approach enhances security, DFLTs remain susceptible to advanced structural attacks that exploit the interconnections between the perturb and restore units and their integration with the original circuit [24, 33, 68, 69].

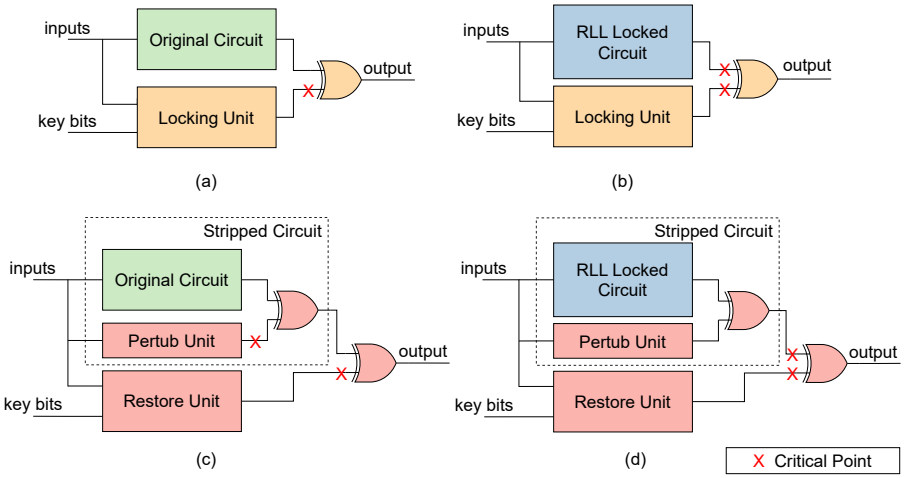


Figure 4: High-level architecture of (a) SFLT, (b) RLL + SFLT, (c) DFLT, and (d) RLL + DFLT in a CLL scheme. This figure is reproduced from Figure 2 in Publication III.

Efforts to strengthen LL techniques have taken various directions, aiming to overcome their perceived weaknesses. These approaches include the insertion of cyclic logic [70], the use of emerging materials [71], and look-up table (LUT)-based obfuscation [56, 72]. Each method introduces additional complexity to the locking mechanism, making it more challenging for adversaries to bypass the protection. However, no single method has been proven entirely secure against all attacks, which has led to the development of more advanced strategies.

Despite the potential of CLL, research into attacks specifically targeting this hybrid technique remains limited, and the exploration of such attacks has only begun to scratch the surface. For example, the combined use of SAT-based and structural attacks against CLL has been studied [73, 74], but these studies are confined to specific combinations of techniques. This underscores the broader need for more comprehensive

research to understand and fully mitigate the vulnerabilities in CLL designs. One of the main contributions of this thesis is that it introduces a novel attack strategy that reveals previously neglected weaknesses in CLL, offering critical insights into its security limitations.

Figures 4(b) and 4(d) exemplify CLL, which integrates double-layer LL techniques to improve the security of ICs. Note that RLL is always used as it delivers the critical feature of (high) output corruption. By combining RLL with other techniques, CLL strives to take advantage of their respective strengths while mitigating individual weaknesses. This combined strategy fortifies security by exploiting complementary aspects of diverse LL techniques, selecting specific corruption levels, and tailoring SAT resilience to optimize protection against attacks. In these cases, a CG is identified in which one of its inputs consists exclusively of key inputs from RLL, while the other input of the CG incorporates key inputs from PSLL. This configuration enhances security by intertwining distinct locking techniques and increasing the complexity of the attack. However, the presence of CGs also introduces a potential dependency between the two layers that attackers may attempt to exploit.

2.3 Threat Models in Logic Locking

Understanding the threat models involved in attacking LL techniques is essential for evaluating their effectiveness. Threat models help identify potential adversaries and their capabilities, which is crucial for securing hardware designs against attacks. There exist two main threat models: **OG** and **OL** models.

2.3.1 OG Threat Model

In addition to the locked netlist, the attacker has access to a functional IC that can be used to query inputs and observe the corresponding outputs. This model assumes that the attacker has the locked netlist and the capability to apply inputs to the functional IC but does not have direct access to the secret key.

SAT-based attacks are the most prominent example of OG attacks [16, 23, 75, 76]. These attacks use an oracle to iteratively eliminate incorrect key guesses by applying DIPs that expose inconsistencies between the locked netlist and the functional IC. The objective is to eliminate incorrect key guesses and continually reduce the search space. These attack methods and others have highlighted the effectiveness of SAT-based methods in exploiting vulnerabilities in LL schemes.

Beyond SAT-based attacks, adversaries may employ other methods to undermine logic locking. Removal attacks focus on identifying and bypassing the obfuscation structures within the locked circuit [25]. Approximation attacks, on the other hand, attempt to create a simplified model of the circuit that replicates its behavior without requiring the original key [23].

2.3.2 OL Threat Model

The OL threat model represents a more restrictive scenario for an attacker, assuming that only the locked netlist is available without access to the functional IC. Unlike the OG threat model, where the attacker can apply inputs to a functional IC and observe its

outputs, the OL model does not allow direct simulation of the correct circuit behavior. This makes the OL threat model significantly more challenging than the OG threat model since there is no oracle to check the original circuit behavior.

In OL attacks, adversaries often use machine learning (ML) techniques, structural analysis, or resynthesis-based attacks. These attacks can propagate a constant to find anomalies in the design after synthesis or take advantage of EDA tools to generate functionally equivalent but structurally different versions of the locked netlist [34, 77, 78]. By comparing these versions, attackers can identify patterns that may reveal the secret key.

Despite the increased difficulty of the OL model, recent advances in attack strategies have demonstrated its abilities. ML approaches have been shown to accurately predict significant portions of the key by analyzing structural features in the locked netlist [79]. Structural analysis techniques, such as identifying discrepancies in wire delays or redundant paths introduced by locking mechanisms, further expose potential weaknesses [80]. Resynthesis-based attacks remain particularly potent, exploiting design inconsistencies during synthesis and optimization, effectively reducing the design's obfuscation strength [34]. These methods illustrate how attackers exploit the lack of functional IC by taking advantage of sophisticated tools and algorithms.

Moreover, advanced techniques, such as structural pruning and graph-based analysis, have been applied to detect areas of high obfuscation density, effectively narrowing down the key search space [81]. These findings highlight that while the OL model imposes significant constraints on the attacker, the rapid evolution of attack methodologies necessitates more resilient and multi-layered defense strategies in LL.

In addition to netlist-based analysis, OL adversaries may exploit physical access to extract the secret key. In [82], it was shown that the pathway between the key storage and the logic gates can be vulnerable, even when tamper-proof memory is used. Optical probing techniques were successfully used to extract the values of key inputs during runtime. This ability to recover the secret key without a functional IC reinforces the need to protect both the key storage and the signal path within the chip.

2.4 LL Defenses

Various defense techniques have been developed to strengthen LL's security against known attacks, focusing on improving resilience while balancing area, power consumption, and delay overhead. These techniques are generally classified into Pre-SAT, Post-SAT, and Beyond-SAT approaches, each addressing different aspects of attack resistance and security enhancement.

2.4.1 Pre-SAT Techniques

Traditional LL techniques were developed before SAT-based attacks became a significant threat. Among the earliest methods, RLL aimed to obfuscate circuit functionality by randomly selecting wires to be locked and inserting additional gates controlled by key inputs. While XOR/XNOR gates were commonly used, other gate types have been introduced depending on the locking strategy. In this case, the locked circuit works as intended only when the correct key is applied [14]. However, due to its random nature,

RLL remained vulnerable to structural analysis and SAT-based attacks that exploited its predictable behavior.

Another early approach, the LUT-based technique, replaced certain circuit parts with programmable LUTs that required key inputs to function correctly. Unlike RLL, which inserted additional gates, LUT-based locking modified the circuit topology, making structural analysis more challenging for attackers attempting to reverse engineer or extract sensitive information [83]. However, while increasing complexity, this technique did not directly counteract SAT-based attacks.

MUX-based techniques introduced an alternative form of obfuscation by replacing certain logic elements with multiplexers (MUXes), where the secret key determined the active logic path. This approach increased ambiguity by encoding multiple logic paths, ensuring that only the correct key selected the intended functionality. Compared to RLL and LUT-based locking, MUX-based locking created structural redundancy, complicating key extraction through direct circuit analysis. However, similar to other Pre-SAT LL methods, it remained vulnerable to OG attacks, where an attacker could infer the correct logic paths by comparing locked and unlocked circuit responses.

Strong logic locking (SLL) was an improvement over RLL aimed at increasing the complexity of key recovery attacks by introducing interdependencies between key bits. Unlike RLL, where each key bit could be analyzed independently, SLL made key bits interdependent, preventing simple brute-force approaches from solving them individually. Additionally, incorrect key values resulted in significant output corruption, making it difficult for an attacker to reconstruct the correct circuit behavior through trial and error [22]. However, despite strengthening resistance against brute force and structural attacks, SLL remained vulnerable to SAT-based attacks.

In addition, some LL techniques were introduced to counter fault-injection attacks, which attempt to extract key information by inducing controlled errors into the circuit. Fault-based logic locking (FLL) was designed to protect against such threats by incorporating fault detection and correction mechanisms, ensuring that any induced faults propagated misleading information to the attacker [84]. FLL was particularly useful in scenarios where physical attacks were a concern, such as embedded systems and cryptographic hardware. However, like other Pre-SAT techniques, it did not directly counter SAT-based attacks, as its primary function was to mislead adversaries by manipulating circuit behavior rather than preventing logic decryption.

2.4.2 Post-SAT Techniques

Anti-SAT was one of the first techniques explicitly designed to mitigate SAT-based attacks by introducing complementary logic trees. As an SFLT, it employs two complementary functions, e.g., AND- and NAND-tree structures, in such a way that the locking circuit shown in Figure 4(a), always generates a constant logic value under all input patterns if the key values are correct and otherwise, generates either 0 or 1 depending on the inputs. It also guarantees that the maximum number of unique wrong keys is close to the exponential number of key inputs. Thus, Anti-SAT forces the SAT-based attacks to explore an exponential number of key possibilities, significantly increasing the time complexity of the attack [27]. However, despite its SAT resilience, Anti-SAT's logic structure can be identified and removed through structural analysis,

making it vulnerable to removal-based attacks.

Anti-SAT-DTL (Diversified Tree Logic) was introduced to address the structural detectability of Anti-SAT. Instead of relying solely on AND/NAND trees, this technique randomly replaces some AND gates with OR/NAND/XOR gates, making the logic structure more unpredictable [36]. These modifications prevent an attacker from quickly identifying and eliminating the Anti-SAT circuit, ensuring that SAT resilience remains intact while reducing vulnerabilities to structural analysis.

Also, as an SFLT, SARLock ensures that only a small subset of inputs yields differing outputs when incorrect keys are applied, limiting the attacker's ability to find DIPs that can eliminate more than one wrong key in each iteration [16]. It adds a layer of protection by making SAT solvers ineffective, as the output for most incorrect keys remains unchanged [27].

InterLock and CASLock were introduced as advanced LL techniques to enhance security against attacks. InterLock improves security by linking critical key bits to specific circuit functions and intercorrelating logic and routing paths, ensuring that incorrect keys lead to unpredictable behavior across various circuit regions. This interdependence creates a complex structure that makes SAT-based attacks highly ineffective, complicating the analysis for attackers [85]. CASLock, on the other hand, uses cascaded logic gates, significantly increasing the number of DIPs and forcing attackers to explore a much larger search space. The cascaded structure provides multiple layers of protection, each with its own locking mechanism, making it challenging for attackers to find a solution, even with sophisticated SAT attacks [63].

SKG-Lock extends the concept of SAT resistance by embedding SAT-resistant key gates (SKG) at multiple locations within the circuit. Unlike Anti-SAT, which relies on structured complementary logic trees, SKG-Lock strategically distributes SKGs across different regions of the design to increase attack complexity [65]. This approach disrupts the ability of SAT attacks to systematically eliminate incorrect keys while also making structural analysis-based attacks more challenging.

A significant advancement in post-SAT techniques was the introduction of stripped functionality logic locking (SFLL), which shifted the focus from DIP-based defenses to functionality removal strategies. Rather than limiting an attacker's ability to extract key information, as a DFLT, SFLL removes specific circuit functionalities, which can only be restored with the correct key [62]. This ensures that an incorrect key results in missing or altered behavior, making it significantly harder for an attacker to infer the correct logic structure. SFLL is particularly effective because it does not introduce easily detectable locking structures, increasing its resilience to both SAT-based and removal attacks.

A refinement of SFLL, tenacious and traceless logic locking (TTLock) enhances its effectiveness by ensuring that the locked design only differs from the original design for a single specific input pattern. This guarantees that, under incorrect keys, the circuit behaves identically to the unlocked version for all other input patterns, making it resistant to common attack analysis methods [28]. However, despite its traceless nature, an attacker can still identify critical points in the locked netlist [30, 69].

An evolution of SFLL, SFLL-HLS, took SFLL to a new level by incorporating obfuscation techniques during the high-level synthesis stage, creating context-sensitive

complex locks. As a DFLT, SFLL-HLS applies multiple layers of protection across functional units, making the circuit more resistant to SAT-based attacks [86].

Cyclic obfuscation introduces cyclic dependencies into the logic, creating feedback loops that mislead SAT solvers [87]. These cycles disrupt the fundamental assumption of acyclic combinational circuits used by traditional SAT-based attacks, causing the solver to enter infinite loops or fail to converge on a solution.

Logic optimization-based cyclic logic locking (LOOPLock) introduces cyclic dependencies into the logic, creating feedback loops that significantly increase the difficulty for SAT solvers to break the locking scheme [88]. By carefully optimizing these loops, LOOPLock ensures that the circuit is resistant to SAT-based attacks, as the solver is forced to handle the inherent complexity of cyclic dependencies, which creates misleading paths and significantly enlarges the solution space. Unlike traditional cyclic locking techniques, LOOPLock employs optimization techniques to minimize performance overhead while maintaining security, making it a more efficient and resilient defense mechanism.

CLL combines two or more LL techniques to create a multi-layered defense against various attack methods. The combination of techniques in CLL is designed to exploit the strengths of each method, thereby making the circuit highly resistant to SAT attacks while also complicating structural attacks [30]. The trade-off for this enhanced security is an increase in hardware overhead in terms of area, power, and delay. However, the benefits of improved security often outweigh the cost in applications demanding high security. Examples include bilateral logic encryption (BLE) and Anti-SAT combined with RLL. BLE introduces two complementary functions, ensuring that only the correct key can yield the correct output, while Anti-SAT with RLL leverages Anti-SAT's SAT resistance alongside RLL's high output corruption to enhance security by utilizing strengths from each technique [29]. These are examples of a CLL technique that integrates multiple locking strategies to reinforce protection against attacks.

Corrupt and Correct (CAC) 2.0 aims to combat structural analysis attacks. CAC 2.0 builds on previous LL techniques by implementing double CAC, applying obfuscation at multiple nodes with different protected inputs, and introducing decoy key values [59]. These enhancements increase the attack complexity, forcing adversaries to analyze multiple incorrect key paths before identifying the correct key. By integrating SAT resistance, structural complexity, and decoy-based misdirection, CAC 2.0 represents an advanced LL defense, effectively neutralizing both SAT-based and structural attacks.

2.4.3 Beyond SAT Techniques

Physically unclonable function-based (PUF-based) LL leverages intrinsic device properties to generate unique keys for each chip. PUFs generate keys based on the physical variations inherent to each manufactured chip, making it impossible for attackers to duplicate or reverse engineer the secret key [89]. This method is tied directly to the manufacturing process, ensuring that each device has a unique and unclonable identity. However, while PUFs prevent large-scale key extraction across multiple chips, they do not invalidate the proposed attacks. Since an adversary's primary goal is to copy the IP rather than compromise an entire batch, breaking a single chip remains sufficient to expose the locked design, making the chip and its IP potentially recoverable regardless of the presence of a PUF.

2.5 LL Attacks

LL has been widely adopted as a defense mechanism in hardware security to protect the IP. However, adversaries have continuously developed sophisticated attack techniques to bypass these protections.

This subsection presents an overview of the principal LL attack methodologies, categorizing them into OG attacks, which leverage a functional IC as an oracle, and OL attacks, which rely solely on the locked netlist for key recovery. Recently, a classification of adversaries based on their knowledge of the locking techniques was introduced in [30], defining three types of adversaries: specific adversary (SA), knowledgeable adversary (KA), and oblivious adversary (OA). SA attacks are tailored to a single, specific LL technique; KA attacks target a broader set of LL techniques where the adversary has prior knowledge of the locking mechanisms; and OA attacks use generic tools capable of yielding effective results without requiring prior knowledge of the LL techniques applied. Most of the existing attacks fall into the SA and KA categories. However, real-world security challenges increasingly demand the development of more OA-based attack strategies.

2.5.1 OG Attacks

One of the first OG attacks was based on automatic test pattern generation (ATPG), a technique initially designed to detect hardware faults. This method was later adapted into the ATPG-based attack, which leverages ATPG techniques to expose secret key values in LL designs. The ATPG-based attack treats locked gates as faults and generates input patterns that reveal key dependencies. By effectively avoiding brute-force techniques, this method exposes weaknesses in locked designs that depend on simple signal dependencies [15]

The SAT-based attack remains one of the most important OG attacks, as it systematically refines the key space using DIPs [20, 76, 84]. The attack begins by duplicating the locked netlist, using the same input signals but with different key inputs. A miter circuit is then introduced, which compares the outputs of the two locked instances and generates a conjunctive normal form (CNF) representation of this circuit. The SAT solver is then tasked with finding a DIP that produces different outputs in the two locked versions. Once a DIP is identified, it is applied to the oracle to determine the correct output. This process ensures that at least one of the found keys is incorrect, allowing the solver to eliminate invalid key values progressively. The attack then adds the CNF constraints corresponding to the observed DIP to prevent the rediscovery of the same pattern and iterates the process. This continues until the SAT solver fails to find a new DIP, at this point, the last remaining key is guaranteed to be the correct secret key. This systematic refinement enables the SAT-based attack to break many traditional LL schemes, making it a formidable threat to hardware security.

An extension of the SAT-based approach, the Double DIP attack, improves attack efficiency by leveraging two sets of DIPs to accelerate key recovery. By reducing the number of iterations required to reach the correct key, Double DIP remains particularly effective against single-layer LL methods [75].

The AppSAT (approximate SAT) attack was introduced to mitigate high-complexity

SAT computations by allowing an approximation mechanism. Instead of fully solving the key recovery problem, AppSAT finds an approximate key that produces a functionally equivalent circuit, even if not an exact match to the original design, using a small number of queries. This approximation is useful against complex LL schemes, as it finds an approximate correct key that makes the circuit functional for the applied queries [23].

Expanding on AppSAT, the AppSAT guided removal (AGR) attack targets CLL techniques, particularly those incorporating point functions such as Anti-SAT. AGR combines AppSAT's approximation approach with structural analysis, refining key recovery through logical inference. This method applies AppSAT to deduce an approximate key and then performs structural analysis to pinpoint the exact key bits associated with the point function-based LL techniques [25].

The Fa-SAT (fault-aided SAT) attack targets CLL designs using a fault-based approach. It introduces faults in the circuit to guide the SAT solver more efficiently. By inducing controlled faults, attackers create additional DIPs that enhance the ability of the SAT attack to converge on the correct key, allowing secret key extraction even in advanced CLLs such as BLE and RLL combined with Anti-SAT. This attack is particularly effective against locking schemes that rely heavily on input-output behavior, as the fault responses provide the SAT solver with rich data, expediting the solution [74].

The query attack applies a set of carefully selected queries to determine the values of key bits of the secret key using the concept of proof by contradiction. Query attack is often combined with structural analysis techniques to enhance efficiency and are exceptionally successful against LL methods that rely solely on input-output behavior for security [30, 35].

The quantified Boolean formula (QBF) solver generalizes the SAT attack by incorporating existential ($\exists$) and universal ($\forall$) quantifiers, allowing adversaries to construct complex logical formulations that expose key dependencies. The QBF-based attack targets the locking/restore unit in SFLT and DFLT in two steps: it first constructs a QBF problem by combining the CNF formulas of individual gates and then generates two separate QBF problems, one where the output is *0* and another where it is *1* for all possible input combinations. The secret key can be extracted by solving these problems using a QBF solver. If a solution exists for either QBF problem, the key for the locked circuit is discovered. This dual-problem formulation ensures that all output possibilities are evaluated. It is effective against designs like SFLT, whose locking unit produces a constant value under the secret key for all inputs [26].

Functional analysis attacks exploit input-output behavior to deduce the correct key in LL schemes by systematically analyzing functional dependencies. This approach bypasses structural defenses, revealing that even obfuscated circuits may be vulnerable to attacks based solely on observable functionality, allowing attackers to break the lock by simplifying the circuit model through functional analysis [68].

The Valkyrie tool is a comprehensive vulnerability assessment and attack framework created to test and break LL techniques that claim to be secure. Valkyrie probes the locked circuit's logic and signal paths by combining structural and functional analysis to identify areas where key bits impact overall functionality, simulating potential attack scenarios to reveal weaknesses. It examines signal flow traces to understand how key values influence the circuit, effectively isolating critical regions susceptible to attack.

This approach allows Valkyrie to uncover the correct key by focusing on critical points within the circuit. Valkyrie is a KA attack where the adversary knows the techniques employed in advance. Effectiveness across fifteen different techniques demonstrates that even theoretically secure LL methods may exhibit vulnerabilities in real-world applications, underscoring the need for continued innovation in hardware security [69].

2.5.2 OL Attacks

The synthesis-based constant propagation attack for security evaluation (SCOPE) attack identifies constant signal propagation within circuits to expose key values by analyzing the impact of setting a constant logic value $0/1$ to a key input on area, delay, power, fan-in, fan-out, and other critical design metrics of the locked netlist. Adversaries can deduce key-related patterns, significantly narrowing down possible key values [78].

The topology-guided attack (TGA) is a structural analysis attack and leverages the circuit's structural layout to infer key dependencies. TGA deduces key bits relationships by mapping the circuit's topology and identifying sensitive nodes, enabling attackers to break LL schemes based solely on circuit structure. TGA is quite effective against LL techniques that do not mask structural dependencies, as it exploits the inherent connections between key bits and critical nodes [90].

Structural analysis techniques serve as the foundation for many OL attack strategies. These approaches identify weaknesses in LL designs by detecting irregularities in circuit connectivity, enabling attackers to extract key dependencies without an oracle [91].

Advances in ML-based attacks have introduced graph neural networks (GNNs) as a powerful tool for predicting key bits. Attackers can identify patterns in new, unseen circuits by training ML models on known locking schemes. This approach provides an OL attack to predict key bits, which is especially effective in modern circuits with complex locking patterns, as GNNs excel at detecting relational dependencies [92].

Resynthesis-based attacks exploit the synthesis process to reveal key dependencies. These attacks re-synthesize the locked circuit using various design constraints, generating functionally equivalent but structurally different versions of the netlist. By analyzing structural differences across iterations, adversaries can systematically expose key dependencies [34].

As LL advances, attackers quickly adapt, developing sophisticated methods to bypass protections. SAT-based attacks remain central among OG approaches, with variants like Double DIP, AppSAT, and Fa-SAT bringing notable gains in effectiveness against complex LL designs. Meanwhile, the rise of synthesis-based and ML-driven techniques has expanded the range of OL attacks, enabling intrusions that sidestep the need for output data. This continuous evolution in attack strategies emphasizes the need for LL defenses to keep pace, ensuring robust protection against these persistent threats.

2.6 Benchmark Circuits and Metrics

The benchmark circuits analyzed in this thesis include combinational designs from the ISCAS'85 [93] and ITC'99 [94] suites, which are widely used to evaluate the resilience of LL schemes against both OG and OL attacks. Additionally, circuits from the CSAW'19 [95] benchmark set were used to assess CLL techniques, as they integrate

two LL strategies simultaneously. FIR filter circuits were also included to evaluate the impact of hybrid protection in more application-driven contexts.

Throughout the thesis, a wide range of LL techniques are investigated. These include RLL [14] as a baseline, as well as more sophisticated approaches such as Anti SAT [27], Anti SAT DTL [96], CASLock [63], SARLock [16], SKG Lock [65], TTLock [28], and SFLL [64]. Combinations of RLL with other PSLL schemes were also employed to construct CLL circuits. Although all these techniques are explored in detail in later chapters, in this subsection, we focus on evaluating the impact of five LL techniques, namely RLL, Anti-SAT, TTLock, RLL+Anti-SAT, and RLL+TTLock, on the hardware complexity in terms of area, power, and delay and on the resiliency to the existing attacks. These techniques were applied to five circuits from the ISCAS'85 suite and synthesized using the Cadence Genus logic synthesis tool with a commercial 65*nm* standard cell library. The number of key inputs varied between 32 and 64 bits, depending on the circuit size, to balance security and hardware complexity on a single LL technique. In CLL, we use the same number of key inputs in the RLL and PSLL techniques.

Table 1 presents the characteristics of the original ISCAS'85 circuits, where #in and #out denote the number of inputs and outputs, respectively, *area* is the total area in μm^2, *power* is the total power dissipation in mW, and *delay* is the critical path delay in ps. The corresponding implementation overheads introduced by each LL scheme—measured in terms of area, power, and delay—are summarized in Table 2.

Table 1: Details of ISCAS'85 circuits.

Circuit	Original Netlist				
	#in	#out	area	power	delay
c2670	157	64	1046	3.36	1264
c3540	50	22	1518	6.58	1977
c5315	178	123	2460	9.9	1864
c6288	32	32	3133	8.48	4621
c7552	206	105	2702	1.32	1663

Table 2: Overhead in area, power, and delay for each LL technique, and run-time of attacks.

Technique	**Area (%)**	**Power (%)**	**Delay (%)**	**Attack**	**Run-time (s)**
RLL	19.1 to 31.2	58.2 to 78.0	6.3 to 17.6	SAT [20]	0.29 to 2.69
Anti-SAT	8.2 to 31.9	11.5 to 35.1	2.3 to 10.1	KRATT [26]	0.26 to 1.13
TTLock	5.7 to 30.0	4.4 to 31.8	1.3 to 6.9	KRATT [26]	72.38 to 333.20
RLL+Anti-SAT	20.3 to 42.9	35.8 to 47.3	3.6 to 13.2	RESAA [30]	540.43 to 929.78
RLL+TTLock	24.7 to 54.6	34.8 to 47.7	6.4 to 15.3	RESAA [30]	418.11 to 662.32

To illustrate the impact of these techniques, RLL incurs area overheads ranging from 19.1% to 31.2%, with corresponding power increases from 58.2% to 78.0% and delay overheads from 6.3% to 17.6%. Anti-SAT exhibits lower power and delay overheads than RLL, while its impact on area varies from 8.2% to 31.9%. TTLock presents an overhead ranging from 5.7% to 30.0% in area, between 4.4% and 31.8% in power, and between 1.3% and 6.9% in delay. When RLL is combined with Anti-SAT or TTLock in CLL configurations, overheads increase significantly. RLL combined with Anti-SAT leads to area, power, and delay overheads of up to 42.9%, 47.3%, and 13.2%, respectively, while RLL combined with TTLock has the largest impact on the hardware complexity.

Despite the additional complexity, all these locked circuits remain vulnerable to known attacks, as shown in Table 2. RLL is broken by SAT-based attacks [20] within 0.29 to 2.69 seconds. Anti-SAT and TTLock are defeated using the structural analysis attack KRATT [26] in 0.26 to 1.13 seconds and 72.38 to 333.20 seconds, respectively. CLL schemes, such as RLL combined with Anti-SAT and RLL combined with TTLock, are broken by the RESAA attack [30], which increases attack time to a range between 418.11 and 929.78 seconds, still within practical limits for an adversary. These results and evaluations serve as a baseline for the subsequent experimental discussions, which further analyze attack strategies and the complexity and security tradeoff.

3 Discussion

This chapter provides an overview of the three papers on which this thesis is based, showing the main findings, defense methods, and proposed attacks in the field of hardware security for digital circuits. These works demonstrate the growing sophistication of defense and attack mechanisms focused on hybrid techniques and CLL methods that could integrate resilience against attacks while maintaining performance.

3.1 Hybrid Protection of Digital FIR Filters

FIR filters are fundamental blocks in digital signal processing (DSP) due to their stability and predictable phase characteristics. Unlike generic digital circuits, where the IP is often linked to architecture and functional logic, the actual value in FIR filters lies in their coefficients. These coefficients define the filter's frequency response and, consequently, its behavior in applications such as communications, image processing, and biomedical signal analysis. Without adequate protection, adversaries can extract these coefficients through reverse engineering, effectively replicating the IP while bypassing the cost and effort associated with its design [32]. However, only a limited number of techniques have been proposed to obfuscate DSP circuits, particularly digital filters. Existing methods have significant vulnerabilities when faced with more advanced attacks [32, 97], necessitating more robust security mechanisms.

This subsection provides an overview of the research published in *IEEE TVLSI* 2023 as given in Appendix A. This study introduces a novel **query attack** that exploits weaknesses in existing protection schemes. An adversary can extract values of key bits by performing targeted queries, ultimately reconstructing the filter's coefficients. This method highlights the limitations of conventional hardware obfuscation and demonstrates that existing LL techniques alone are insufficient to secure FIR filter designs.

As a countermeasure, a **hybrid protection mechanism** is proposed, integrating hardware obfuscation with LL to safeguard both the filter's coefficients and functionality. This approach enhances obfuscation by leveraging decoy constants and a key-based technique, making it significantly harder for adversaries to extract the correct coefficients or replicate the filter's behavior.

The security and efficiency of locked architectures depend on the underlying multiplication method used. Constant multiplication generally occurs in many DSP applications, particularly in FIR filters, where coefficients are multiplied by input values. Three architectures are commonly used for such operations depending on the filter architecture, i.e., transposed or direct form: constant array vector multiplication (CAVM), multiple constant multiplication (MCM), and time-multiplexed constant multiplication (TMCM). CAVM performs the multiplication of a $1 \times n$ constant array by an $n \times 1$ input vector, generating a single output through the summation of all partial products. MCM, on the other hand, performs the multiplication of a set of *n constants* by a single input variable. Finally, TMCM allows for the time-multiplexed selection of one constant from a set of *n constants*, multiplying it by an input variable at each time step. These architectures enable efficient implementation of FIR filters, particularly in hardware-constrained environments where reducing multiplications through addition and shift operations is essential.

Table 3: Results of obfuscated and protected multiplier blocks. This table is the same as the Table V in Publication I.

Block	Architecture	Technique	Synthesis			Attacks							
						SAT	ATPG	AppSAT	DoubleDIP	Query		SCOPE	
			area	delay	power	time	time	time	time	prv	time	cdk/dk	time
CAVM	CAVM-MUL	Decoy [32]	15435	4616	4641	155143	OoT	OoT	OoT	32	9893	20/32	8
		Proposed Hybrid	15710	4611	4757	OoT	OoT	OoT	OoT	0	OoT	1/1	13
	CAVM-SA	Decoy [32]	15465	4611	4475	36083	4539	OoT	OoT	32	9944	20/32	8
		Proposed Hybrid	15704	4715	4497	OoT	OoT	OoT	OoT	0	29328	1/1	12
	CAVM-CRK	Constant [58]	18737	3982	4756	110	1446	OoT	OoT	32	897	21/27	11
		Proposed Hybrid	18976	4265	4809	OoT	OoT	OoT	OoT	0	1937	2/3	16
MCM	MCM-MUL	Decoy [32]	10949	3102	2839	106	OoT	324	243	32	176	27/32	7
		Proposed Hybrid	11173	3031	2897	OoT	OoT	OoT	OoT	0	197	1/1	11
	MCM-SA	Decoy [32]	10493	3112	2493	119	OoT	342	254	32	152	27/32	7
		Proposed Hybrid	10705	3159	2495	OoT	OoT	OoT	OoT	0	115	1/1	11
	MCM-CRK	Constant [58]	12799	2772	2412	159	OoT	415	300	32	459	18/32	7
		Proposed Hybrid	13038	2970	2456	OoT	OoT	OoT	OoT	0	759	1/1	11
TMCM	TMCM-MUL	Decoy [32]	1545	3517	610	241	6783	378	496	32	7	21/32	4
		Proposed Hybrid	1794	4278	639	OoT	OoT	OoT	OoT	0	17	2/3	2
	TMCM-SA	Decoy [32]	2043	4452	1037	738	963	935	OoT	32	37	17/32	4
		Proposed Hybrid	2245	4536	1065	OoT	OoT	OoT	OoT	0	42	1/2	3
	TMCM-CRK	Constant [58]	1566	2997	623	1035	15571	1032	OoT	32	29	20/32	4
		Proposed Hybrid	1776	3655	642	OoT	OoT	OoT	OoT	0	48	1/1	2

The experimental results are presented in Tables 3 and 4. These tables compare the performance and security resilience of various locked multiplier blocks, showing the superiority of the hybrid approach. These tables present the evaluation metrics and attack results. Note that *area*, *delay*, and *power* represent the total area in μm^2, critical path delay in ps, and total power dissipation in μW, respectively. It includes results from SAT, ATPG, AppSAT, DoubleDIP, and SCOPE attacks, where *cdk* and *dk* denote correctly and total deciphered key bits for SCOPE, respectively. All attacks had a 2-day time limit, and designs are highlighted where the secret key remained undiscovered.

Different architectures implementing the multiplication block of the FIR filter with varying hardware complexity were tested to validate the proposed protection mechanism. MUL-based represents a conventional approach where multiplications are performed directly using hardware multipliers, as in the CAVM-MUL, MCM-MUL, and TMCM-MUL implementations. SA architecture eliminates the need for multipliers by implementing constant multiplications using only addition, subtraction, and shift operations, as seen in CAVM-SA, MCM-SA, and TMCM-SA. Constant replacement with key bits (CRK) enhances security by replacing selected constant values with key bits, making it harder for an adversary to infer the correct functionality, as demonstrated in CAVM-CRK, MCM-CRK, and TMCM-CRK. Each of these architectures was locked and evaluated using different LL techniques, providing a robust dataset for comparing the effectiveness of hybrid protection against traditional LL techniques.

Table 3 compares the performance of locked multiplier blocks under different obfuscation techniques. The hybrid protection method consistently outperforms existing obfuscation techniques, particularly in its resilience to the query attack. Although existing obfuscation techniques may be broken within a specific timeframe, the hybrid method provides a more robust defense, enduring prolonged attack attempts without compromising the secret key.

As seen in Table 4, hybrid protection generally demonstrates greater resilience against the query attack than traditional LL techniques, though some locked designs remain secure. While LL can be compromised within certain time limits, the hybrid technique offers a more robust defense, resisting extended attacks without leaking critical

Table 4: Results of locked multiplier blocks. This table is the same as the Table VI in Publication I.

Block	Logic Locking	Synthesis			Attacks							
					SAT	ATPG	AppSAT	DoubleDIP	Query		SCOPE	
		area	delay	power	time	time	time	time	prv	time	cdk/dk	time
CAVM	RLL	16411	3385	4183	171	OoT	205	2609	32	254	0/0	9
	RLL+AntiSAT	16492	3416	4127	OoT	OoT	OoT	OoT	16	364	0/0	14
	RLL+CASLock	16473	3502	4110	OoT	OoT	OoT	OoT	16	443	0/0	13
	RLL+SARLock	16512	3572	4191	OoT	OoT	48855	OoT	32	441	8/8	13
	RLL+SFLL	16506	3473	4205	339	829	39664	OoT	32	436	0/0	14
	RLL+SKGLock	16559	3894	4308	OoT	OoT	OoT	OoT	19	513	5/6	14
MCM	RLL	8090	2398	2039	86	122	293	371	32	89	0/0	6
	RLL+AntiSAT	8244	2434	2065	OoT	OoT	OoT	OoT	16	249	0/0	9
	RLL+CASLock	8121	2404	2024	OoT	OoT	1054	OoT	16	164	0/0	9
	RLL+SARLock	8209	2467	2041	OoT	OoT	45013	OoT	32	228	10/10	9
	RLL+SFLL	8166	2452	2019	576	7870	2626	OoT	32	243	0/0	9
	RLL+SKGLock	8252	2464	2056	OoT	OoT	OoT	OoT	19	160	5/5	9
TMCM	RLL	1587	3712	646	8	39	57	77	32	15	0/0	2
	RLL+AntiSAT	1659	3545	632	OoT	OoT	OoT	OoT	13	28	0/0	2
	RLL+CASLock	1653	3664	628	OoT	OoT	OoT	OoT	15	20	0/0	2
	RLL+SARLock	1659	3756	658	OoT	OoT	2662	OoT	31	36	6/6	3
	RLL+SFLL	1644	3800	653	1095	1089	5363	OoT	32	36	0/0	2
	RLL+SKGLock	1694	3739	676	OoT	OoT	OoT	OoT	18	34	10/10	2

information. This enhanced resilience, however, incurs a modest trade-off in synthesis metrics: hybrid protection typically increases the total area and power consumption by around 1–5%. The delay overhead is usually modest but varies significantly with the underlying architecture, ranging from negligible to approximately 21% in extreme cases, such as the TMCM multiplier block. Nonetheless, this trade-off remains reasonable, given the substantial security improvements.

In conclusion, the hybrid protection technique significantly improves the security of FIR filters against advanced attack methods such as query attacks. While not entirely immune to future attack evolutions, this approach considerably strengthens the defense of critical components in digital signal processing systems, ensuring their integrity against unauthorized access.

3.2 Resynthesis-based Attacks Against Logic Locking

Attacks using the EDA tools engine have emerged as a powerful technique to break LL schemes by exploiting changes to the netlist or constant propagation to uncover secret keys [34, 98]. Synthesis tools translate a high-level behavioral description of a circuit into standard cells from a target technology library. This process involves mapping abstract logic functions to specific components available in the library, ensuring that the generated design meets constraints such as area, power, and delay.

This subsection presents a novel **resynthesis-based attack** that enhances existing methodologies under the OG and OL threat models. The attack leverages structural diversity generated by multiple synthesis iterations with varying parameters, enabling key recovery by exposing inconsistencies across locked netlists. This work demonstrates that resynthesis inadvertently weakens the security of LL designs, making them more susceptible to previously ineffective attacks.

The proposed approach and its implications for LL security have been published in the *ISQED Conference*, 2023. The complete experimental results and further details are provided in the Appendix B. The main contribution of this work is to demonstrate

that resynthesis not only increases the chances of key recovery but also weakens the locked circuits against attacks that would otherwise fail.

The resynthesis process systematically generates a large set of unique, yet functionally equivalent, locked circuits by varying synthesis parameters such as timing and logic optimization effort. These variations introduce subtle differences in the circuit's structure, which attackers exploit to recover key information. This resynthesis-based attack is applicable to a wide range of LL techniques, including Anti-SAT [27], CASLock [63], SFLL [64], and SKG-Lock [65]. By creating multiple unique versions of the locked netlists, the attack uncovers significantly more key bits than traditional attacks on the original netlists. Furthermore, by combining OL and OG attacks, the proposed technique strengthens key recovery accuracy, making it a powerful tool against single LL and CLL schemes.

Figures 5 and 6 highlight the structural differences between two synthesized netlists generated with different delay constraints. These differences, though subtle, are critical for understanding why resynthesis amplifies the success of attacks like SCOPE. The figure shows how small changes in synthesis parameters can drastically alter the circuit's gate count and logic depth. Such structural diversity is critical to exposing vulnerabilities in the locked design, as it allows the attack to leverage variations in the logic structure to recover more key bits.

Table 5 shows the results of OL attacks on locked ISCAS'85 circuits. The table compares the success of the SCOPE attack on the original locked netlist versus the resynthesized netlists. It is evident that while the SCOPE attack is not entirely successful on the original locked netlist, resynthesis enables the recovery of a significant number of key bits. For instance, circuits like *c2670*, when locked with SKG-Lock, have 100% of their key bits recovered after resynthesis, compared to partial or unsuccessful recovery without resynthesis. The secret key is also found in other benchmarks when the undetermined key inputs are assigned a constant logic value of *0* or *1*. This underscores the effectiveness of resynthesis in compromising LL techniques previously considered resilient to attacks like SCOPE. The results demonstrate that even SKG-Lock, an SAT-resilient locking technique, is susceptible to resynthesis, revealing the hidden key bits.

Table 6 extends the analysis to CSAW'19 circuits, which use CLL combining RLL and SFLL-rem [95]. The table compares the results of the proposed resynthesis-based attack with traditional OL attacks. Full key recovery is achieved for RLL, even under complex circuits. While SFLL-rem presents a more difficult challenge, the resynthesis-based attack still outperforms other OL methods, demonstrating a large number of recovered key bits, recovering up to approximately 44% of the previously undiscoverable key bits. This result underscores the robustness of the resynthesis approach in handling both OL and OG threat models. The ability to break RLL and still recover partial key bits under more complex schemes like SFLL-rem highlights the broad applicability of the method across different LL techniques.

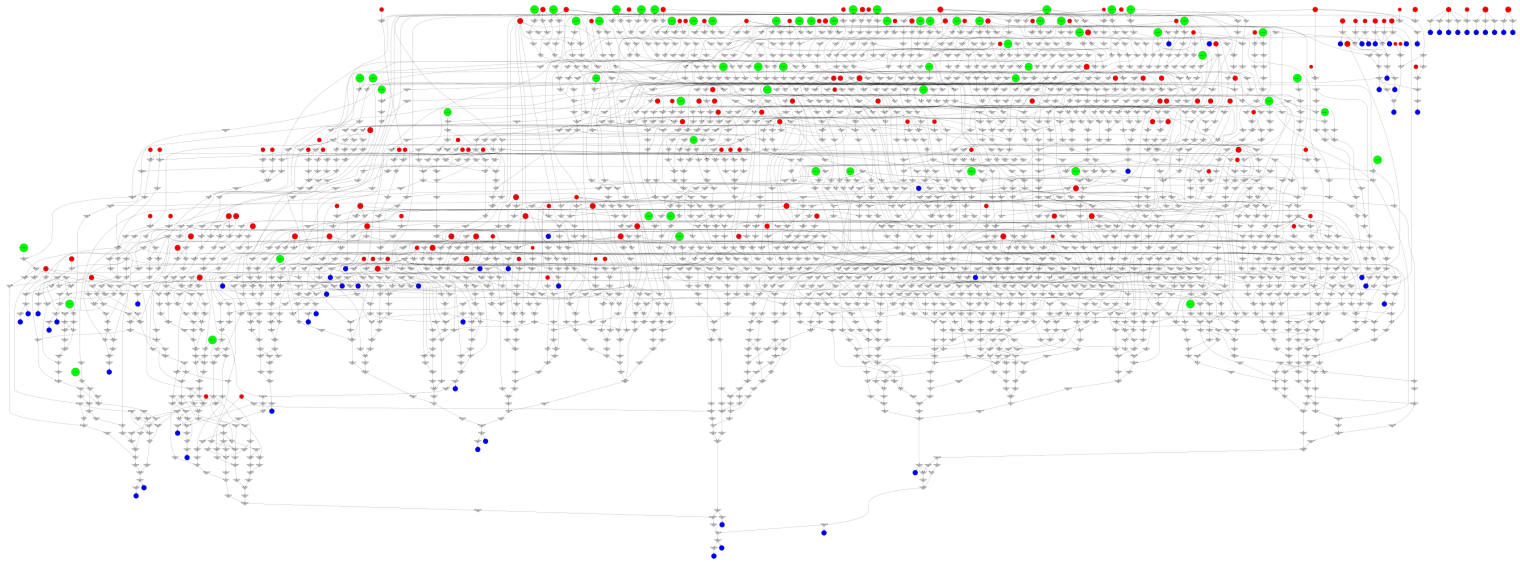


Figure 5: Graph of the netlist resynthesized when the delay constraint is 990 ps. This figure is reproduced from Figure 5(a) in Publication II.

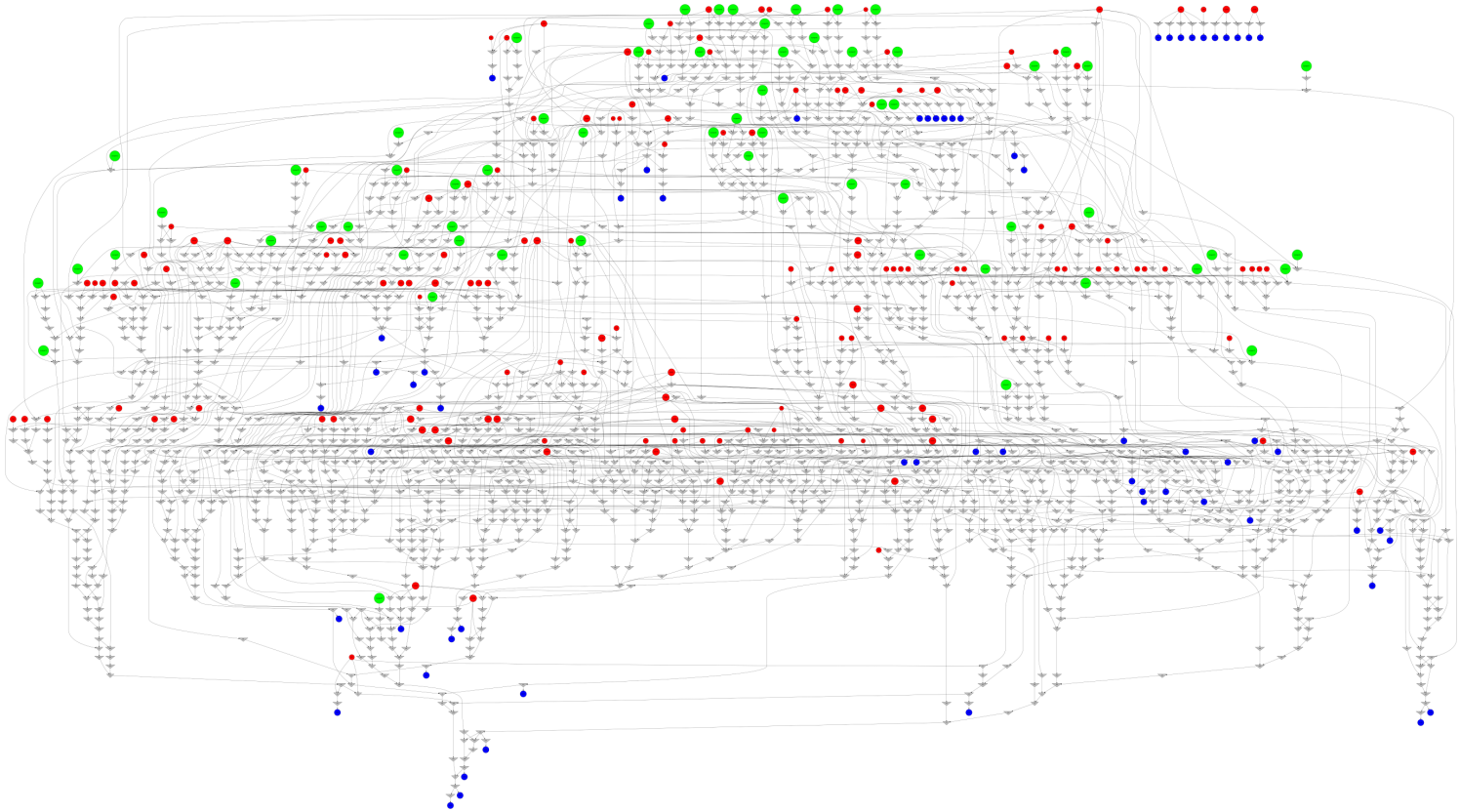


Figure 6: Graph of the netlist resynthesized when the delay constraint is 496 ps. This figure is reproduced from Figure 5(b) in Publication II.

Table 5: Results of OL Attacks on the locked ISCAS'85 Circuits. This table is the same as the Table III in Publication II.

Circuit	Anti-SAT			CASLock			SFLL			SKG-Lock		
	SCOPE	Resynthesis		SCOPE	Resynthesis		SCOPE	Resynthesis		SCOPE	Resynthesis	
	cdk/dk	time	cdk/dk time	cdk/dk	time	cdk/dk time	cdk/dk	time	cdk/dk time	cdk/dk	time	cdk/dk time
c2670	0/0	4s	37/64 34m18s	0/0	4s	35/64 33m47s	0/0	4s	34/64 37m32s	32/32	4s	64/64 44m37s
c3540	0/0	3s	17/32 21m27s	0/0	3s	17/32 18m12s	0/0	2s	19/32 21m29s	17/17	2s	32/32 24m30s
c5315	0/0	5s	38/64 42m34s	0/0	5s	30/64 43m54s	0/0	5s	33/64 46m23s	32/32	5s	62/62 52m06s
c6288	0/0	3s	18/32 29m08s	0/0	3s	16/32 27m18s	0/0	3s	16/31 33m19s	16/16	3s	31/31 34m24s
c7552	0/0	6s	38/64 45m31s	0/0	6s	47/64 49m13s	0/0	6s	38/63 52m26s	32/32	6s	61/61 56m45s

Table 6: Results of attacks on the locked CSAW'19 Circuits. This table is the same as the Table VII in Publication II.

Approach	Attack Scenario	Circuit							
		small (40+40)		medium (60+60)		large (80+80)		bonus (128+128)	
		RLL	SFLL-rem	RLL	SFLL-rem	RLL	SFLL-rem	RLL	SFLL-rem
Key sensitization [15]	OG	40/40	—	60/60	—	80/80	—	—	—
Hamming distance-based attack [95]	OG	30/30	—	50/50	—	72/72	—	—	—
Automated analysis + SAT [95]	OG	11/18	—	31/50	—	10/34	—	—	—
Sub-circuit SAT [95]	OG	17/17	—	29/29	—	—	—	—	—
Redundancy-based [99]	OL	28/28	4/12	35/35	23/28	45/45	0/51	66/66	8/27
Bit-flipping attack [76]	OG	40/40	—	60/60	—	80/80	—	—	—
Topology guided attack [90]	OL	15/32	—	19/50	—	36/73	—	75/108	—
Resynthesis-based attack	OG	40/40	20/39	60/60	29/60	80/80	35/79	128/128	55/124

These results demonstrate that a resynthesis-based attack is critical in exposing weaknesses in logic-locked circuits. The proposed attack significantly improves the effectiveness of OL attacks by exploiting the structural variations introduced during resynthesis. Even for circuits locked with SAT-resistant techniques, resynthesis enables attackers to recover key bits that would have otherwise remained hidden. Furthermore, by combining OL and OG methods, this approach achieves higher key recovery accuracy, making it a robust tool against advanced locking schemes like SFLL-rem.

This study highlights the importance of resynthesis in hardware security research. By systematically altering the structure of locked circuits, we expose vulnerabilities that are not easily detectable with traditional attacks. The results achieved under both OL and OG threat models reinforce the effectiveness of the proposed attack.

3.3 RESAA: A Removal and Structural Analysis Attack Against Compound Logic Locking

CLL schemes have been introduced to integrate a multi-layered LL strategy to protect ICs against advanced attack methodologies. By combining multiple LL techniques, CLL aims to leverage the strengths of different approaches while mitigating their weaknesses. However, recent studies have identified structural vulnerabilities that adversaries can exploit [25, 74]. These vulnerabilities arise from how LL techniques interact within the CLL structure, potentially exposing CGs or enabling key differentiation through attack strategies that partition the locked design.

This subsection introduces **RESAA**, a novel attack framework that leverages both removal and structural analysis techniques to break CLL. RESAA was developed as part of this thesis and has been published in *IEEE TVLSI 2025*. For further details, the full version of this work, including extended experimental results, is provided in Appendix C. The following sections detail the attack methodology, experimental validation, and implications for future LL defenses.

RESAA exposes vulnerabilities in CLL designs by leveraging a combination of SAT-

based attacks, structural analysis, and query-based attacks. The core of RESAA lies in classifying the keys of a CLL-locked circuit into two categories: RLL and PSLL. By leveraging the CG, where all key inputs converge before reaching the primary output (PO), RESAA partitions the design into two distinct netlists. This partitioning not only simplifies the attack but also reduces the complexity of applying traditional attack techniques, such as SAT-based or SCOPE attacks.

Figure 7 provides the overview of the RESAA, illustrating the entire workflow from pre-processing the CLL-locked circuit to uncovering the secret key. The left side of the figure demonstrates the initial pre-processing steps, where the original circuit is locked using CLL techniques. It begins by applying RLL to the original circuit, adding XOR/XNOR gates driven by key inputs, which offer minimal overhead in terms of area and power but contribute to an increased delay, as described in the paper. Then, CLL techniques such as Anti-SAT [27], Anti-SAT-DTL [96], CASLock [63], SARLock [16], or TTLock [28] are applied. These methods increase resistance to attacks but also introduce hardware complexity and overhead. This locked circuit is then translated into a mapped Verilog netlist by the synthesis process, which forms the basis for the subsequent steps.

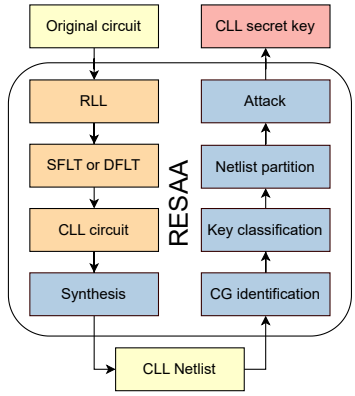
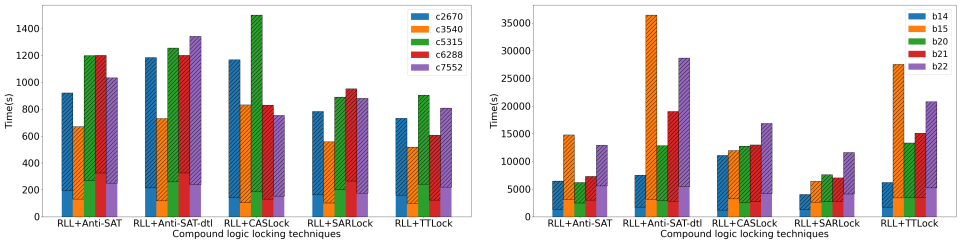


Figure 7: Overview of the RESAA framework. This figure is reproduced from Figure 4 in Publication III.

The classification process starts by identifying the CG and a key classification between RLL and PSLL. Subsequently, the netlist is divided into two distinct partitions. Each partition is processed separately, with the RLL-protected portion subjected to SAT-based and/or query-based attacks. In contrast, the PSLL-protected portion is analyzed using advanced techniques like QBF-based attacks. This partitioning strategy improves the attack's efficiency and allows RESAA to target specific vulnerabilities in each part of the circuit. By isolating one partition with only RLL keys and another with only PSLL keys and attacking the distinct portions of the CLL-protected design, the RESAA framework effectively exposes the secret keys embedded within, overcoming many of the defenses presented by the CLL mechanisms.

The experiments were conducted on ten benchmark circuits from the ISCAS'85 [93]



(a) ISCAS'85 benchmark *(b) ITC'99 benchmark*

Figure 8: Classification and execution times (seconds) for attacking ISCAS'85 and ITC'99 benchmarks in the CLL scheme. Bottom: Classification and partition time. Hatched: Attack time. Combined: Total execution time. This figure is reproduced from Figure 9 in Publication III.

and ITC'99 [94] suites. The locking mechanisms were implemented using a combination of the Neos tool [100], Python, and Perl scripts.

Table 7: Details of existing attacks in ISCAS'85 and ITC'99 circuits locked using a CLL scheme. This table is the same as the Table IV in Publication III.

Circuit	Locked Netlist																									
	RLL+Anti-SAT					RLL+Anti-SAT-DTL					RLL+CASLock					RLL+SARLock					RLL+TTLock				
	sat	appsat	dp	qatt		sat	appsat	dp	qatt		sat	appsat	dp	qatt		sat	appsat	dp	qatt		sat	appsat	dp	qatt	
	time	time	time	prv	time	time	time	time	prv	time	time	time	time	prv	time	time	time	time	prv	time	time	time	time	prv	time
c2670	OoT	998	OoT	52	48	OoT	170	OoT	50	45	OoT	238	OoT	50	25.4	OoT	114	OoT	55	59	OoT	1055	OoT	48	48
c3540	OoT	766	OoT	30	20	OoT	203	OoT	32	19	OoT	66	OoT	30	8	OoT	91	5	31	22	48656	4499	2	31	19
c5315	OoT	239	OoT	62	50	OoT	9949	OoT	62	52	OoT	131	OoT	62	42	OoT	64	OoT	62	72	OoT	2768	OoT	62	60
c6288	OoT	OoT	OoT	32	77	OoT	58470	OoT	32	69	OoT	3936	OoT	32	90	OoT	935	OoT	32	119	OoT	6270	OoT	32	93
c7552	OoT	172	OoT	55	96	OoT	543	OoT	55	105	OoT	279	OoT	55	59	OoT	247	OoT	55	108	OoT	51	OoT	55	79
b14_C	OoT	OoT	OoT	109	1822	OoT	OoT	OoT	112	1383	OoT	OoT	OoT	106	2082	OoT	OoT	OoT	109	1335	OoT	OoT	OoT	111	1783
b15_C	OoT	OoT	OoT	98	466	OoT	OoT	OoT	97	635	OoT	OoT	OoT	96	836	OoT	OoT	OoT	80	513	OoT	OoT	OoT	87	671
b20_C	OoT	OoT	OoT	115	1562	OoT	OoT	OoT	115	1647	OoT	OoT	OoT	109	2645	OoT	OoT	OoT	115	1645	OoT	OoT	OoT	113	2187
b21_C	OoT	OoT	OoT	113	1119	OoT	OoT	OoT	114	1337	OoT	OoT	OoT	113	1935	OoT	OoT	OoT	110	1385	OoT	OoT	OoT	109	1738
b22_C	OoT	OoT	OoT	113	1034	OoT	OoT	OoT	113	1214	OoT	OoT	OoT	108	1489	OoT	OoT	OoT	105	938	OoT	OoT	OoT	111	1342

Table 7 shows the runtime of attacks required to find the secret key, where out-of-time (OoT) indicates that no solution could be found within the allowed 48-hour time limit. As observed from Table 7, the SAT-based (sat) and DoubleDIP (dp) attacks exhibit low efficiency in deciphering key inputs, as expected. They only found a solution for one single RLL+TTLock case in the *c5315* circuit. The AppSAT attack showed promising results for small circuits but demanded significant execution time compared to other attacks. Although AppSAT (appsat) demonstrated nearly 100% efficiency in breaking ISCAS'85 circuits, it could not solve certain more complex cases, such as the *c6288* circuit locked with RLL+Anti-SAT. Lastly, the query attack from [101], qatt, displayed varying execution times and degrees of success in deciphering key inputs, as shown in Table 7. The average proportion of proven values of key inputs prv is approximately 41%, with execution times ranging from 8 to 2645 seconds.

Next, Figure 8 presents the classification and execution times under the OG threat model. In this context, "classification time" refers to the duration required for categorizing the LL technique utilized in the CLL design, depicted in the lower section of the graph. Conversely, "attack time" is indicated by the hatched portion of the graph. In contrast, "execution time" represents the total time, including both classification/partition time and the subsequent attack on each circuit, as depicted by both sections in the graph.

Table 8 compares the results of RESAA under the OL threat model with the SCOPE attack. RESAA successfully recovers a substeantial portion of the key bits across

multiple benchmarks. For example, the attack reveals 37 out of 64 key bits for the *c2670* circuit locked with RLL + Anti-SAT, and a similar performance is observed across other circuits. These results demonstrate RESAA's ability to uncover a large portion of the key even when no functional IC (oracle) is available for validation. In contrast, traditional attacks, such as SCOPE, fail to reveal any key bits, further underscoring the effectiveness of RESAA's partitioning strategy. This attack methodology is particularly potent when applied to complex CLL designs, where a direct attack on the whole circuit would otherwise be computationally infeasible.

Table 8: Results of OL Attacks on the locked ISCAS'85 and ITC'99 circuits. This table is the same as the Table V in Publication III.

Circuit	RLL+Anti-SAT				RLL+Anti-SAT-DTL				RLL+CASLock				RLL+SARLock				RLL+TTLock			
	SCOPE		RESAA		SCOPE		RESAA		SCOPE		RESAA		SCOPE		RESAA		SCOPE		RESAA	
	cdk/dk	time	cdk/dk	time	cdk/dk	time	cdk/dk	time	cdk/dk	time	cdk/dk	time	cdk/dk	time	cdk/dk	time	cdk/dk	time	cdk/dk	time
c2670	0/0	14	73/104	4	0/0	13	80/104	4	0/0	9	84/104	4	32/64	8	97/105	4	16/106	10	68/105	9
c3540	0/0	6	37/43	2	0/0	6	40/64	2	0/0	4	40/45	2	15/32	4	41/43	2	8/42	7	32/45	4
c5315	0/0	15	77/107	5	0/0	16	78/107	5	0/0	12	81/107	5	32/64	10	100/107	5	30/107	12	69/109	11
c6288	0/0	7	41/56	3	0/0	7	42/56	2	0/0	5	42/56	3	17/32	5	49/56	3	4/53	6	40/56	5
c7552	0/0	17	78/108	5	0/0	17	79/105	5	0/0	8	81/108	4	40/64	11	98/107	6	28/107	13	82/109	12
b14_C	0/0	91	160/210	67	0/0	91	168/215	69	0/0	67	180/213	44	68/128	71	168/200	46	36/203	87	38/72	58
b15_C	0/0	117	166/210	87	0/0	120	190/214	90	0/0	88	180/210	58	72/128	89	186/214	59	49/202	109	58/82	72
b20_C	0/0	155	172/203	115	0/0	156	182/200	117	0/0	116	191/211	77	67/128	118	188/209	77	44/201	136	62/86	93
b21_C	0/0	159	183/210	119	0/0	164	185/212	122	0/0	119	173/213	78	77/128	121	178/200	82	52/196	142	60/70	97
b22_C	0/0	231	180/209	172	0/0	233	185/212	175	0/0	177	190/205	116	86/128	175	189/199	116	48/192	193	56/70	139

In summary, RESAA offers a powerful new approach for attacking CLL-locked designs by partitioning the circuit and applying a combination of SAT-based, structural analysis, and/or query-based attacks. The results highlight the vulnerabilities in current LL techniques, even in advanced configurations like TTLock and SARLock combined with RLL. While these techniques provide better resistance to attacks, RESAA demonstrates that, with CG identification and proper partitioning, even these defenses can be overcome. The study emphasizes the need for further research into more robust locking mechanisms to counter the growing sophistication of attacks like RESAA.

4 Conclusions and Future Work

This thesis focuses on one of the most important fields of hardware security, i.e., LL techniques and their resiliency against existing attacks. As the complexity of ICs increases, so do the risks of IP theft, reverse engineering, and overproduction. LL, a heavily studied hardware obfuscation method, is crucial in addressing these threats by embedding secret keys that ensure proper circuit functionality only when the correct key is applied. Through rigorous experimentation and novel theoretical contributions, this thesis makes significant advancements in understanding LL techniques' vulnerabilities and potential enhancements.

The research presented in this thesis emphasizes the need for continuous threat evaluation in the field of hardware security. It introduces a novel query attack that exploits vulnerabilities in obfuscated FIR filters, revealing how attackers can use carefully crafted inputs to reveal the secret key. This attack exposes critical flaws in the current defenses and underscores the importance of continuously evaluating new threats in LL applications. A hybrid protection mechanism combining hardware obfuscation and LL was proposed to counter these vulnerabilities. This hybrid solution is more resilient than standalone LL techniques, mainly when dealing with parallel direct and transposed FIR filter designs. Experimental results showed that traditional LL methods had 100% key exposure, while the proposed hybrid protection had a 0% success rate of key extraction. This was at the expense of modest increases in area and power, typically between 1% and 5%, and up to approximately 21% in some instances of delay overhead. The effectiveness of this solution is demonstrated through comprehensive experimental results, revealing that hardware complexity remains competitive while security levels are significantly improved.

Second, the thesis introduces a resynthesis-based attack demonstrating how attackers can exploit structural weaknesses in locked circuits. This attack, applied under both OG and OL threat models, illustrates the limitations of traditional LL techniques, including SAT-resilient schemes. By leveraging commercial EDA tools, this attack identifies design vulnerabilities in the LL mechanisms and successfully compromises a significant number of key bits. For instance, experiments demonstrated 100% of key recovery from circuits like *c3540* locked with SKG-Lock after resynthesis. This work underscores the importance of considering the effects of synthesis tools in evaluating LL security and introduces new challenges for defending against these types of attacks.

Finally, the development of the RESAA framework represents a significant advancement in attacking CLL techniques. The framework was proposed to be an OA attack, where the adversary does not previously know the LL techniques applied in the CLL. It systematically identifies critical gates, applies structural analysis, and partitions the design, revealing specific weaknesses in the CLL technique. RESAA's ability to break down designs and efficiently target different layers of security emphasizes the complexities of securing modern ICs. For example, the average proportion of proven values of key inputs is approximately 41% under the OG attack, with execution times ranging from 8 to 2645 seconds. Under the OL attack, RESAA recovered a good portion compared to the SCOPE attack. For example, the attack reveals 37 out of 64 key bits for the *c2670* circuit locked with RLL + Anti-SAT. The experimental results demonstrate

the framework's ability to overcome even the most SAT-resilient defenses, providing valuable insights into how CLL techniques can be exploited. The findings call for more sophisticated methods that take into account the interactions between different LL techniques.

While this research has made significant progress in understanding and addressing hardware security challenges, it also emphasizes the need for continued innovation in this rapidly evolving field. Future work should focus on enhancing the resilience of LL techniques, particularly in the face of more sophisticated attacks. Integrating AI and ML tools into defensive strategies may offer new opportunities for improving LL's robustness. These technologies could detect attack patterns, predict vulnerabilities, and dynamically adapt the locking mechanisms to different attack scenarios.

Moreover, the interplay between high-level obfuscation methods, such as hardware watermarking and camouflaging, and LL should be further explored to develop a more robust and comprehensive defense strategy. Combining these techniques could enable layered protection, effectively addressing vulnerabilities at multiple stages of the IC supply chain. Additionally, the integration of novel materials offers promising opportunities for enhancing hardware security. This integrated approach creates multiple interdependent defense layers, significantly increasing the complexity of attacks and providing a resilient protective framework. These advances could redefine the way how LL is implemented, solidifying its role as a critical component in safeguarding future hardware designs against evolving threats.

Future work can investigate adapting the LL techniques presented in this thesis to emerging computing paradigms, such as neuromorphic architectures. These systems offer promising avenues for dynamic security by enabling LL implementations capable of real-time adaptation to evolving threats. Such adaptability can significantly enhance resistance to contemporary attacks. Investigating these novel architectures may lead to more resilient protection mechanisms and extend hardware security strategies to technologies with inherently different vulnerabilities.

In conclusion, as the semiconductor industry evolves, so must the security measures that protect it. This thesis demonstrates that while LL remains a powerful tool in the fight against IP theft and reverse engineering, it is far from secure. The contributions made here, including the novel attacks and hybrid defenses, represent critical steps forward in the ongoing effort to secure the integrity of IC designs. The challenges outlined in this work highlight the need for a dynamic and proactive approach to hardware security, ensuring that as attackers evolve their methods, defenders are equipped with the tools and strategies necessary to protect the core of modern technology.

References

[1] H.-C. Hung, Y.-C. Chiu, and M.-C. Wu, "Analysis of competition between idm and fabless–foundry business models in the semiconductor industry," *IEEE Transactions on Semiconductor Manufacturing*, vol. 30, no. 3, pp. 254–260, 2017.

[2] 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.

[3] C. Wiesen, S. Becker, M. Fyrbiak, N. Albartus, M. Elson, N. Rummel, and C. Paar, "Teaching hardware reverse engineering: Educational guidelines and practical insights," in *IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE)*, pp. 438–445, 2018.

[4] G. Zarrinchian, "A chip activation protocol for preventing ic recycling," *Microprocessors and Microsystems*, vol. 101, p. 104872, 2023.

[5] U. Guin, K. Huang, D. DiMase, J. M. Carulli, M. Tehranipoor, and Y. Makris, "Counterfeit integrated circuits: A rising threat in the global semiconductor supply chain," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1207–1228, 2014.

[6] T. M. Supon, M. Seyedbarhagh, R. Rashidzadeh, and R. Muscedere, "A method to prevent hardware trojans limiting access to layout resources," *Microelectronics Reliability*, vol. 124, p. 114212, 2021.

[7] T. D. Perez and S. Pagliarini, "A survey on split manufacturing: Attacks, defenses, and challenges," *IEEE Access*, vol. 8, pp. 184013–184035, 2020.

[8] H. Chakraborty and R. Vemuri, "Combined split manufacturing and logic obfuscation based on emerging technologies at high level for secure 3d ic design," in *IEEE 67th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1403–1407, 2024.

[9] F. Koushanfar and G. Qu, "Hardware metering," in *38th Annual Design Automation Conference (DAC)*, p. 490–493, 2001.

[10] F. Koushanfar, "Provably Secure Active IC Metering Techniques for Piracy Avoidance and Digital Rights Management," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 7, no. 1, pp. 51–63, 2012.

[11] G. Qu and L. Yuan, *Secure Hardware IPs by Digital Watermark*, pp. 123–141. Springer New York, 2012.

[12] A. B. Kahng, J. Lach, W. H. Mangione-Smith, S. Mantik, I. L. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe, "Watermarking Techniques for Intellectual Property Protection," in *Design Automation Conference (DAC)*, pp. 776–781, 1998.

[13] J. Zhang, "A practical logic obfuscation technique for hardware security," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 3, pp. 1193–1197, 2016.

[14] J. A. Roy, F. Koushanfar, and I. L. Markov, "EPIC: Ending Piracy of Integrated Circuits," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1069–1074, 2008.

[15] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security Analysis of Logic Obfuscation," in *Design Automation Conference (DAC)*, pp. 83–89, 2012.

[16] M. Yasin, B. Mazumdar, J. Rajendran, and O. Sinanoglu, "SARLock: SAT Attack Resistant Logic Locking," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 236–241, 2016.

[17] S. Dupuis and M.-L. Flottes, "Logic Locking: A Survey of Proposed Methods and Evaluation Metrics," *J. Electron. Test.*, vol. 35, no. 3, pp. 273–291, 2019.

[18] J. Zhou and X. Zhang, "Generalized sat-attack-resistant logic locking," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 16, pp. 2581–2592, 2021.

[19] V. S. Rathor, M. Singh, K. S. Sahoo, and S. P. Mohanty, "Gatelock: Input-dependent key-based locked gates for sat resistant logic locking," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 32, no. 2, pp. 361–371, 2024.

[20] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 137–143, 2015.

[21] H. M. Kamali, K. Z. Azar, F. Farahmandi, and M. Tehranipoor, "Advances in logic locking: Past, present, and prospects," *Cryptology ePrint Archive*, 2022.

[22] M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri, "On improving the security of logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 35, no. 9, pp. 1411–1424, 2016.

[23] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "Appsat: Approximately deobfuscating integrated circuits," in *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 95–100, 2017.

[24] 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 (TIFS)*, vol. 14, no. 10, pp. 2778–2786, 2019.

[25] M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran, "Removal attacks on logic locking and camouflaging techniques," *IEEE Transactions on Emerging Topics in Computing (TETC)*, vol. 8, no. 2, pp. 517–532, 2020.

[26] L. Aksoy, M. Yasin, and S. Pagliarini, "Kratt: Qbf-assisted removal and structural analysis attack against logic locking," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1–6, 2024.

[27] Y. Xie and A. Srivastava, "Anti-SAT: Mitigating SAT Attack on Logic Locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 38, no. 2, pp. 199–207, 2019.

[28] M. Yasin, B. Mazumdar, J. J. V. Rajendran, and O. Sinanoglu, "Ttlock: Tenacious and traceless logic locking," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 166–166, 2017.

[29] A. Rezaei, Y. Shen, and H. Zhou, "Rescuing logic encryption in post-sat era by locking & obfuscation," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 13–18, 2020.

[30] F. Almeida, L. Aksoy, and S. Pagliarini, "Resaa: A removal and structural analysis attack against compound logic locking," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 3, pp. 1–13, 2025.

[31] R. S. Chakraborty and S. Bhunia, "Harpoon: An obfuscation-based soc design methodology for hardware protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 28, no. 10, pp. 1493–1502, 2009.

[32] L. Aksoy, Q.-L. Nguyen, F. Almeida, J. Raik, M.-L. Flottes, S. Dupuis, and S. Pagliarini, "High-level intellectual property obfuscation via decoy constants," in *IEEE 27th International Symposium on On-Line Testing and Robust System Design (IOLTS)*, pp. 1–7, 2021.

[33] Z. Han, M. Yasin, and J. Rajendran, "Does Logic Locking Work with EDA Tools?," in *USENIX Security Symposium*, pp. 1055–1072, 2021.

[34] F. Almeida, L. Aksoy, Q.-L. Nguyen, S. Dupuis, M.-L. Flottes, and S. Pagliarini, "Resynthesis-based attacks against logic locking," in *24th International Symposium on Quality Electronic Design (ISQED)*, pp. 1–8, 2023.

[35] L. Aksoy, Q.-L. Nguyen, F. Almeida, J. Raik, M.-L. Flottes, S. Dupuis, and S. Pagliarini, "Hybrid protection of digital fir filters," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 31, no. 6, pp. 812–825, 2023.

[36] K. Shamsi, M. Li, K. Plaks, S. Fazzari, D. Z. Pan, and Y. Jin, "IP Protection and Supply Chain Security through Logic Obfuscation: A Systematic Overview," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 24, no. 6, 2019.

[37] X. Yuan, J. Weng, C. Wang, and K. Ren, "Secure integrated circuit design via hybrid cloud," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 29, no. 8, pp. 1851–1864, 2018.

[38] A. Kahng, J. Lach, W. Mangione-Smith, S. Mantik, I. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe, "Constraint-based watermarking techniques for design ip protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 20, no. 10, pp. 1236–1252, 2001.

[39] R. S. Chakraborty, S. Narasimhan, and S. Bhunia, "Hardware trojan: Threats and emerging solutions," in *IEEE International High Level Design Validation and Test Workshop (HLDVT)*, pp. 166–171, 2009.

[40] A. Basak, S. Bhunia, T. Tkacik, and S. Ray, "Security assurance for system-on-chip designs with untrusted ips," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 12, no. 7, pp. 1515–1528, 2017.

[41] M. R. Muttaki, S. Saha, H. M. Kamali, F. Rahman, M. Tehranipoor, and F. Farahmandi, "Rtlock: Ip protection using scan-aware logic locking at rtl," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1–6, 2023.

[42] S. Muzaffar and I. A. M. Elfadel, "Logic locking of finite-state machines using transition obfuscation," in *IFIP/IEEE 30th International Conference on Very Large Scale Integration (VLSI-SoC)*, pp. 1–6, 2022.

[43] T. Hoque, R. S. Chakraborty, and S. Bhunia, "Hardware Obfuscation and Logic Locking: A Tutorial Introduction," *IEEE Design & Test*, vol. 37, no. 3, pp. 59–77, 2020.

[44] IEEE Standards Association, "IEEE Standard for Encryption and Management of Electronic Design Intellectual Property (IP)." `https://standards.ieee.org/ieee/1735/7237/`, 2014. IEEE P1735-2014.

[45] R. S. Chakraborty and S. Bhunia, "Hardware protection and authentication through netlist level obfuscation," in *IEEE/ACM International Conference on Computer-Aided Design*, pp. 674–677, 2008.

[46] T. Perez, M. Imran, P. Vaz, and S. Pagliarini, "Side-channel trojan insertion - a practical foundry-side attack via eco," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2021.

[47] M. Nagata, T. Miki, and N. Miura, "Physical attack protection techniques for ic chip level hardware security," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, no. 1, pp. 5–14, 2022.

[48] R. Torrance and D. James, "The State-of-the-Art in IC Reverse Engineering," in *Cryptographic Hardware and Embedded Systems (CHES)*, pp. 363–381, 2009.

[49] Y.-C. Chen, W.-L. Wang, and M.-S. Hwang, "Rfid authentication protocol for anti-counterfeiting and privacy protection," in *The 9th International Conference on Advanced Communication Technology*, vol. 1, pp. 255–259, 2007.

[50] C. Profentzas, M. Günes, Y. Nikolakopoulos, O. Landsiedel, and M. Almgren, "Performance of secure boot in embedded systems," in *15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pp. 198–204, 2019.

[51] V. Lakafosis, A. Traille, H. Lee, G. Orecchini, E. Gebara, M. M. Tentzeris, J. Laskar, G. DeJean, and D. Kirovski, "An rfid system with enhanced hardware-enabled authentication and anti-counterfeiting capabilities," in *IEEE MTT-S International Microwave Symposium*, pp. 840–843, 2010.

[52] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," *IEEE Design & Test of Computers*, vol. 27, no. 1, pp. 10–25, 2010.

[53] S. Dupuis, P.-S. Ba, G. Di Natale, M.-L. Flottes, and B. Rouzeyre, "A novel hardware logic encryption technique for thwarting illegal overproduction and hardware trojans," in *IEEE 20th International On-Line Testing Symposium (IOLTS)*, pp. 49–54, 2014.

[54] C. Pilato, F. Regazzoni, R. Karri, and S. Garg, "Tao: techniques for algorithm-level obfuscation during high-level synthesis," in *Proceedings of the 55th Annual Design Automation Conference*, DAC '18, (New York, NY, USA), Association for Computing Machinery, 2018.

[55] K. Zamiri Azar, H. M. Kamali, S. Roshanisefat, H. Homayoun, C. P. Sotiriou, and A. Sasan, "Data flow obfuscation: A new paradigm for obfuscating circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 4, pp. 643–656, 2021.

[56] Z. U. Abideen, S. Gokulanathan, M. J. Aljafar, and S. Pagliarini, "An overview of fpga-inspired obfuscation techniques," *ACM Comput. Surv.*, vol. 56, no. 12, 2024.

[57] A. R. Desai, M. S. Hsiao, C. Wang, L. Nazhandali, and S. Hall, "Interlocking obfuscation for anti-tamper hardware," in *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, 2013.

[58] C. Pilato, A. B. Chowdhury, D. Sciuto, S. Garg, and R. Karri, "ASSURE: RTL Locking Against an Untrusted Foundry," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 7, pp. 1306–1318, 2021.

[59] L. Aksoy, M. Yasin, and S. Pagliarini, "Cac 2.0: A corrupt and correct logic locking technique resilient to structural analysis attacks," in *IEEE 25th Latin American Test Symposium (LATS)*, pp. 1–6, 2024.

[60] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security Analysis of Integrated Circuit Camouflaging," in *ACM Conference on Computer and Communications Security (ACM CCS)*, pp. 709–720, 2013.

[61] J. Mellor, A. Shelton, M. Yue, and F. Tehranipoor, "Attacks on logic locking obfuscation techniques," in *IEEE International Conference on Consumer Electronics (ICCE)*, pp. 1–6, 2021.

[62] M. Yasin, A. Sengupta, M. T. Nabeel, M. Ashraf, J. Rajendran, and O. Sinanoglu, "Provably-secure logic locking: From theory to practice," *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017.

[63] B. Shakya, X. Xu, M. Tehranipoor, and D. Forte, "CAS-Lock: A Security-Corruptibility Trade-off Resilient Logic Locking Scheme," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2020, no. 1, pp. 175–202, 2019.

[64] A. Sengupta, M. Nabeel, N. Limaye, M. Ashraf, and O. Sinanoglu, "Truly Stripping Functionality for Logic Locking: A Fault-Based Perspective," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 39, no. 12, pp. 4439–4452, 2020.

[65] Q.-L. Nguyen, M.-L. Flottes, S. Dupuis, and B. Rouzeyre, "On Preventing SAT Attack with Decoy Key-Inputs," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 114–119, 2021.

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

[67] A. Sengupta, N. Limaye, and O. Sinanoglu, "Breaking CAS-Lock and Its Variants by Exploiting Structural Traces," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2021, no. 3, p. 418–440, 2021.

[68] D. Sirone and P. Subramanyan, "Functional analysis attacks on logic locking," in *Design Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 936–939, 2019.

[69] N. Limaye, S. Patnaik, and O. Sinanoglu, "Valkyrie: Vulnerability assessment tool and attack for provably-secure logic locking techniques," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 17, pp. 744–759, 2022.

[70] A. Rezaei, Y. Shen, S. Kong, J. Gu, and H. Zhou, "Cyclic locking and memristor-based obfuscation against cycsat and inside foundry attacks," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 85–90, 2018.

[71] D. Divyanshu, R. Kumar, D. Khan, S. Amara, and Y. Massoud, "Logic locking using emerging 2t/3t magnetic tunnel junctions for hardware security," *IEEE Access*, vol. 10, pp. 102386–102395, 2022.

[72] H. Mardani Kamali, K. Zamiri Azar, K. Gaj, H. Homayoun, and A. Sasan, "Lut-lock: A novel lut-based logic obfuscation for fpga-bitstream and asic-hardware protection," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 405–410, 2018.

[73] M. John, A. Hoda, R. Chouksey, and C. Karfa, "Sat based partial attack on compound logic locking," in *Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, pp. 1–6, 2020.

[74] N. Limaye, S. Patnaik, and O. Sinanoglu, "Fa-sat: Fault-aided sat-based attack on compound logic locking techniques," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1166–1171, 2021.

[75] Y. Shen and H. Zhou, "Double dip: Re-evaluating security of logic encryption algorithms," in *Great Lakes Symposium on VLSI (GLSVLSI)*, p. 179–184, 2017.

[76] Y. Shen, A. Rezaei, and H. Zhou, "SAT-based Bit-Flipping Attack on Logic Encryptions," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 629–632, 2018.

[77] A. Raj, N. Avula, P. Das, D. Sisejkovic, F. Merchant, and A. Acharyya, "Deepattack: A deep learning based oracle-less attack on logic locking," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2023.

[78] A. Alaql, M. M. Rahman, and S. Bhunia, "SCOPE: Synthesis-Based Constant Propagation Attack on Logic Locking," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 8, pp. 1529–1542, 2021.

[79] D. Sisejkovic, F. Merchant, L. M. Reimann, and R. Leupers, "Deceptive logic locking for hardware integrity protection against machine learning attacks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 41, no. 6, pp. 1716–1729, 2022.

[80] D. Sisejkovic, L. M. Reimann, E. Moussavi, F. Merchant, and R. Leupers, "Logic locking at the frontiers of machine learning: A survey on developments and opportunities," in *IFIP/IEEE 29th International Conference on Very Large Scale Integration (VLSI-SoC)*, pp. 1–6, 2021.

[81] L. Alrahis, S. Patnaik, J. Knechtel, H. Saleh, B. Mohammad, M. Al-Qutayri, and O. Sinanoglu, "Unsail: Thwarting oracle-less machine learning attacks on logic locking," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 16, pp. 2508–2523, 2021.

[82] M. T. Rahman, S. Tajik, M. S. Rahman, M. Tehranipoor, and N. Asadizanjani, "The key is left under the mat: On the inappropriate security assumption of logic locking schemes," in *2020 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 262–272, 2020.

[83] H. Mardani Kamali, K. Zamiri Azar, K. Gaj, H. Homayoun, and A. Sasan, "Lutlock: A novel lut-based logic obfuscation for fpga-bitstream and asic-hardware protection," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 405–410, 2018.

[84] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri, "Fault analysis-based logic encryption," *IEEE Transactions on Computers (TC)*, vol. 64, no. 2, pp. 410–424, 2015.

[85] H. M. Kamali, K. Z. Azar, H. Homayoun, and A. Sasan, "Interlock: An intercorrelated logic and routing locking," in *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pp. 1–9, 2020.

[86] M. Yasin, C. Zhao, and J. J. Rajendran, "Sfll-hls: Stripped-functionality logic locking meets high-level synthesis," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–4, 2019.

[87] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "Cyclic obfuscation for creating sat-unresolvable circuits," in *Great Lakes Symposium on VLSI (GLSVLSI)*, p. 173–178, 2017.

[88] H.-Y. Chiang, Y.-C. Chen, D.-X. Ji, X.-M. Yang, C.-C. Lin, and C.-Y. Wang, "Looplock: Logic optimization-based cyclic logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 39, no. 10, pp. 2178–2191, 2020.

[89] J. B. Wendt and M. Potkonjak, "Hardware obfuscation using puf-based logic," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 270–271, 2014.

[90] Y. Zhang, P. Cui, Z. Zhou, and U. Guin, "Tga: An oracle-less and topology-guided attack on logic locking," in *Proceedings of the 3rd ACM Workshop on Attacks and Solutions in Hardware Security Workshop*, p. 75–83, 2019.

[91] W. Zeng, A. Davoodi, and R. O. Topaloglu, "Obfusx: Routing obfuscation with explanatory analysis of a machine learning attack," in *26th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 548–554, 2021.

[92] L. Alrahis, S. Patnaik, F. Khalid, M. A. Hanif, H. Saleh, M. Shafique, and O. Sinanoglu, "Gnnunlock: Graph neural networks-based oracle-less unlocking scheme for provably secure logic locking," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 780–785, 2021.

[93] F. Brglez and H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Circuits and a Targeted Translator in FORTRAN," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 663–698, 1985.

[94] F. Corno, M. Reorda, and G. Squillero, "Rt-level itc'99 benchmarks and first atpg results," *IEEE Design & Test of Computers (D&T)*, vol. 17, no. 3, pp. 44–53, 2000.

[95] B. Tan, R. Karri, N. Limaye, A. Sengupta, O. Sinanoglu, M. M. Rahman, S. Bhunia, D. Duvalsaint, R. D., Blanton, A. Rezaei, Y. Shen, H. Zhou, L. Li, A. Orailoglu, Z. Han, A. Benedetti, L. Brignone, M. Yasin, J. Rajendran, M. Zuzak, A. Srivastava, U. Guin, C. Karfa, K. Basu, V. V. Menon, M. French, P. Song, F. Stellari,

G.-J. Nam, P. Gadfort, A. Althoff, J. Tostenrude, S. Fazzari, E. Breckenfeld, and K. Plaks, "Benchmarking at the frontier of hardware security: Lessons from logic locking." arXiv preprint arXiv:2006.06806, 2020.

[96] K. Shamsi, T. Meade, M. Li, D. Z. Pan, and Y. Jin, "On the approximation resiliency of logic locking and ic camouflaging schemes," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 14, no. 2, pp. 347–359, 2019.

[97] L. Aksoy, A. Hepp, J. Baehr, and S. Pagliarini, "Hardware obfuscation of digital fir filters," in *2022 25th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, pp. 68–73, 2022.

[98] A. Alaql, "Scope." Available: `https://github.com/alaql89/SCOPE`.

[99] L. Li and A. Orailoglu, "Piercing Logic Locking Keys through Redundancy Identification," in *Design, Automation and Test in Europe Conference (DATE)*, pp. 540–545, 2019.

[100] K. Shamsi, "Netlist Encryption and Obfuscation Suite." `https://bitbucket.org/kavehshm/neos/src/master/`, 2021.

[101] L. Aksoy, "Qatt query attack." Available: `https://github.com/leventaksoy/qatt`.